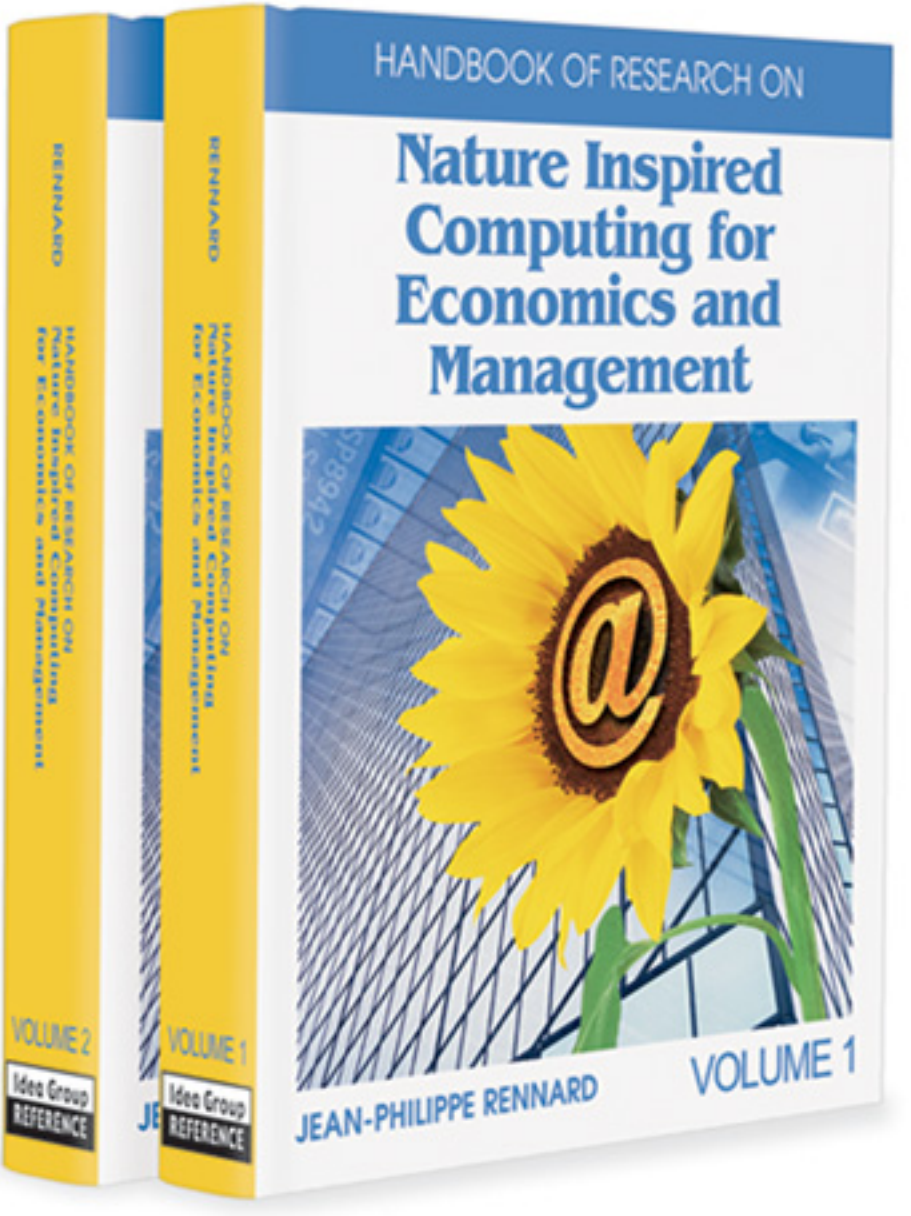

HANDBOOK OF RESEARCH ON
Nature Inspired Computing for Economics and Management
JEAN-PHILIPPE RENNARD
VOLUME 1
VOLUME 2
Idea Group REFERENCE

Handbook of Research on Nature-Inspired Computing for Economics and Management

Jean-Philippe Rennard
Grenoble Graduate School of Business, France

IDEA GROUP REFERENCE
Hershey · London · Melbourne · Singapore

Acquisitions Editor: Michelle Potter
Development Editor: Kristin Roth
Senior Managing Editor: Jennifer Neidig
Managing Editor: Sara Reed
Copy Editor: Maria Boyer
Typesetter: Diane Huskinson
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Idea Group Reference (an imprint of Idea Group Inc.)
701 E. Chocolate Avenue, Suite 200
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@idea-group.com
Web site: http://www.idea-group-ref.com

and in the United Kingdom by
Idea Group Reference (an imprint of Idea Group Inc.)
3 Henrietta Street
Covent Garden
London WC2E 8LU
Tel: 44 20 7240 0856
Fax: 44 20 7379 0609
Web site: http://www.eurospanonline.com

Library of Congress Cataloging-in-Publication Data

Handbook of research on nature inspired computing for economics and management / Jean-Philippe Rennard, editor.
p. cm.
Summary: "This book provides applications of nature inspired computing for economic theory and practice, finance and stock-market, manufacturing systems, marketing, e-commerce, e-auctions, multi-agent systems and bottom-up simulations for social sciences and operations management"--Provided by publisher.
ISBN 1-59140-984-5 (hardcover) -- ISBN 1-59140-985-3 (ebook)
1. Economics--Data processing. 2. Economics, Mathematical. 3. Management--Data processing. 4. Evolutionary computation. 5. Evolutionary programming (Computer science) I. Rennard Jean-Philippe.
HB143.5.H36 2007
330.0285'632--dc22

2006019134

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Editorial Advisory Board

To Nicolas Rennard

List of Contributors

Table of Contents

Volume II
Nature–Inspired Computing and Management

Section IV
Design and Manufacturing

Section VIII
Marketing

Section IX
Finance

Detailed Table of Contents

these areas. It finally points out some future directions for a bi-directional relationship between the social sciences and multi-agent systems research which hopefully will help researchers in both research areas, as well as researchers in management and organization theory.

The author builds an agent-based model wherein the societal corruption level is derived from individual corruption levels optimally chosen by heterogeneous agents with different risk aversion and human capital. The societal corruption level, in turn, affects the risk-return profile of corruption for the individual agents. Simulating a multi-generational economy with heterogeneous agents, The author shows that there are locally stable equilibrium corruption levels with certain socio-economic determinants.

This chapter pleads for more inspiration from human nature in agent-based modeling. As an illustration of an effort in that direction, it summarizes and discusses an agent-based model of the build-up and adaptation of trust between multiple producers and suppliers. The central question is whether, and under what conditions, trust and loyalty are viable in markets. The chapter explores a line of further research on the basis of notions of mental framing and frame switching on the basis of relational signaling, derived from social psychology.

This chapter proposes a more cognitively realistic approach to social simulation. It begins with a model for capturing the growth of academic science. Using this agent model, results comparable to human data are obtained. It is found that while different cognitive settings may affect the aggregate number of scientific articles produced by the model, they do not generally lead to different distributions of number of articles per author.

Economic models exhibit a multiplicity of behavior characteristics that are nonlinear and time-varying. ‘Emergent’ behavior appears when reduced order models of differing characteristics are combined to give rise to new behavior dynamics. In this chapter we apply the algorithms and methodologies developed for nature-inspired intelligent systems to develop models for economic systems.

sibilities, by a population of profit-seeking inventors. The key finding of the experiments is that if search heuristics are confined to those that are rooted in past experience, or to heuristics that merely generate variety, limited product advance occurs. The results demonstrate the importance of human direction and expectations in invention.

Volume II
Nature–Inspired Computing and Management

Section IV
Design and Manufacturing

The chapter introduces the concept of user-centric evolutionary design and decision-support systems, and positions them in terms of interactive evolutionary computing. This improved understanding can contribute to the iterative improvement of initial machine-based representations.

Modularity is widely used in system analysis and design such as complex engineering products and their organization, and modularity is also the key to solving optimization problems efficiently via problem decomposition. We first discover modularity in a system and then leverage this knowledge to improve the performance of the system. In this chapter, we tackle both problems with the alliance of organizational theory and evolutionary computation.

This chapter presents the notion of autonomous engineered systems working without central control through self-organization and emergent behavior. It argues that future large-scale applications from domains as diverse as networking systems, manufacturing control, or e-government services will benefit from being based on such systems. The goal of this chapter is to highlight engineering issues related to such systems, and to discuss some potential applications.

This chapter proposes an agent-oriented method for modeling and simulation of distributed production environments. The proposed method views a manufacturing enterprise as consisting of active entities—agents. The method makes use of the Radical Agent-Oriented Process (RAP) methodology introduced by Taveter and Wagner (2005) which is based on Agent-Object-Relationship (AOR) modeling. The method is aimed at the creation of environments for modeling and simulation of distributed manufacturing.

This chapter describes the application of the RAP/AOR methodology proposed by Taveter and Wagner (2005a, 2005b) to the modeling and simulation of a real ceramic factory. The chapter addresses the modeling of the ceramic factory from the interaction, information, and behavior aspects of the framework. The method is aimed at the creation of simulation environments and automation systems of distributed manufacturing.

This chapter examines the use of emergent computing to optimize solutions to logistics problems. These problems are based on real-world data in terms of geography and constraints. The author hopes that this chapter will inform researchers as to the suitability of emergent computing in real-world scenarios and the abilities of agent-based systems to mimic social systems.

The use of evolutionary computation is significant for the development and optimization of strategies for dynamic and uncertain situations. This chapter introduces three cases in which evolutionary computation has already been used successfully for strategy generation in the form of work on the Iterated Prisoners Dilemma, Rubinstein's Alternating Offers Bargaining Model, and the Simple Supply Chain Model. The authors hope that the chapter will promote this approach, motivate further work in this area, and provide a guide to some of the subtleties involved in applying evolutionary computation to different problems.

This chapter focuses on significant applications of self-organizing maps (SOMs), that is, unsupervised learning neural networks in two supply chain applications: cellular manufacturing and real-time management of a delayed delivery vehicle. Neural networks have and will continue to play a significant role in solving effectively complex problems in supply chain applications, some of which are also highlighted in this chapter.

Java Genetic Algorithm (JGA) is a computational object-oriented framework for rapid development of evolutionary algorithms for solving complex optimization problems. This chapter describes the JGA framework and illustrates its use on the dynamic inventory lot-sizing problem.

Two of the most complex activities in Productions and Operations Management (POM) are inventory planning and operations scheduling. This chapter presents two problems related to these activities, namely, the Capacitated Lot-Sizing and Scheduling Problem and the Capacitated Vehicle Routing Problem. The purpose of this chapter is to illustrate how to use JGA to model and solve complex business problems arising in POM.

The low price of coffee in the international markets has forced the Federación Nacional de Cafeteros de Colombia (FNCC) to look for cost-cutting opportunities. An alternative that has been considered is the reduction of the operating infrastructure by closing some of the FNCC-owned depots. These bi-objective optimization models are solved by means of NSGA II, a multi-objective evolutionary algorithm (MOEA). From a computational perspective, this chapter presents the Multi-Objective Java Genetic Algorithm (MO-JGA) framework, a new tool for the rapid development of MOEAs built on top of the Java Genetic Algorithm (JGA). We illustrate MO-JGA by implementing NSGA II-based solutions for the bi-objective location models.

The intentions of the chapter are twofold: firstly, to raise awareness of the importance of considering human performance variation in such simulation models; and secondly, to present some concep-

tual ideas for developing a multi-agent based approach for representing worker performance in manufacturing systems simulation models.

Section VI
Information Systems

This chapter presents a meta-model of information systems as a foundation for the methodology of caste-centric agent-oriented software development, which is suitable for applications on the Internet/ Web platform and the utilization of mobile computing devices. In the model, the basic elements are agents classified into a number of castes. This chapter also illustrates the advantages of agent-oriented information systems by an example.

Based on the meta-model of information systems presented by Zhu this year, this chapter presents a caste-centric agent-oriented methodology for evolutionary and collaborative development of information systems. The features of agent-oriented information systems in general and our methodology in particular are illustrated by an example throughout the chapter.

This chapter describes the application of self-organizing principles to the field of e-learning. It argues that traditional managed approaches to e-learning suffer from deficiencies both in cost and adaptativity that are addressed through the application of nature-inspired processes such as stigmergy and evolution. The chapter describes some example applications and explores some of the remaining challenges in the field, most notably in encouraging pedagogically useful structures to evolve.

In this chapter we describe a mechanism to search for resources in unstructured peer-to- peer (P2P) networks using ant algorithms implemented through software agents. Traditional resource search algorithms in P2P networks use an uninformed or blind search among the various nodes of the network. We describe and compare different reinforcement strategies used by ants to perform efficient resource search in P2P networks.

Section VII
Commerce and Negotiation

Work to date on negotiation protocols has focused almost exclusively on defining contracts consisting of one or a few independent issues and a relatively small number of possible contracts. Many real-world contracts, by contrast, are much more complex, consisting of multiple interdependent issues and intractably large contract spaces. This chapter describes a simulated annealing-based approach appropriate for negotiating such complex contracts that achieves near-optimal social welfare for negotiations with binary issue dependencies.

This chapter describes a generic multi-issue negotiating agent that is designed for a dynamic information-rich environment. The agent strives to make informed decisions by observing signals in the marketplace and by observing general information sources including news feeds.

The aim of this chapter is to give a background about auction theory and to present how evolutionary computation techniques can be applied to auctions. Finally, an explained example shows how a genetic algorithm can help automatically find bidders' optimal strategies for a specific dynamic multi-unit auction—the Ausubel auction—with private values, drop-out information, and with several rationing rules implemented. The results suggest that the approach leads to strategies that outperform sincere bidding when rationing is needed.

This chapter exposes our work towards the development of an agent-based electronic institution (EI) providing a virtual normative environment that assists and regulates the creation and operation of VOs, through contract-related services. It includes a presentation of the EI framework, knowledge representation structures for norms in contracts, and a description of two main institutional services, namely negotiation mediation and contract monitoring.

Section VIII
Marketing

Using empirical market data from brand rivalry in a retail ground-coffee market, we model each idiosyncratic brand's pricing behavior using the restriction that marketing strategies depend only on profit-relevant state variables, and use the genetic algorithm to search for co-evolved equilibria, where each profit-maximizing brand manager is a stimulus-response automaton, responding to past prices in the asymmetric oligopolistic market. It is part of a growing study of repeated interactions and oligopolistic behavior using the GA.

Many new products fail, despite preliminary market surveys having determined considerable potential market share. This effect is too systematic to be attributed to bad luck. We suggest an explanation by presenting a new percolation theory model for product propagation, where agents interact over a social network.

In this chapter we discuss the use of agent-based modeling to produce decision support tools to enhance this understanding. We use the example of modeling opinion diffusion within a customer population and its effect on product adoption to illustrate how the agent-based modeling technique can be an ideal tool to create models of complex socioeconomic systems. We consider the advantages compared to alternative, more conventional approaches available to analysts and management decision makers.

Section IX
Finance

This chapter compares forecasts of the median neighborhood prices of residential single-family homes in Cambridge, Massachusetts, using parametric and nonparametric techniques. Prices are

measured over time (annually) and over space (by neighborhood). To demonstrate their efficacy, forecasts of the median prices are first obtained using a standard statistical method: weighted least squares. Genetic programming and neural networks are then used to produce two other forecasts.

This chapter introduces the capability of the numerical multidimensional approach to solve complex problems in finance. The authors with the examples in the chapter would like to demonstrate how a multidimensional approach based on the mimic of nature could be useful to solve modern complex problems in finance.

This work addresses a real-world adjustment of economic models where the application of robust and global optimization techniques is required. The problem dealt is the search of a set of parameters to calculate the reported claim amount. Several functions are proposed to obtain the reported claim amount, and a multi-objective optimization procedure is used to obtain parameters using real data and decide the best function to approximate the reported claim amount. Results show the advantages of MOEAs in the proposal in terms of effectiveness and completeness in searching solutions, compared with particular solutions of classical EC approaches (using an aggregation operator) in problems with real data.

This chapter presents an artificial stock market created to analyze market dynamics from the behavior of investors. It argues that information—delivered by financial intermediaries as rating agencies and considered as cognitive institution—directs the decisions of investors who are heterogeneous agents endowed with capabilities of learning in a changing environment. The objective is to demonstrate that information influences market dynamics as it allows the coordination of the decisions of investment in the same direction: information is a focal point for investors and contributes to generate a speculative dynamic on the market.

Foreword

A CURE FOR THE DISMAL SCIENCE

I earn a living solving real economic and strategic problems for real businesses. After a number of years of this regimen, I have lost my taste for big theories of human behavior—especially economic theories. My version of game theory consists of playing the game and forgetting about the theory. That's because my incentive system is based not on the aesthetics of theorem proving, but rather on my ability to ease the pain of suffering managers. I no longer derive any satisfaction from irrelevant elegance. The foundations of economics have for so many years ignored the realities of human behavior and decision making that it has become a joke for business practitioners, managers, and consultants. I believe that the mutual distaste of practitioners and theorists is coming to an end as a new breed of economists is emerging. These new economists embrace the complexities and subtleties of human behavior; they acknowledge the dynamic, evolving nature of the economy; they design economic experiments that involve, God forbid real people; while they do not reject mathematics as a tool, they do not view it as a purpose; they believe that computational experiments can take them beyond confined, provable situations. The handbook that Jean-Philippe Rennard has assembled is a wonderfully diverse collection of points of view from this new breed of economists and social scientists, a vibrant cross-section of the field of economics as I hope it will evolve in the near future.

The main thread throughout this collection of essays is human behavior, individual or collective, and how it can be understood, modeled, approximated, or even enhanced using a range of techniques and approaches from "complexity science." Evolutionary algorithms, co-evolution, swarm intelligence, social networks, multi-objective decision making, and agent-based modeling are some of the techniques employed. That there is a need for such approaches is crystal clear from the viewpoint of practical applications. Let me use some examples from consumer behavior and marketing to illustrate why in particular agent-based modeling (ABM) is the keystone of the new computational edifice.

When a customer makes a purchase or switch decision, it is often the result of a history. Impulse decisions, while they do happen, are the exception rather than the rule. That does not mean that most decisions are rational, simply that they cannot be explained by just looking at the time they happen. When a wireless phone customer decides to switch carriers, such a decision is the result of all the interactions and experiences this customer had with his carrier as well as with other sources of information. Failing to recognize the temporal dimension of decision making can lead to dramatic prediction errors. ABM, and only ABM, can explicitly deal with all aspects of time: learning, waiting, simmering, habituation, forgetting, and so forth. For example, in the casino industry, common

wisdom holds that customers have a fixed budget and stop playing when their budget is exhausted. An ABM fed with real slot data from a loyalty card program showed that in reality customers stop playing when their total experience over time (TEOT—a combination of the dynamics of their wins and losses weighted by demographic attributes and day of the week, and, yes, budget) reaches a threshold. TEOT is a much better predictor than budget or any combination of demographic attributes which enables a major casino owner and operator to implement effective real-time marketing and promotional offers. Of course the dirty little secret is data and how to use it effectively to estimate complex time-dependent models of decision making. When the data exists, in the absence of a coherent theoretical framework, not to mention theorems, one has to perform rigorous computational experiments based on statistical machine learning techniques.

Another example is health insurance, where a customer's demographic attributes are not sufficient to predict which plan he or she will select. Instead, the characteristics of each plan are viewed through a looking glass that puts more weight on certain characteristics as a function of the customer's experience with his current plan, which is a combination of his and his family's health in the past year and satisfaction or dissatisfaction with the health care afforded by the plan. Furthermore, if specific adverse health events have happened in the recent past, they strongly affect the way the possibility of catastrophic losses is perceived. By using an ABM that explicitly deals with the effects of experience and recency, prediction error could be reduced by an order of magnitude at Humana, a leading U.S. health insurer. No amount of traditional econometric modeling with demographic attributes as explanatory variables would have been able to achieve this level of accuracy.

In retail, the layout of a supermarket is known to be a key sales driver, yet shopper behavior is an emergent property with a strong spatio-temporal component that is never taken into account in traditional econometric modeling: while the trajectory of a shopper in a supermarket is influenced by the shopper's shopping list, the trajectory in turn influences what the shopper buys beyond the shopping list. Through the use of a spatial behavioral model of shoppers in a supermarket, Pepsi was able to predict hot spots in any supermarket as a function of the supermarket's layout and the demographic attributes of its shopper population. With the knowledge of hot spot locations, Pepsi could determine the best location not only for its products, but also for promotional signs. Here again, the dirty little secret is data and how to use it. Not only did we have to develop special estimation techniques to infer trajectories and reconcile them with scanner data, data collection itself was a challenge: shoppers were given "smart carts" equipped with tags for path tracking.

Customers experience, learn, adapt, adjust. Their decisions are *path dependent*: in other words, decisions are dependent upon a contingent history. Existing statistical or econometric techniques do not deal satisfactorily with path dependence. When done properly (and that's a big IF) ABM combines the statistical rigor of existing techniques with the ability to accurately model the temporal components of decision making. As a result, not only does predictability go through the roof, the outputs of what-if scenarios also become more reliable because behavioral models are fundamentally causal rather than correlational. Knowing that two variables are correlated is good enough to predict the past, but robustly predicting the future requires understanding the underlying causal mechanisms of decision making.

At the risk of repeating myself, taking data seriously is the dirty little secret of success. We must not lose sight of data in the excitement of playing with our synthetic little worlds. It is ok for the theory to be ahead of the data, but not by light-years. A case in point is the over-theorization of social networks in the last few years. In my experience good social network data (whatever that means) is a rarity. The data is often inadequate, ranges from incomplete to sparse, is noisy, sensitive to minute details, and lacks such important characteristics as frequency, quality, and nature of the interactions. In other words it is unusable in practice for predictive purposes. For example, in 1954 pharmaceutical giant Pfizer was interested in determining how physicians decide to adopt a new drug so that it could more effectively market its products through detailing and traditional media. By knowing how physicians acquire reliable information and who they trust, Pfizer could market its new drugs more effectively, optimizing the allocation of marketing resources among detailing, media advertisement, continuing medical education, and so forth. They funded a landmark social network study aimed at showing the effect of interpersonal influences on behavior change in relation to the adoption of Tetracycline, a powerful and useful antibiotic just introduced in the mid-1950s. Pfizer hoped tetracycline would diffuse rapidly because it was a tremendous improvement over existing antibiotics. The Pfizer-funded study contained two major advances over previous studies in that it relied on a behavioral measure of time of adoption by looking at prescription records and used network analysis to identify opinion leaders. However, numerous subsequent studies of this work revealed a number of weaknesses in the collection and analysis of the data, and the consensus among social network scientists today is that the study is inconclusive: the uptake in tetracycline adoption cannot be assigned with confidence to social network effects. Over the last fifty years, in a movement that accelerated over the last ten, social network researchers have been developing more and more complex models of network diffusion, but there is very little data to back them up; there is a lot of "anecdotal" evidence, a euphemism for poor statistical work on ambiguous data.

One of the issues facing those who want to study the influence of social networks on the diffusion and adoption of innovations to design marketing interventions is the lack of reliable data. There are situations, however, where the community of adopters is sufficiently small that it can be mapped accurately. My team and I studied the adoption of a new drug used exclusively in intensive care units (ICUs), where the number of individuals (doctors, nurses, pharmacologists) involved in the decision to prescribe the drug is between 10 and 20 in a given hospital. The study revealed that the temporal structure of the social network is the key to prescription behavior. While a snapshot of the social network is unhelpful (we found no correlation between such snapshots and probabilities of adoption), its dynamics over time are a great predictor of the speed with which the drug is adopted: this can be explained by the fact that in many hospital ICUs, physicians work only a few months per year and teach or work in other departments for the rest of the year, so that the only opportunities physicians have to interact is when their assignments overlap for a few days. We discovered that the degree of overlap correlates positively with the speed of adoption, suggesting that ICUs that are organized to provide more overlap between physicians are more favorable marketing targets. Promoting the drug to these ICUs first accelerates adoption. ICUs that are more difficult to penetrate can be targeted in a second marketing wave, as it is easier to market a product when you can tell your customers that their competitors or peers are already using it.

So, where does that leave us? Clearly, from the perspective of a practitioner, there is a need for the various approaches advocated in this book's chapters. The authors are leading the way in defining a cure for the dismal science. I am convinced that it is only by combining these new approaches with a relentless focus on data and reality that the cure will gain credibility.

Eric Bonabeau
Icosystem Corporation, Cambridge, MA
January 17, 2006

Eric Bonabeau is the chief scientist at Icosystem Corporation, a U.S.-based "idea incubator" that uses behavioral modeling to discover business opportunities and invent the technology to support them. Prior to his current position, Dr. Bonabeau was the CEO of Eurobios, a joint venture with Cap Gemini Ernst & Young applying the science of complex adaptive systems to business issues. He has been a research director with France Telecom R&D, an R&D engineer with Cadence Design Systems, and an interval research fellow at the Santa Fe Institute. He is the co-author of more than 100 science articles and three books: *Intelligence Collective* (Hermès Sciences, 1994); *Swarm Intelligence in Natural and Artificial Systems* (Oxford University Press, 1999); and *Self-Organization in Biological Systems* (Princeton University Press, 2001). Dr. Bonabeau is also co-editor-in-chief of *Advances in Complex Systems* and *ACM Transactions in Adaptive and Autonomous Systems*. He is a member of the editorial and scientific committees of more than 25 international journals and conferences. He graduated from Ecole Polytechnique, France, holds a telecommunications engineering degree from Telecom Paris, and a post-graduate degree in applied mathematics and a PhD in theoretical physics, both from Paris X University. Some of his recent business publications include "Swarm Intelligence: A Whole New Way to Think About Business" (*Harvard Business Review,* May 2001), "Predicting the Unpredictable" (*Harvard Business Review,* March 2002), "Don't Trust Your Gut" (*Harvard Business Review,* May 2003), "The Perils of the Imitation Age" (*Harvard Business Review,* June 2004), "You Heard It Here First" (*Harvard Business Review,* February 2005), and "What Is Luxury Without Variety" (*Harvard Business Review,* February 2006).

Preface

WHY SOCIAL SCIENCES NEED NATURE-INSPIRED COMPUTING

For nearly four billions years, life has invaded Earth. Throughout eras geological upheavals have deeply transformed the environment. Here, deserts left place to a tropical environment; there ices were replaced by conifers forests …. Despite these transformations, life is so vigorous that one can find it on the top of the highest mountains, in the depths of oceans, as well as in clouds or deep underground.

Life has proved its capacity to cope with deep transformations—its ability to overcome mass extinctions and to rebirth after disasters. These adapting and surviving capabilities have inspired computer scientists who try to conceive algorithms imitating nature, hoping to confer in them some of the aptitudes of living beings.

Darwinian natural adaptation has been transferred into *evolutionary algorithms; artificial neural networks* are a metaphor of nervous systems; ants foraging behaviors gave rise to *ant colony optimization;* birds flocks or fish schools inspired *particle swarm optimization; artificial immune systems* mimic the biological ones; insect or animal autonomy and abilities inspired *distributed artificial intelligence, multi-agent systems,* and *artificial societies*.

In the fields of social sciences, economics, and management, two types of contributions must be emphasized:

- Social insects, buffalo herds, or human societies show that social life dominates the macrofauna. This fascinating characteristic of living systems, and more particularly the study of social insect behavior, inspired the rebirth of artificiality. Starting with artificial life and artificial intelligence, modern artificiality now reaches social sciences with the development of artificial societies which contributes to renewed approaches of social and economic phenomena.
- Living systems are supple and able to adapt to huge transformations in their environment. Transposition of these properties into algorithms provide with ground-breaking tools able to deal with complex problems.

After a six-chapter introduction to nature-inspired computing for modeling and optimization, the first volume of the handbook is oriented toward social sciences (sociology and economics) modeling and experiments; the second volume mainly handles modeling, exploration, and optimization for management.

VOLUME I

The core hypothesis of artificial societies is that human societies are *complex adaptive systems* (CASs) whose properties emerge from non-linear interactions between their members. Since the famous *Jurassic Park* by Steven Spielberg, everyone has been aware of the existence of "chaos" and "complexity." Despite the highly disputable treatment of chaos in that movie, it addressed a core problem of modern science that Nobel Prize recipient Illya Prigogine nicely termed "The End of Certainty" (Prigogine, 1997).

At least since the French philosopher René Descartes in the 17th century, science has mainly been based on *analysis*. Analysis tries to understand reality by breaking down observed phenomena, thanks to the fundamental hypothesis that a global process is the result of the mere (linear) aggregation of its components.

At the end of the 19th century, French mathematician Henri Poincaré proved that analysis is unable to describe most systems based on *non-linear* interactions. Those systems are now said to be *more than the sum of their parts* (Waldrop, 1992). The resulting complex dynamic is unpredictable. The only way to know their state at a given step is to compute each step. The usual analytical method is of little help; the necessary mathematics are still to be invented. Non-linearity thus challenges the traditional approach, which tries to understand a system by analyzing its components: "The key feature of non-linear systems is that their primary behaviors of interest are properties of the *interactions between parts,* rather than being properties of the parts themselves, and these interactions-based properties necessarily disappear when the parts are studied independently" (Langton, 1989, p. 41, original italics). How do we study processes that are "more than the sum of their parts?" How do we analyze properties that cannot be forecasted? The solution proposed by computer scientists is termed *bottom-up modeling*.

Since core properties disappear when the components are studied independently, bottom-up modeling is based on the gathering of interacting components. Such constructions and the study of the dynamic resulting from non-linear interactions of the simple components constitute the "bottom-up method." Instead of modeling the global dynamic of the studied system ("top-down method" usually based on differential equations), one merely models the components to study the potentially emerging regularities.

Bottom-up modeling is based on *multi-agent systems* (MASs). Agents are a kind of "living organism," whose behavior—which can be summarized as communicating—and acting are aimed at satisfying its needs and attaining its objectives (Ferber, 1999, p. 10). MASs are a gathering of interacting agents. In social sciences (mainly sociology and economics), they are the base of artificial societies.

Section II: Social Modeling includes seven chapters providing a global view of this research. Chapters by Robert Axelrod and Harko Verhagen demonstrate the huge potential contribution of artificial societies to social sciences. Corruption, trust, and academic science are then studied in the light of MASs, showing the cross-fertilization of social sciences and multi-agent systems.

Section III: Economics includes thirteen chapters providing global coverage of the use of nature-inspired computing for economics. After an introduction to agent-based computational economics (ACE), original research using multi-agent systems, evolutionary algorithms, or neural networks to deal with fundamental economic forces are presented. Clusters, innovation, and technol-

ogy are then particularly emphasized to enlighten the complex cross dynamics of space and technology.

VOLUME II

Management is confronted with challenges of growing difficulty:

- The complexity of the environment, the unprecedented speed of its evolution, and the unmanageable resulting mass of information require specific powerful tools.
- The never-ending search for productivity has made optimization a core concern for engineers. Quick process, low energy consumption, and short and economical supply chains are now key success factors.

The six-section, thirty-two-chapter second volume provides comprehensive coverage of the contributions of nature-inspired computing to management. It shows its ability to solve problems beyond the capabilities of more traditional methods.

Section IV: Design and Manufacturing presents pioneering research, particularly using evolutionary algorithms. Applied to design, project management, as well as to manufacturing, this research clearly demonstrates the capacity of nature-inspired algorithms to stimulate design creativity and to manage complex associated issues.

Section V: Operations and Supply Chain Management contains twelve chapters. After an introduction to evolutionary optimization and ant colony optimization for operations management, main nature-inspired tools are used to solve very diverse operations and supply chain management problems (scheduling, organization of production, distribution, etc.). The section includes the presentation of a powerful Java framework designed to use evolutionary computation to solve operations and supply chain problems.

Section VI: Information Systems presents the novel agent-oriented paradigm of information systems and provides innovative research, demonstrating the power and suppleness of nature-inspired computing when applied to information management, e-learning, and peer-to-peer systems.

Section VII: Commerce and Negotiation includes a synthesis of agents for multi-issue negotiation, and presents original research on automatic negotiations and auctions using agent-based modeling and evolutionary computation. This research outstandingly leads the way toward future virtual organizations.

Section VIII: Marketing uses evolutionary computation and agent-based modeling to analyze price wars and word-of-mouth, and to contribute to the understanding of complex socio-economic systems to provide a decision support tool for commercial organizations.

Section IX: Finance uses genetic programming, evolutionary computation, neural networks, and agent-based modeling to deal with complex financial problems. They are applied to housing prices, financial decision aid, insurance-linked derivatives, and stock-market simulations.

The fifty-eight chapters of this two-volume handbook provide a unique cross section of research using nature-inspired computing for economics and management. Social scientists, economists, and people dealing with management science will find both an introduction and a valuable presentation of state-of-the-art research in these fields, giving them a unique reference tool for their own research. Students in computer sciences, social sciences, and management will find all the necessary material to master the field and to help them in their training. Managers, engineers, and practitioners will find a great deal of efficient and powerful tools to help them solve difficult problems, and to anticipate the use of tools that will undoubtedly be part of tomorrow's key success factors.

Jean-Philippe Rennard
Grenoble Graduate School of Business, France
March 15, 2006

REFERENCES

Ferber, J. (1999). *Multi-agent systems.* London: Addison-Wesley.

Langton, C. G. (1989). Artificial life. In C. G. Langton (Ed.), *Artificial life 1*. Redwood City, CA: Addison-Wesley.

Prigogine, I. (1997). *The end of certainty.* New York: The Free Press.

Waldrop, M. (1992). *Complexity: The emerging science at the edge of chaos.* New York: Simon and Schuster.

Acknowledgment

The editor would like to acknowledge the help of all involved in the collation and review process of the handbook, without whose support the project could not have been satisfactorily completed. A further special note of thanks goes also to all the staff at Idea Group Inc., whose contributions throughout the whole process from inception of the initial idea to final publication have been invaluable.

Nearly 100 researchers were involved in the review process of this handbook. I'd like to particularly thank for their comprehensive, critical, and constructive comments: Aguirre, Arturo Hernandez Aguirre, Sylvie Blanco, David Catherine, Jean-Jacques Chanaron, Mattia Ciprian, Benédicte Daudé, Raffi Duymedjian, Juan Flores, Robert Goldberg, David Gotteland, Serge Hayward, Francesco Herrera, Karim Hirji, Peter Keenan, Jerzy Korczak, Bartoz Kozlowski, Michel Leseure, Gabriel Lopardo, Hervé Luga, Gonzalo Mejía, Ronaldo Menezes, Emilio Migueláñez, Philippe Naccache, Akira Namatame, Baikunth Nath, Godfrey Onwubolu, Zengchang Qin, Bala Ram, Michal Ramsza, Art Sedighi, Franciszek Seredynski, Michele Sonnessa, Flaminio Squazzoni, Utku Ünver, and Zhao Wei Zhong.

Special thanks also go to the publishing team at Idea Group Inc., in particular to Kristin Roth, who continuously prodded via e-mail for keeping the project on schedule and to Jan Travers and Mehdi Khosrow-Pour, whose enthusiasm motivated me to initially accept his invitation for taking on this project.

I express my gratitude to Eric Bonabeau for his nicely provocative foreword.

In closing, I wish to thank all of the authors for their insights and excellent contributions to this handbook. I also want to thank all of the people who assisted me in the review process. Finally, I want to thank my wife and children for their love and support throughout this project.

Jean-Philippe Rennard
Grenoble Graduate School of Business, France

Volume I

Nature-Inspired Computing and Social Sciences

Section I

Nature-Inspired Computing

Chapter I
Artificiality in Social Sciences

Jean-Philippe Rennard
Grenoble Graduate School of Business, France

ABSTRACT

This chapter provides an introduction to the modern approach of artificiality and simulation in social sciences. It presents the relationship between complexity and artificiality, before introducing the field of artificial societies which greatly benefited from the fast increase of computer power, gifting social sciences with formalization and experimentation tools previously owned by the "hard" sciences alone. It shows that as "a new way of doing social sciences," artificial societies should undoubtedly contribute to a renewed approach in the study of sociality and should play a significant part in the elaboration of original theories of social phenomena.

INTRODUCTION

The "sciences of the artificial" deal with synthesized things which may imitate natural things, which have functions and goals and which are usually discussed in terms of imperatives as well as descriptives. Imitation with computer is now usually termed simulation and is used to understand the imitated system (Simon, 1996).

Artificiality has invaded science over the last thirty years, and physicists, chemists, or biologists now daily use widespread computing tools for simulations. Social sciences did not set this trend aside (Halpin, 1999). This chapter will first introduce the essential link between complexity and artificiality before presenting the highly promising field of artificial societies.

Complexity and Artificiality

Since the seminal book of Herbert Simon in 1969 (Simon, 1996), the sciences of the artificial knew a jerky evolution. In the field of artificial intelligence, the excessive ambitions of the sixties were considerably lowered in the seventies, before knowing a new wave of opti-

mism in the mid-eighties. The renewed interest toward artificiality originates in new approaches of artificial intelligence and in the success of the highly innovative related fields of *artificial life* (Langton, 1989) and *artificial societies* (Gilbert & Conte, 1995; Epstein & Axtell, 1996). Artificial life is at the crossroad of the rebirth of artificiality and offers many nice examples illustrating this revival, like this one:

Many ant species tend to form piles of corpses (cemetery) in order to clean their nest. Experiments with different species showed that if corpses are randomly distributed, ants tend to gather them in some clusters within a few hours.

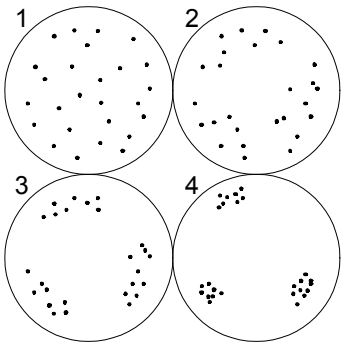
Deneubourg et al. (1991) proposed a simple model of corpses gathering (see also Bonabeau, Dorigo, & Théraulaz, 1999). They designed virtual ants having the following behaviors:

- The probability for an ant to pick up a corpse is $p_p=(k_1/(k_1+f))^2$ with k_1 a threshold constant and f the fraction of perceived corpses in the neighborhood.
- The probability for an ant to deposit a corpse is: $p_d=(f/(k_2+f))^2$ with k_2 a threshold constant. Deneubourg et al. (1991) computed f as the number of items perceived during the last t periods divided by the largest number of items that can be encountered during the last t periods.

To put it simply, virtual ants tend to pick up isolated corpses to drop them in dense zones. The result (see Figure 1) is close to the real phenomenon.

Highly simple virtual individuals ("agents") without any knowledge of the global process manage to carry out cemetery building. Furthermore, it "suffices" to define different types of objects to obtain sorting capabilities, like for example larval sorting observed in anthills. The gathering or the sorting process *emerges* from the interactions of simple agents.

Figure 1. Virtual ant cemetery



Emergence

Emergence can be defined as the qualities or properties of a system which are new compared with the qualities or properties of the components isolated or differently organized (Morin, 1977). According to Gilbert (1995b):

Emergence occurs when interactions among objects at one level give rise to different types of objects at another level. More precisely, a phenomenon is emergent if it requires new categories to describe it that are not required to describe the behavior of the underlying components. For example, temperature is an emergent property of the motion of atoms. An individual atom has no temperature, but a collection of them does.

Most authors consider that emergence relies on three conditions:

1. The global process is distributed, there is no central control, and the result depends on the interactions between components.
2. The process is autonomous, there is no external controller.

3. The process is not at the same *level* as the components. The language or concepts used to describe the emergent process are different from the language or concepts used to describe the components. The "test of emergence" thus relies on the *surprise* engendered by the difference between the language L_1 used to design components, and the language L_2 used to describe the resulting process (Ronald, Sipper, & Capcarrère, 1999). According to Steels (1997), this change of language is sufficient to characterize emergence.

We can clarify this concept with the classical and very general formalization proposed by (Baas, 1994), which is based on three elements:

1. A set of *first-order structures* $\{S^1_{i_1}\}, i_1 \in J_1$ with J_1 some index set finite or not. First-order structures are primitive objects, abstract, or physical; they can be organizations, machines, as well as fields or concepts.
2. An *observational* mechanism *Obs*.
3. *Interactions Int*.

The new kind of structure resulting from the observed interactions of first-order structures is: $S^2 = R(S^1_{i_1}, Obs^1, Int^1)_{i_1 \in J_1}$, where R is the result of the interaction process and $Obs^1 \equiv Obs(S^1_{i_1})$. Baas calls S^2 a *second-order structure*, and $\{S^2_{i_2}\}, i_2 \in J_2$ *families*.

The properties of the new unity resulting from the collection S^2 can be measured with an observational mechanism Obs^2. Then P is an *emergent property* of S^2 if $P \in Obs^2(S^2)$, but $P \notin Obs^2(S^1_{i_1})$ for all i_1.

The property P belongs to the emergent structure S^2, but is absent from the components.

The S^2s can interact to form a third-order structure and so on. An *N-th-order structure* is then:

$$S^N = R(S^{N-1}_{i_{N-1}}, Obs^{N-1}, Int^{N-1}), i_{N-1} \in J_{N-1}$$

Baas calls it a *hyperstructure* and he considers that "complexity often takes the form of a hyperstructure" (Baas, 1994, p. 525, original italics). According to Simon (1996), hierarchies are necessary to allow the evolution of complex structures.

Baas distinguishes two different types of emergence:

- **Deductible or Computable Emergence:** A process or theory D exists which allows to determine $P \in Obs^2(S^2)$ from $(S^1_{i_1}, Obs^1, Int^1)$. That is typically the case of engineering constructions or "trivial emergence" like temperature evoked above.
- **Observational Emergence:** The emerging property P cannot be deduced (e.g., consequences of Gödel's theorem).

Bedau (1997) considers less drastically that *weak emergence* characterizes emerging properties that can only be derived by simulation. Most of the recent modeling works deal with this type of weak emergence (Chalmers, 2002).

Despite the thousands of pages published on emergence, or the recent emphasis on the *reduction principle* (the macro-behavior is reducible to the interactions of the components) (e.g., Holland, 1999; Kubik, 2003), we are still far from an ontological concept of emergence (Emmeche, Koppe, & Stjernfelt, 1997), but considering its success, emergence is undoubtedly epistemologically a fertile concept.

Bottom-Up Modeling

Emergence is a key feature of those famous *non-linear systems* which are said to be *more than the sum of their parts* (Waldrop, 1992). Non-linear systems do not obey the *superposi-*

tion principle. Their dynamic cannot be reduced to the simple (linear) combination of their components ones.

We have known since at least the end of the nineteenth century and Henri Poincaré (1892) that the dynamic of such complex systems is unpredictable. The only way to know their state at a given step is to compute each step. The usual analytical method is of little help; the necessary mathematics are still to be invented. Even the small body of mathematics which directly deals with non-linearity depends upon linear approximations (Holland, 1999). Non-linearity thus challenges the traditional approach which tries to understand a system by analyzing its components: "The key feature of non-linear systems is that their primary behaviors of interest are properties of the *interactions between parts,* rather than being properties of the parts themselves, and these interactions-based properties necessarily disappear when the parts are studied independently" (Langton, 1989, p. 41, original italics).

How to deal with emergence? How to study processes which are "more than the sum of their parts"? How to analyze properties that cannot be forecasted? The solution proposed by computer scientists is termed *bottom-up modeling*.

Bottom-up modeling is a very new way of building artificial systems. Since core properties disappear when the components are studied independently, bottom-up modeling is based on the gathering of interacting components. Corpses clustering or larval sorting models are then based on the building of rather simple *agents* (see below) which interact both with one another and with the environment. Such constructions and the study of the dynamic resulting from non-linear interactions of the simple components constitute the "bottom-up method." Instead of modeling the global dynamic of the studied system ("top-down method" usually based on differential equations), one merely models the components to study the potentially emerging regularities.

This *synthetic method* is at the heart of the revival of artificiality. Commenting on the first workshop on artificial life, C. Langton stated: "I think that many of us went away…with a very similar vision, strongly based on themes such as *bottom-up* rather than *top-down* modeling, *local* rather than *global* control, *simple* rather than *complex* specifications, *emergent* rather than *pre-specified* behavior, *population* rather than *individual* simulation, and so forth" (Langton, 1989, p. xvi, original italics).

The 19th century ended with Poincaré's discovery of the limits of the analytical method faced with non-linear systems. The twentieth century ended with the unprecedented quick spread of a machine able to deal with these systems. Computers are in fact surprisingly adapted to the analysis of non-linear systems. Besides their ability to iteratively compute equations which do not have analytical solutions, computers—particularly since the development of object oriented programming—can easily deal with populations of interacting agents, so contributing to the study of Bedau's weak emergence. "…Computer-based models offer a halfway house between theory and experiment [and computer-based non-linear modeling] will certainly improve our understanding of emergence" (Holland, 1999, p. 232).

Bottom-up modeling is based on the interactions of (usually) simple virtual individuals. It massively uses *multi-agent systems* (MASs).

Multi-Agent Systems

MASs originate in *distributed artificial intelligence* (DAI) and in artificial life. The basic idea of DAI is that intelligence is not only a matter of phenotype (brain), but also depends on the interactions with other individuals. Intelligence has a "social dimension" (Drogoul, 2005). The emergence of DAI is directly linked

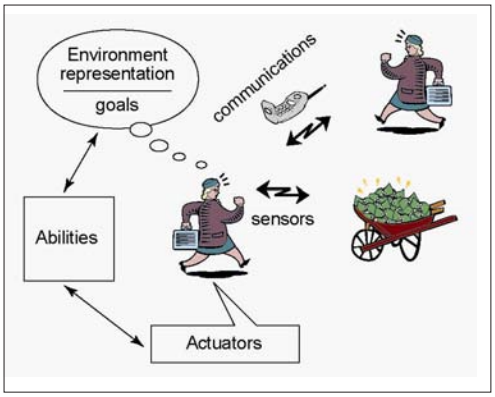
to the limits of the traditional symbolic AI (GOFAI), which tries to embed intelligence in a unique entity. The *cognitive school* of DAI associates a few complex agents to obtain some kind of group expertise (e.g., Demazeau & Müller, 1991). The *reactive* school of DAI is more original. Strongly rooted in artificial life, it uses the insect (and animal) societies metaphor to try to obtain *emergent intelligent behaviors* by associating simple ("sub-cognitive") agents (Steels, 1990; Deneubourg, Goss, Beckers, & Sandini, 1991).

We have seen that cemetery building was modeled with "virtual insects"—that is, some software processes that imitate insects' behaviors. These virtual insects are *agents.* Jacques Ferber, one of the founders of the field, considers that an agent is a physical or virtual entity (Ferber, 1999):

- capable of acting;
- capable of communicating with other agents;
- driven by a set of tendencies—autonomous agents act according to their own goals;
- having its own resources, but these resources depend on the environment—agents are then open systems since they find resources in the environment, and close systems since they manage the use of these resources;
- having a partial representation of their environment—an agent thus does not have to "fully understand" its environment; above all it does not have to perceive the global result of its actions;
- possessing skills;
- possibly able to reproduce itself; and
- tending to act according to its objectives.

The agent is thus a kind of 'living organism', whose behavior, which can be summarized as communicating, acting and perhaps, reproducing, is aimed at satisfying its needs and attaining its objectives, on the basis of all the other elements (perception, representation, action, communication and resource) which are available to it. (Ferber, 1999, p. 10)

Figure 2. An agent in its environment



Ferber's definition is restrictive and one can limit the characterization of agents to the following core properties (Wooldridge & Jennings, 1995):

- **Autonomy:** Agents operate according to their own control.
- **Social Ability:** Agents can interact with one another through some kind of language.
- **Reactivity:** Agents can perceive their environment and react according to its change.
- **Pro-Activeness:** Agents act according to their own goals.

Figure 2 summarizes the structure of an agent.

Bottom-up modeling uses interacting agents by building *multi-agent systems.* An MAS contains the following elements (Ferber, 1999): An environment *E*; a set of objects *O* having a

specific position in the environment; a set of agents A with $A \subseteq O$; a set of relations R linking the objects to each other; a set of operations Op allowing the agent to "perceive, produce, consume, transform, and manipulate" objects; and operators able to apply the operations and to process the reaction of the environment. MASs and *agent-based modeling* (ABM) are the base of social simulation (e.g., the *Iterated Prisoners Dilemma*—IPD of Axelrod, 1984, 1997) and artificial societies (Conte, Gilbert, & Sichman, 1998).

Artificial Societies

How to connect virtual agents with human societies? Humans are quite different from ants and despite real progress—thanks to the quick growth of computer power—the intelligence of the most sophisticated agent ever programmed cannot be compared to human intelligence. The 2005 Nobel Prize in Economics was attributed to Thomas C. Schelling (along with Robert J. Aumann) who proposed in 1971 (Schelling, 1971, 1978) a far-ahead-of-one's-time experiment that will help us understand the link between agents and human societies.

The Seminal Model of Thomas Schelling

Schelling wanted to understand the pre-eminence of geographical segregation between black and white in American cities despite the fact that when they are questioned, citizens refute any desire of segregation. He designed very simple agents of two distinct colors ("black and white"), having the following abilities:

- Each agent can compute the fraction of neighbors having the same color.
- If this fraction is below the agent preference, then the agent moves to an unoccupied place which satisfies its preference.

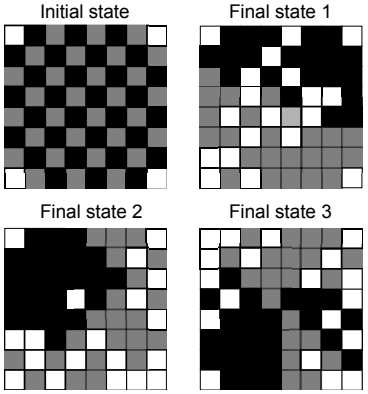
Schelling used cellular automata to implement its experiment. Very briefly, cellular automata are a lattice of sites whose states—belonging to a finite set—evolve in discrete time step according to rules depending on the states of the neighbors' sites. In a two-dimension implementation, Schelling used a "Moore" neighborhood—that is, neighbors are the eight closest squares. The rules were:

- If an agent has two neighbors, it will not move if at least one is of the same color.
- If an agent has three to five neighbors, it will not move if at least two are of the same color.
- If an agent has six to eight neighbors, it will not move if at least three are of the same color.

These rules are compatible with a fully integrated structure. The initial state of Schelling (see Figure 3) is thus an attractor since no agent needs to move. Schelling showed that a slight perturbation of this initial state is sufficient to give rise to a dynamic quite inevitably leading to segregation (see Figure 3).

Schelling's model clearly demonstrates that local interactions (*micro-motives*) lead to global structures (*macro-behavior*) (Schelling, 1978). More importantly, he showed that the

Figure 3. Schelling's model



macro-behavior can be different from the underlying micro-motives, since segregation occurs even when preference rules are compatible with integrated structure. Nowak and Latané (1993) used an extended model to study *dynamic social impact*—that is, the change of attitudes or beliefs resulting from the action of other individuals. They notably showed that the system achieved stable diversity. The minority survived, thanks to a clustering process of attitudes, not because individuals moved, but due to the attitude change process (Latané, 1996). The observed macro-behaviors are very robust. Schelling's and Latané's models were tested under a wide range of parameters and quite always evolve towards the same type of attractors. Pancs and Vriend (2003) recently enlarged the study of segregation process, showing that it tends to occur even if people are anxious that segregation should not occur.

Both these examples show that some complex social dynamics can be modeled from simple basis: "…there is a spirit in the air which suggests that we should look for simple explanations of apparent complexity" (Gilbert, 1995b). Wolfram (2002) recently brought a strong justification to this quest for simplicity. Its *Principle of Computational Equivalence* states:

...almost all processes that are not obviously simple can be viewed as computations of equivalent sophistication…So this implies that from a computational point of view even systems with quite different underlying structures…can always exhibit the same level of computational sophistication.…And what it suggests is that a fundamental unity exists across a vast range of processes in nature and elsewhere: despite all their detailed differences every process can be viewed as corresponding to a computation that is ultimately equivalent in its sophistication. (Wolfram, 2002, pp. 717-719)

Without going as far as Wolfram, it is now clear that at least some social phenomena can be modeled with interacting sub-cognitive agents.

Social sciences are traditionally based on two different models (Lansing, 2002):

- The Newtonian model uses systems of differential equations to study equilibrium—the best example being equilibrium theory in economics, which is also a brilliant example of the consequences of over-simplification motivated by the will to obtain tractable equations, the results having few to do with reality.
- Considering the difficulty to write the equations of the system, the statistical model tries to discover regularities—the best example being the study of "social forces" by Durkheim in 1897 (Durkheim, 2004).

Schelling's or Latané's models are then quite a new way of doing social sciences based on virtual experiments inside artificial societies.

Artificial Societies as a New Way of Doing Social Sciences

The field of *artificial societies* is based on the strong assumption that human societies are complex systems (Goldspink, 2000). Analysis is unable to point to the source of macro-properties since there is no localized source, but a distributed process which obliges to consider the system as a whole (Goldspink, 2002). Furthermore, they are *complex adaptive systems* (CASs), which are systems where agents can learn and modify their rules according to their

previous success (that is of course also the case of animal or insect societies, but the specificity of human—cognitive—societies is that they can also learn from their failures). Schelling's segregation process or Nowak and Latané's clustering process of people sharing the same opinion are emergences or "regularities at the global level" (Gilbert, 1995a). "As the number of elements and interactions of a system is increased, we can observe an *emergent complexity*. But *somehow*, regularities arise and we can observe *emergent simplicity*" (Gershenson, 2002, original italics).

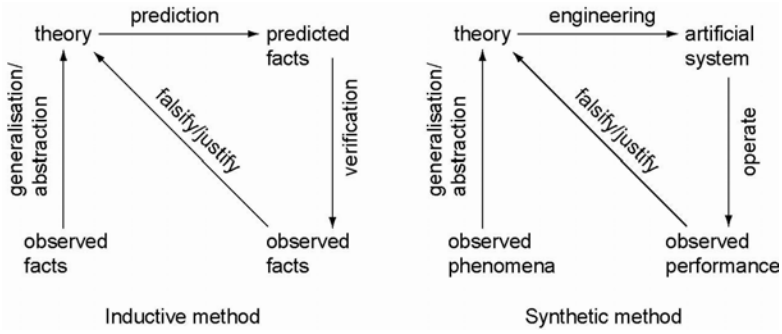
Artificial societies then try to obtain emergent regularities: "…the defining feature of an artificial society model is precisely that *fundamental social structures and group behaviors emerge from the interaction of individual agents operating on artificial environments*…" (Epstein & Axtell, 1996, p. 6, original italics). Considering European contributions to social modeling, Gilbert wrote: "One of the major objectives of the approach being reviewed here is to generate through simulation, emergent phenomena and thus to understand and explain the observable macro-level characteristics of societies" (Gilbert, 2000).

This is quite a new way of doing science, so new that simulation is said to be "a third way of doing sciences" (Axelrod, 2006) different from deduction and from induction. In the fields of artificial intelligence and artificial life, Luc Steels termed it the *synthetic method* (see Figure 4) (Steels & Brook, 1994).

Induction starts from observed facts and uses inferences to build a theory potentially able to globally explain the observed facts. The theory is then validated through the test of predicted facts. The synthetic method starts like induction from the observed facts and the inferred theory (but it can also start like deduction from a set of assumptions). On this basis, the synthetic method engineers an artificial system, the objective being that, while operating, this system will behave like the real one, thus confirming the tested theory.

In their seminal work, Epstein and Axtell (1996) considered that artificial society models may change the way we think about *explanation* in the social sciences. "Clearly, agent-based social science does not seem to be either deductive or inductive in the usual senses. But then what is it? We think *generative* is an appropriate term. The aim is to provide initial micro-specifications that are *sufficient to generate* the macrostructures of interest" (Epstein & Axtell, 1996, p. 177). This generative interpretation is directly linked to the disjunction between determinism and predictability which is a huge epistemological consequence of complexity sciences. Even if we perfectly understand the concerned forces, we are unable to

Figure 4. Inductive vs. synthetic method (Adapted from Steels & Brook, 1994)



predict the evolution of the system (Croquette, 1997).

A High Potential to Stimulate Novelty

Agent-based modeling is potentially a highly powerful tool for social scientists. Axelrod and Tesfatsion (forthcoming) recently synthesized its goals with four forms:

- **Empirical Understanding:** Why have regularities emerged?
- **Normative Understanding:** How can models help to define the good norms/ design? How to know if a given decision is positive for the society?
- **Heuristic:** How to attain greater insight about fundamental mechanisms in social systems?
- **Methodological Advancement:** How to give researchers the method and tools to rigorously study social systems?

Practically, these four forms rely on three pillars: formalization, experiments, and the ability to study the *macro to micro problem*.

Formalization

Apart from the verbal and mathematical symbol systems, computer simulation can be considered as the "third symbol system" (Ostrom, 1988). Any theory originating in the first two models can be expressed in the third one. Simulation can then be considered as formal models of theories (Sawyer, 2004). That is an important point since computer symbols are more adapted to social sciences than mathematical ones (Gilbert & Troitzsch, 2005, pp. 5-6):

- Programming languages are more expressive and less abstract than mathematical techniques.
- Programs deal more easily with parallel processing.
- Programs are modular. Major changes can easily be made; that is not the case of mathematical systems.

Computer modeling thus helps social scientists to formalize their theories. The difficulty—not to say the impossibility—to mathematically formalize many social sciences theories is considered to be a great weakness by "hard" scientists. This inability is closely linked to the inability of mathematics to deal with distributed emergent processes. Computer modeling can thus contribute to give social sciences some of the scientific tools they need to rigorously express their theoretical models.

Experiments

Simulation can be considered as a new experimental methodology. Gilbert and Conte (1995) defined it as "exploratory simulation." Such explorations can contribute to social sciences notably in the following ways:

- Modeling allows a *culture-dish methodology*. The modeler designs the agents and the initial state of its society, and studies its temporal evolution (Tesfatsion, 2002). Any sort of experiments can be carried out since the modeler has complete control of the model. It is then possible to study the consequences of any given modification. This will notably contribute to the analysis of the minimal set of parameters and system characteristics necessary to give rise to a given behavior, as well as to the analysis of the attractors of dynamic social systems (Goldspink, 2002).

  The ability to carry out experiments is something very new for social scientists that usually cannot test their theory in the

field. Like formalization, this contributes to bring closer social and "hard" science methods.

- Modeling is potentially able to contribute to original discoveries. The same way the classification of cellular automata permitted the proposal of an original analysis of complex systems (Wolfram, 1984; Langton, 1990), simulations can play a role in the discovery of general, yet unattainable laws. Implicit unknown effects can be detected (Gilbert & Conte, 1995). This ability to stimulate discovery does not only stand on the possibility to carry out otherwise impossible experiments, but also on the capacity of emergent modeling to give rise to original cognitive processes. In the field of artificial life, Cariani (1992)—emphasizing non-stochastic models like the Game of Life—pointed out the fact that emergence relies on a cognitive process; a process is emergent only according to its observer: "*The interesting emergent events that involve artificial life simulations reside not in the simulations themselves, but in the way that they change the way we think and interact with the world.* Rather than emergent devices on their own right, these computer simulations are catalyst for emergent processes in our minds; they help us create new ways of seeing the world." (Cariani, 1992, p. 790, original italics)
- Modeling can go beyond some of the limits of the statistical tools usually used by social scientists; for example, qualitative changes can be analyzed through simulation (see Pyka, 2006). Simulation also helps the study of processes. Usual statistical analyses study the correlations between variables at a given time. Simulations embed the processes which lead to these correlations (Gilbert & Troitzsch, 2005). Since social systems are fundamentally dynamic, simulation allows formalizing processes beyond the scope of statistical analysis. Furthermore, statistic is based on linearity assumptions which oblige to oversimplify the observed facts. Simulation does not suffer from this limit.
- Modeling is not concerned with the technical limits of mathematical formalization. For example, mathematical formalization obliges to consider agents as equivalent, whereas simulation is able to manage heterogeneous population. In the same vein, simulation allows the relaxation of assumptions necessary to obtain tractable equations (e.g., the rationality of economic agents). In economics, the highly promising field of Agent-Based Computational Economics (ACE) (Tesfatsion, 2002) clearly illustrates the potential of simulations.
- More generally, the same way artificial life allows the study of "life as it could be" (Langton, 1989), artificial societies allow the study of "societies as they could be" (Gilbert, 2000), thus giving social sciences an unprecedented tool to understand fundamental invariants (Rennard, 2004).

Study of the Macro to Micro Problem

The *macro to micro problem*—how to describe the relationship between macro-phenomena characterizing the dynamic of a system as a whole and micro-phenomena characterizing the dynamic of the components of the system—is a central issue of social sciences, but also of DAI (Schillo, Fischer, & Klein, 2000).

Simulation is a ground-breaking tool to study the core problem of the micro/macro relations. The relations between different levels (individual, organization, societal) and the potential associated lock-in can be studied. Artificial life, with its widely studied concept of *dynamical*

hierarchy, which "refers to a system that consists of multiple levels of organization having dynamics within and between the entities described at each of the different levels" (Lenaerts, Chu, & Watson, 2005, p. 403), should contribute to this study. Simulation can be used to study both the *micro to macro* and the *macro to micro problems* (Sawyer, 2003). Schelling's or Latané's models thus show how regularities can arise from micro-interactions. But such models also show that these regularities then constraint the system and impact the behaviors of individual agents. More directly, it is possible to conceive simulations that specifically study the impact of macro-phenomena. For example, Axtell (2000), while studying retirement behaviors, showed that modifying the sole network connections between agents can lead to great changes of the overall society behavior.

The study of the micro/macro problem through simulation remains nevertheless very difficult while studying societies. In fact, humans are not limited to basic behavior; they notably have the ability to grasp macro-level phenomena and they can adjust their behavior according to this. That is what Gilbert (2000) terms *second-order emergence,* characterizing systems where agents can detect and react to emergent properties. Models should then embed both the emergence of macro-properties and the ability to deal with the effects of these macro-properties on self-aware individuals. This remains a challenge (Gilbert, 1995b).

Limits

Artificial societies is a very recent field in which huge problems still are to be solved that challenges this research. A first set of problems relies on the cognitive dimension of human societies. Guided by the success of artificial life, many artificial societies are based on reactive DAI, one of the most famous examples being the *Sugarscape* of Epstein and Axtell (1996). The complexity of human cognition has a deep impact on the structuring of societies.

- Self-awareness and the related second-order emergence should be modeled.
- Interpretativism in sociology leads to the idea that meanings are parts of the actions: "…meanings and concepts describing both the physical and the social world are said to be socially constructed by members of society" (Gilbert, 2000). Simulations should then embed the corresponding social constructions.

As a consequence, artificial societies must find a way to associate cognitive and reactive DAI. This remains both a theoretical (how to build cognitive agents) and a practical (how to have sufficient computing power) problem.

A second set of problems is linked to the tools and methods used for modeling and simulation. First of all, simulation uses tools that may make implicit assumptions having nothing to do with the tested theory. For example, the use of cellular automata assumes that the world is a regular grid, which may have massive consequences on the global dynamic of the simulation (Troitzsch, 1997). Then simulation tends to develop its own finality, hence the importance to ground it in social theories in order to avoid the trend to develop simulations for themselves and to mistake them for reality. The balance is difficult to find: "If our 'toy models' serve only to reify and naturalize the conventional social science wisdom, then they are indeed a Medusan mirror, freezing the victim by the monster's glance" (Lansing, 2002, p. 289).

The gap between social sciences and computer sciences also challenges the field. Some social sciences theories are mainly descriptive and discursive, and such approaches may be very difficult to formalize through simulation. Moreover, despite common issues, the discussion between computer scientists and social

scientists remains very difficult. For computer scientists, non-formalized discursive social theories often seem blurred and they have difficulty understanding them. Social scientists are often reluctant facing computer programming, and they usually consider that computer scientists do not understand the complexity of human societies.

Finally, the core problem (which is not limited to artificial societies) of "how to obtain from local design and programming, and from local actions, interests, and views, *some desirable and relatively predictable/stable emergent results*" (Castelfranchi, 2000, original italics) still remains to be solved.

CONCLUSION

The field of artificial societies, despite old roots, is now only ten years old. Along with artificial life, it participates in an emerging way of doing science. This way still has to reach maturity, but will undoubtedly contribute to complement more traditional methods. The debate now is not to choose between usual methods and methods originating in artificiality, but to convince "traditional" scientists that artificiality is not limited to some kind of possibly funny computer game and to find ways of building stronger bridges between these practices of science. The growing easiness of computer programming and the quick spread of computer culture among young scientists is potentially a promise of quick evolution of artificiality in social sciences; no doubt this will contribute to renew the field.

REFERENCES

Axelrod, R. (1984). *The evolution of cooperation.* New York: Basic Books.

Axelrod, R. (1997). *The complexity of cooperation: Agent-based models of competition and collaboration.* London: University College London Press.

Axelrod, R. (2006). Advancing the art of simulation in the social sciences. In J.-P. Rennard (Ed.), *Handbook of research on nature-inspired computing for economics and management.* Hershey, PA: Idea Group Reference.

Axelrod, R., & Tesfatsion, L. (forthcoming). A guide for newcomers to agent-based modeling in the social sciences. In L. Tesfatsion & K. Judd (Eds.), *Handbook of computational economics, vol. 2: Agent-based computational economics.* Amsterdam: North-Holland.

Axtell, R.L. (2000). *Effect of interaction topology and activation regime in several multi-agent systems.* Technical Report No. 00-07-039, Santa Fe Institute, USA.

Baas, N.A. (1994). Emergence, hierarchies, and hyperstructures. In C. Langton (Ed.), *Artificial life 3* (pp. 515-537). Reading, MA: Addison-Wesley.

Bedau, M.A. (1997). Weak emergence. In T.J. (Ed.), *Philosophical perspectives: Mind, causation and world* (pp. 375-399). Malden, MA: Blackwell.

Bonabeau, E., Dorigo, M., & Théraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems.* Oxford: Oxford University Press.

Cariani, P. (1992). Emergence and artificial life. In C. Langton (Ed.), *Artificial life 2* (pp. 775-797). Redwood City, CA: Addison-Wesley.

Castelfranchi, C. (2000). Engineering social order. *Proceedings of the 1st International Workshop on Engineering Societies in the Agent World* (vol. 1972, pp. 1-18). London: Springer.

Chalmers, D.J. (2002). *Varieties of emergence*. Retrieved December 15, 2005, from http://consc.net/papers/granada.html

Conte, R., Gilbert, N., & Sichman, J. (1998). MAS and social simulation: A suitable commitment. In J. Sichman, R. Conte, & N. Gilbert (Eds.), *Proceedings of the 1st International Workshop on Multi-Agent Based Simulation* (vol. 1534, pp. 1-9). Berlin: Springer-Verlag.

Croquette, V. (1997). Déterminisme et chaos. In Pour la Science (Ed.), *L'ordre du chaos* (pp. 64-87). Paris: Belin.

Demazeau, Y., & Müller, J.-P. (1991). *Decentralized AI 2.* Amsterdam: Elsevier North Holland.

Deneubourg, J.-L., Goss, S., Beckers, R., & Sandini, G. (1991). Collectively self-solving problems. In A. Babloyantz (Ed.), *Self-organization, emergent properties and learning.* New York: Plenum.

Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., & Chretien, L. (1991). The dynamics of collective sorting: Robots-like ant and ant-like robots. In J. Meyer & S. Wilson (Eds.), *Proceedings of the 1st Conference on Simulation of Adaptive Behavior: From Animals to Animats* (pp. 356-365). Cambridge, MA: MIT Press.

Drogoul, A. (2005). Les sytèmes multi-agents. In J. Lautrey & J.-F. Richard (Eds.), *L'Intelligence.* Paris: Hermès.

Durkheim, E. (2004). *Le suicide*. Paris: PUF.

Emmeche, C., Koppe, S., & Stjernfelt, F. (1997). Explaining emergence. *Journal of General Philosophy of Science, 28,* 83-119.

Epstein, J.M., & Axtell, R.L. (1996). *Growing artificial societies. Social sciences from the bottom up.* Cambridge, MA: MIT Press.

Ferber, J. (1999). *Multi-agent systems.* London: Addison-Wesley.

Gershenson, C. (2002). Philosophical ideas on the simulation of social behavior. *Journal of Artificial Societies and Social Simulation*, *5*(3). Retrieved from http://jasss.soc.surrey.ac.uk/5/3/8.html

Gilbert, N. (1995a). Emergence in social simulation. In N. Gilbert & R. Conte (Eds.), *Artificial societies. The computer simulation of social life* (pp. 144-156). London: UCL Press.

Gilbert, N. (1995b, October 27-29). Simulation: An emergent perspective. *Proceedings of the Conference on New Technologies in the Social Sciences,* Bournemouth, UK.

Gilbert, N. (2000). Modeling sociality: The view from Europe. In T. Kohler & G. Gumerman (Eds.), *Dynamics in human and primates societies: Agent-based modeling of social and spatial process* (pp. 355-372). Oxford: Oxford University Press.

Gilbert, N., & Conte, R. (1995). *Artificial societies. The computer simulation of social life.* London: UCL Press.

Gilbert, N., & Troitzsch, K. (2005). *Simulation for the social scientist.* Maidenhead: Open University Press.

Goldspink, C. (2000). Modeling social systems as complex: Towards a social simulation meta-model. *Journal of Artificial Societies and Social Simulation, 3*(2). Retrieved from http://www.soc.surrey.ac.uk/JASSS/3/2/1.html

Goldspink, C. (2002). Methodological implications of complex systems approaches to sociality: Simulation as a foundation of knowledge. *Journal of Artificial Societies and Social Simulation, 5*(1). Retrieved from http://www.soc.surrey.ac.uk/JASSS/5/1/3.html

Halpin, B. (1999). Simulation in society. *American Behavioral Scientist, 42,* 1488-1508.

Holland, J. (1999). *Emergence*. Cambridge, MA: Perseus Books.

Kubik, A. (2003). Toward a formalization of emergence. *Artificial Life, 9,* 41-65.

Langton, C.G. (1989). Artificial life. In C.G. Langton (Ed.), *Artificial life 1*. Redwood City, CA: Addison-Wesley.

Langton, C.G. (1990). Computation at the edge of chaos: Phase transition and emergent computation. *Physica D, 42,* 12-37.

Lansing, J. (2002). "Artificial societies" and the social sciences. *Artificial Life, 8,* 279-292.

Latané, B. (1996). Dynamic social impact: Robust predictions from simple theory. In R. Hegselmann, U. Mueller, & K. Troitzsch (Eds.), *Modeling and simulating in the social sciences from a philosophy science point of view* (pp. 287-310). Kluwer.

Lenaerts, T., Chu, D., & Watson, R. (2005). Dynamical hierarchies. *Artificial Life, 11,* 403-405.

Morin, E. (1977). *La méthode 2.* Paris.

Nowak, A., & Latané, B. (1993). Simulating the emergence of social order from individual behavior. In N. Gilbert & J. Doran (Eds.), *Simulating societies: The computer simulation of social phenomena*. London: UCL Press.

Ostrom, T. (1988). Computer simulation: The third symbol system. *Journal of Experimental Social Psychology, 24,* 381-392.

Pancs, R., & Vriend, N. (2003). *Schelling's spatial proximity model of segregation revisited.* Department of Economics, WP 487, University of London.

Poincaré, H. (1892). *Les méthodes nouvelles de la mécanique céleste.* Paris: Blanchard.

Pyka, A. (2006). Modeling qualitative development—Agent-based approaches in economics. In J.-P. Rennard (Ed.), *Handbook of research on nature-inspired computing for economics and management.* Hershey, PA: Idea Group Reference.

Rennard, J.-P. (2004). Perspectives for strong artificial life. In L. de Castro & F. von Zuben (Eds.), *Recent developments in biologically inspired computing.* Hershey, PA: Idea Group Inc.

Ronald, E., Sipper, M., & Capcarrère, M. (1999). Design, observation, surprise! A test of emergence. *Artificial Life, 5*(3), 225-239.

Sawyer, R. (2003). Artificial societies: Multiagent systems and micro-macro link in sociological theory. *Sociological Methods and Research, 31*(3), 325-363.

Sawyer, R. (2004). Social explanation and computational simulation. *Philosophical Explorations, 7,* 219-231.

Schelling, T.C. (1971). Dynamic model of segregation. *Journal of Mathematical Sociology, 1*(2), 143-186.

Schelling, T.C. (1978). *Micro-motives and macro-behavior.* New York: Norton.

Schillo, M., Fischer, K., & Klein, C. (2000). The micro-macro link in DAI and sociology. *Lecture Notes in Artificial Intelligence*, *1979*.

Simon, H. (1996). *The sciences of the artificial* (3rd ed.). Cambridge, MA: MIT Press.

Steels, L. (1990). Cooperation between distributed agents through self-organization. In Y. Demazeau & J.-P. Müller (Eds.), *Decentralized AI* (pp. 175-196). Amsterdam: Elsevier.

Steels, L. (1997). Artificial life roots of artificial intelligence. In C. Langton (Ed.), *Artificial life. An overview* (pp. 75-110). Cambridge, MA: MIT Press.

Steels, L., & Brook, R. (1994). *The artificial life route to artificial intelligence*. New Haven, CT: Lawrence Erlbaum.

Tesfatsion, L. (2002). Agent-based computational economics: Growing economies from the bottom up. *Artificial Life, 8,* 55-82.

Troitzsch, K.G. (1997). Social simulation—Origins, prospects, purposes. In R. Conte, R. Hegselmann, & P. Terna (Eds.), *Simulating social phenomena* (vol. 456, pp. 41-54). Berlin: Springer-Verlag.

Waldrop, M. (1992). *Complexity: The emerging science at the edge of chaos*. New York: Simon and Schuster.

Wolfram, S. (1984). Universality and complexity in cellular automata. *Physica D, 10,* 1-35.

Wolfram, S. (2002). *A new kind of science.* Champain: Wolfram Media.

Wooldridge, M., & Jennings, N. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review, 10*(2), 115-152.

KEY TERMS

Bottom-Up Modeling: Applied to nonlinear systems, bottom-up modeling is based on the gathering of components in order to analyze the corresponding emerging properties.

Cellular Automata: Lattice of sites whose states—belonging to a finite set—evolve in discrete time step according to rules depending on the states of the neighbors' sites.

Complex Adaptive Systems: Complex systems where agents can learn and modify their rules according to their previous success.

Emergence: Characterizes the properties of a system which are new compared to the properties of the components isolated. Bottom-up modeling mainly deals with Bedau's *weak emergence* which characterizes emerging properties that can only be derived by simulation.

GOFAI: Good Old-Fashioned Artificial Intelligence characterizes the traditional symbol-based artificial intelligence.

Macro to Micro Problem: How to describe the relationship between macro-phenomena characterizing the dynamic of a system as a whole and micro-phenomena characterizing the dynamic of the components of the system.

Synthetic Method: The synthetic method starts like induction from the observed facts and the inferred theory (but it can also start like deduction from a set of assumptions). On this basis, the synthetic method engineers an artificial system, the objective being that, while operating, this system will behave like the real one, thus confirming the tested theory.

Chapter II
Multi-Cellular Techniques

Carl Anderson
Qbit, LLC USA

ABSTRACT

Social insects—ants, bees, wasps, and termites—and the distributed problem-solving, multi-agent paradigm that they represent, have been enormously influential in nature-inspired computing. Insect societies have been a source of inspiration and amazement for centuries, but only in the last 25 years or so have we made significant inroads to both understanding just how various collective phenomena arise and are governed, and how we can use the lessons and insights garnered from sociobiological research for more practical purposes. In this chapter, we provide a very brief history of the field, detailing some of the key phenomena, mechanisms, and lessons learned, and a quick tour of some of the different types of applications to which this knowledge has been put to use, including but certainly not limited to distributed problem solving, task allocation, search, and collective robotics.

ARTIFICIAL LIFE

Insect societies owe their illustriousness, in part, to their ubiquity (they are found on every continent except Antarctica) and that almost all of us, at one time or another, has had some food item discovered by a single foraging ant, and only a few moments later we witness the arrival of a whole group—then a trail—of nestmates, ready to carry off the spoils. Move the food, and the trail quickly adapts to the new location. More concretely, provide a colony of certain ant species with a choice of two food sources, say a weak sugar solution and a strong sugar solution, and the colony will select a trail to the better source, moreover, without a single ant ever visiting both food sources (Camazine et al., 2001, and references therein). Watch a trail over time and it can become straighter, and thus more efficient, again without any individual having a global view of the situation (Bruckstein, 1993; see Shao & Hristu-Varsakelis, 2005, for an application). Peer inside a colony and you will find many different tasks being performed concurrently (cleaning, feeding larvae, processing food, etc.), each task with the appropriate number of individuals to meet that task's demands. Remove some of the individuals tack-

ling one of the tasks, and the allocation of workers across the colony will shift to redress the balance (Wilson, 1984). Just how can a colony of these tiny creatures, with necessarily small brains, achieve such amazing, adaptive collective behavior?

People have long pondered this very question, perhaps summed up best by Maeterlink (1927): "What is it that governs here? What is it that issues orders, foresees the future, elaborates plans and preserves equilibrium, administers, and condemns to death?" Many have assumed that it is the queen herself that directs the colony's activities (and in some cases, that it is the relatively inactive ant soldiers, with their larger heads, who direct traffic on trails) (Step, 1924; Ewers, 1927). However, this would require both a sophisticated communication system and a remarkable cognitive ability on the part of the queen to collate all the necessary information, process it, devise some plan of action, and pass those orders back to the workers. The reality is that while there exists some degree of queen control, especially in very small insect societies, this mostly relates to reproductive rights, the queen maintaining her reign. Quotidian tasks such as foraging, cleaning, and nest construction are regulated in a very *distributed* manner relying on direct individual-to-individual interactions or indirect "stigmergic" interactions (Grassé, 1959) mediated through the environment (e.g., ants that lay trail pheromone that influences the foraging behavior of other ants) (e.g., Hölldobler & Wilson, 1990; Camazine et al., 2001).

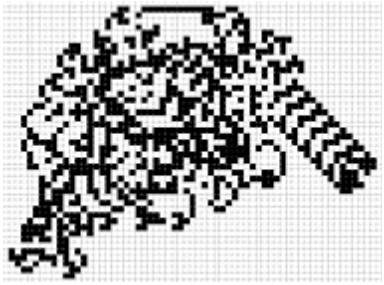
While careful methodical experimentation and detailed mathematical models have helped elucidate some of the proximate mechanisms at work, the popularization of insect societies as a powerful metaphor and new paradigm among the artificial intelligence community owes much to the field of artificial life. (Although we should not forget Hofstadter's 1980 highly influential and Pulitzer prize winning book, *Gödel, Escher and Bach*, in which, in one chapter, "Ant Fugue," he uses an ant colony as a metaphor for the mind.) A-life, a field of artificial biology (usually) using computer simulation to model "life as it is," to explain extant biological phenomena, or "life as it could be," to explore life's possibilities, began in the 1980s. Of particular relevance is one of the seminal models in the field and one of the earliest models of ants. Langton's virtual ants or "vants" (Langton, 1986) are absurdly simple: there is a grid of cells that may be black or white and one or more ants; an ant that lands on a black cell turns the cell white, turns right and moves forward one unit; an ant that lands on a white cell turns the cell black, turns left and moves forward one unit. Despite the apparent triviality of this system, what arises is surprising: ants may mill around in a seemingly chaotic fashion, but in certain situations may interact with each other, mediated through the color of the cells, to form "highways" (see Figure 1) and move the ants in a coordinate fashion across the grid.

It is computer experiments such as these that fired up the imaginations of many researchers and triggered a slew of ant-based simulations that formed the basis for this subfield of nature-inspired computing. This approach of abstracting such systems almost to the point of absurdity, and yet still retain incredibly complex and surprising behavior, seems to have been key in eradicating the mysticism that surrounds many complex systems. Here was a trivial, deterministic system in which all local rules and behavior are known, and yet the long-term collective behavior was in most cases unpredictable from a given set of initial conditions. (In fact, vants is a four-state, two-dimensional Turing machine; Weisstein, 2005.) Now one possessed a mini-world in which one could explore initial conditions and other parameters, and by use of careful experimentations stood a

Figure 1. Langton's (1986) virtual ants or "vants." From an initial configuration, individual vants follow the simple rules (at left), eventually resulting in an ordered cooperative configuration (the "highway" at far right).

Langton's (1986) virtual ants

1) If the ant is on a black square, it turns right 90° and moves forward one unit.
2) If the ant is on a white square, it turns left 90° and moves forward one unit.
3) When the ant leaves a square, it inverts the color.



chance at understanding and linking the simple micro-level behavior with the more complex macro-level collective behavior (Levy, 1992). If it is true that insect societies are not centrally governed, then it must be mechanisms, such as those observed in vants, that link the relatively simple individual-level behavior of ants to the more complex collective phenomena of colonies, such as trails, but moreover, the *adaptiveness* of such emergent behavior. Over the years, collaborations between the field biologists and experimentalists (who provided the possible proximate mechanisms at work), and mathematicians, "A-lifers," and other computer scientists (who could rigorously and objectively test whether those mechanisms were indeed necessary and sufficient to generate an observed collective behavior) made huge inroads in understanding some of key mechanisms and feedback at work.

PROPERTIES OF ANT-LIKE SYSTEMS

As more and more was learned, so the potential of adopting such distributed problem-solving paradigm in other areas and for anthropocentric uses became apparent. While we will review some of the specific mechanisms and applications later, let us first consider why this new paradigm was so appealing. What are the pros and cons of such distributed systems, especially those inspired by sociobiology?

The first advantage of such systems is their *distributed nature* per se. A set of individuals, or more generically "agents," can operate autonomously and do so in a loose social network, interacting only with other individuals they happen to encounter, or employing some other local influencing mechanism, such as pheromone deposition. In other words, information need not always flow to some particular individual (such as a queen) or specific subgroup of individuals. This could be hard to coordinate (how would all two million army ants in a colony "report" to the single queen?) and expensive, both to operate and also to evolve. Thus, that a group of agents can coordinate activities without a fully connected information network—in other words, that "average system connectedness" is low (Anderson & McShea, 2001; Moritz & Southwick, 1992)—is a huge boon, and certainly favorable to collective-roboticists (discussed later).

Second, such systems are highly *robust*. The distributed nature means that there are few if any key individuals. Thus, remove a group of foragers and others can switch tasks and take their place with little detriment to the colony. Contrast this with the removal of a key individual in a centralized paradigm: the system collapses.

Third, and related, such systems are highly *flexible* and *adaptive*. Insect societies do not forecast, they react and respond to current stimuli and conditions. The behavior of individuals is likely tuned through natural selection (at the level of the colony; see Sober & Wilson, 1998) so that the system behavior, such as the foraging network as a whole, is appropriately responsive given the degree of perturbations likely to be encountered in that environment. Consider an ant that finds a new food source and who recruits nestmates thereby establishing a trail between the nest and the food source. That food source is ephemeral, and at some point will no longer supply food; the colony must seek other food sources. Interestingly, foragers are not perfect at following a trail, and with a certain probability per cm of trail, a forager will "fall off" the trail and become lost, possibly never to refind the trail or colony. While this may at first sight sound like maladaptive behavior, in fact, some of these individuals may find new sources, and by the use of landmarks or sheer luck find their way back to the colony and so can establish a new trail to the additional source. Thus, the colony effectively employs a dual-timescale optimization procedure with respect to foraging: those ants who do not lose the trail exploit the current (i.e., immediate timescale supply) food sources, while other ants who do lose the trail become scouts seeking new food sources, thus ensuring future (i.e., longer timescale) reserves. And, returning to our earlier point, the "error rate"—the probability that an ant will lose the trail—is the behavior or parameter that is tuned though selection as a function of the environment—that is, food distribution (Deneubourg, Aron, Goss, Pasteels, & Duernick, 1986).

So, even a very simple, seemingly suboptimal, individual-level behavior such as inability to follow a pheromone trail can give rise to a more complex, multi-objective, multi-timescale collective behavior. In other words, insect societies are complex adaptive systems (Bonabeau, 1998). Another classic example is nest construction. Large social wasp colonies and termites can produce incredibly complex nest architectures, with some termite nests containing spiral staircases and ventilation systems. Individuals do not build these nests by following some comprehensive blueprint. Instead, the mound is a result of an *emergent* process: a termite lays down a pellet that is infused with a pheromone. This pheromone field influences the placement of subsequent pellets in the same place by other nearby termites. This increased pheromone concentration has a stronger influence on other termites, and so this very simple positive feedback mechanism triggers two things: the construction of a pillar, but also the collective, non-negotiated, non-predetermined decision of *where to place the pillar* (Camazine et al., 2001).

Such stigmergic indirect interactions among individuals have other desirable properties: they act as a sort of "collective memory" and an averaging process. Consider a Y-shaped junction on an ant trail. Some ants choose the right trail and lay down pheromone droplets that evaporate at some constant rate, while other individuals choose the left trail and similarly lay down droplets of evaporating pheromone. A new individual arrives at the junction and can sense the relative amounts of pheromone on the two trails. Those two concentrations are a form of summation of the multiple individuals that have chosen and traveled down each trail.

Further, the pheromone evaporation process, which over time eliminates the input from the earlier ants, acts as a sliding window averaging process that is a direct function of the recent rate at which ants have traveled a given trail, thereby providing a newly arrived ant with a more reliable indicator as to which is the best trail to follow.

With all these attractive properties, what are the downsides to such systems? First, in a completely flat, decentralized system, not one individual has a global perspective of the system. Consequently, the system, while it can be extremely adaptive, can also be highly *myopic* and develop pathological behavior. In ants, the classic example is that of circular milling (Schneirla, 1971). *Eciton* army ants are blind, entirely reliant on tactile stimuli and trail following, and in certain situations a circular trail can form in which hundreds of ants can follow each other for hours or days. Second, and related, these systems are not fully connected and so information may travel a convoluted path before it reaches certain individuals and becomes out of date, distorted, or lost entirely. Third, the collective properties are emergent, often composed of a complex and sensitive network of feedback: a slight change in one key parameter can cause a disproportionately large shift in collective behavior. While this is a key component of the adaptiveness of the system, for practical purposes it can be very hard to *design* a truly distributed, self-organized system with desired collective behaviors (Anderson, 2006).

In summary, while such decentralized, distributed problem-solving systems can be hard to design and may develop myopic pathological behavior, they do offer incredible flexibility and adaptiveness, and such systems' adaptiveness can arise from very simple individuals with simple interaction behavior (but complex agents would work too!)—and to engineers, "simple" means *easy* and *cheap* to design—and offer great system robustness. In the next section, we review some of the types of problems and solutions to which this multi-agent paradigm has been used.

EXAMPLES

Ant Colony Optimization

Undoubtedly, the most celebrated direct application of these social insect-inspired mechanisms is the whole new field called "ant colony optimization" (ACO; Moyson & Manderick, 1988; Bonabeau, Dorigo, & Theraulaz, 1999). These ant systems are a strong mimic of the pheromone-based foraging behavior of ants and their collective, distributed problem-solving abilities. This field uses systems of virtual ant-mimics to solve a variety of complex static optimization problems including the traveling salesman problem (TSP), job shop scheduling, graph partitioning, graph coloring, vehicle routing, sequential ordering and knapsack problems, and a variety of dynamic problems including dynamic load balancing in telecommunication systems (reviewed in Bonabeau et al., 1999).

The core idea in a static problem such as the TSP is that we can formulate an optimization problem spatially. Ants explore and travel the solution space in parallel and collectively build partial solutions—that is, a good route through a subset of the available nodes; in short a subgraph—mediated through the strength of the virtual pheromone trails. Good partial solutions will be reinforced as more ants choose a certain node-node path, while poor solutions, perhaps a path randomly chosen by an ant but which happens to be a poor choice, will fail to be reinforced by other ants, and the negative feedback mechanism of pheromone evaporation will degrade this partial solution over time.

Eventually, the network of trails will converge on a good, often (but not guaranteed) optimal, global solution. Thus, when an ant arrives at a city node, it chooses the next city to travel to probabilistically using the relative strength of the pheromone trails as a guide. Like ants assessing the quality of a trail by the quality of the food source at the end of the trail and who deposit pheromone on their return trip as a function of that quality, at the end of its tour the ACO ant will reinforce the pheromone trails along its tour route in proportion to the quality of the tour that that ant experienced. In this case, we only need update a lookup table that defines the relative pheromone concentration of the "edges" emanating from a certain node.

ACO has demonstrated its viability as proof-of-concept on a variety of static NP hard problems. However, it will never be as efficient as the best deterministic methods developed for such problem types (see Table 1). ACO does not guarantee that it will find the optimal solution, and proofs about the convergence properties are difficult. However, ACO systems do tend to find a near optimal solution quickly and they do possess the same system robustness properties as real insect societies. Remove some of the ants or one of the cities in the above example, and the remainder of the virtual ants will continue unaffected and the trail-based solution will merely adapt to the new problem. In a way, the pheromone trail network, which includes both strong and weak trails between nodes, contains a set of inherent backup solutions that can quickly be utilized in the face of problem and system perturbations. It is for these reasons that ACO has a far brighter future (and will garner greater respect from more traditionalist system engineers) in tackling dynamic optimization problems.

With a rapidly changing, complex problem, when the time to compute the globally optimal solution is longer than the timescale at which the problem changes, then a system that will find a near-optimal solution rapidly will outperform a system that produces the optimal solution slowly. Thus, ACO-like systems have proven very successful in dynamic problems such as load balancing in telecommunications network and the Internet (Bonabeau et al., 1999; see also Nakrani & Tovey, 2003, for a honeybee-inspired example).

Table 1. Some key properties of various optimization techniques

	Speed	
Solution Quality	**Fast**	**Slow**
Optimal	Linear programming	Branch and bound
Near-Optimal	Constructive heuristics	Local search: Genetic algorithms, simulated annealing, tabu search, ACO

Threshold Responses and Dominance Hierarchies

Very robust and adaptive task allocation is found in insect societies: alter the relative workload (i.e., ratio of workers to work) of one of the tasks, and the set of individuals will shift tasks to redress the shift, in short they will load balance. How is this achieved in a distributed manner? It is believed that this is achieved using individual task-specific threshold responses (reviewed in Bonabeau & Theraulaz, 1999). Suppose that there is only one task whose workload is positively correlated with some task-specific stimulus: the more work there is to be done, the larger is s. Now suppose that an individual has a threshold q for that task. Simplistically, when $s < \theta$, the worker is not sufficiently stimulated to tackle the task and the larger is s, the more likely the individual is to begin work. More concretely, the probability to begin work:

$$T(\theta, s) = s^2 / (s^2 + \theta^2), \qquad (1)$$

Table 2. Summary of some key social insect-inspired mechanisms and applications

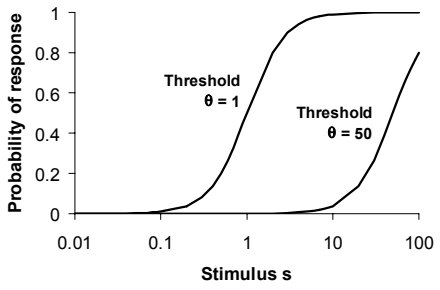
Biological Feature	Mechanism	Applications
Pheromone trails	Individuals influence route choice of others through the use of transient signals left in the environment	Distributed problem solving for route planning and other high-dimensional problems; collective robotics
Threshold responses	Individual i responds to a stimulus with intensity s with probability $s^2/(s^2+\theta_i^2)$ where θ_i = individual i's threshold for stimulus s	Manufacturing scheduling, especially market-based task allocation
Stigmergic construction & piling	When collectively constructing a nest, i) pheromone-infused construction materials (quantitative stigmergy) or ii) local configurations (qualitative stigmergy) shape nest architecture without explicit blueprints and enable effective collective decision making (see Anderson, 2002, for a discussion of qualitative vs. quantitative stigmergy)	Construction tasks in collective robotics (and potentially nanotechnology); data mining
Stigmergic coordination	When collectively transporting an item, individuals sense and respond to the object's overall motion rather than direct communication among the transporters	Collective transport in robot swarms
Group foraging	Multiple individuals perform a parallel search for resources, and recruits center their search on the most promising sites	Search and numerical optimization

an exponential function such that, depending on the magnitude of θ, exhibits a sharp transition in task commencement likelihood (see Figure 2). Now suppose that each individual i has a task-specific threshold $\theta_{i,j}$ for each of the different tasks (subscript j). With this, we can begin to understand how distributed task allocation can work. Remove some workers from task j and this reduces the *worker: work* ratio and so stimulus s_j will increase, thereby making it more likely that an idle worker will commence work—because $T(\theta_{i,j},s_j)$ is high—or that an individual engaged on a different task, say task k, will be more stimulated to switch to task j than remain with task k—because $T(\theta_{i,j},s_j)$ is much less than $T(\theta_{i,k},s_k)$. Place too many workers on a task j, $T(\theta_{i,j},s_j)$ becomes low and individuals are likely to switch to other more stimulating tasks. Thus, in a sense there is indirect interaction among the workers mediated through the work demand itself. A worker does not need to calculate the number of individuals tackling a task, only the relative, current demand for that task. To prevent an undesirable all-or-nothing response where everyone switches to or from a task simultaneously, a key component is inter-individual variability: that different individuals have different thresholds for the same task. In addition, there are strong theoretical grounds (and some empirical evidence, e.g., Weidenmüller, 2004) for threshold reinforcement and learning. That is, suppose that an individual responds to task j. If it then lowers its $\theta_{i,j}$, that is:

new threshold value = old threshold value – some positive amount ξ

then it is more likely to tackle task j in the future for the same level of s_j. This positive reinforce-

Figure 2. Probability that an individual responds (y-axis) vs. stimulus levels (x-axis) (equation 1) for two values of response threshold theta



ment mechanism can help individual i *specialize* on task j. Similarly, an individual that is *not* working on some task k raises is threshold, that is:

new threshold value = old threshold value + some positive amount φ

so that it is unlikely to tackle rare (low *s*) or long forgotten (high $\theta_{i,k}$) tasks, unless there is a strong need. ξ and φ are referred to as the learning and forgetting rates respectively.

How can we apply such a mechanism? There are very strong parallels between the problem that the social insects are solving and problems that humans must solve. Generically: given a set of workers or machines, and a variable inflow of different tasks, how do we optimally assign the work to the workers? This is the core task of systems engineering and operations research, and indeed this is where social insect-inspired systems have found a role. In effect, the threshold response task allocation mechanism above creates an internal market within the colony. The workers place a form of "bid" in terms of the willingness to tackle a certain task, and those individuals with the highest "bids" (= lowest response thresholds) tend to "win" the task. This is precisely the logic behind the first application in this area: assigning trucks to paint booths (Morley, 1996; Campos, Bonabeau, Theraulaz, & Deneubourg, 2001; Kittithreerapronchai & Anderson, 2003; Cicirello & Smith, 2004). In brief, trucks roll off an assembly line and must be painted a certain customer-desired color. However, there is only a certain limited number of paint booths, and if a paint booth must switch paints, then there is both a time and material cost as the booth flushes and refills its paint lines. An effective, distributed solution is for the paint booths to bid against each other: those already primed with the required color and who are currently empty submit the highest bid (lowest response threshold) while those who are currently busy painting a truck in a different color, or indeed are not currently operating submit the lowest bids. Moreover, like the social insects, booths can modify their threshold responses over time and become color specialists. It has been shown that such a system can assign the trucks to the booths in the face of unpredictable workflow (i.e., arrival time and color of new trucks waiting to be painted) in a very efficient and robust manner; interestingly, Morley's (1996) scheme was implemented in an actual *GM* manufacturing plant.

Since those early experiments, Cicirello and Smith (2001, 2002a, 2002b) have examined similar mechanisms in a variety of job-shop and other complex logistical problems (see also chapters 33, 34, and 47 in this volume). They also added another social insect-inspired dimension to their schemes, that of dominance hierarchies. Many social wasp species form a dominance hierarchy in which bouts between individuals establish a form of "pecking order." In cases where two or more individuals respond positively to some job, the authors implemented a similar threshold response mechanism to

emulate the way in which disputes between individuals are resolved. Again, this simple distributed task allocation feature adds positively to the overall effectiveness of the system.

Search and High-Dimensional Numerical Optimization

Genetic algorithms (see Chapters IV and V in this volume) perform a parallel search in (usually) high-dimensional space. Similarly, a group of ants foraging for resources are also performing a parallel search and solving a high-dimensional problem in terms of the collective decisions of which resource(s) should be recruited to given colony size, spatial distribution of foragers and resources, resource quality, distance of resource to the nest, and so on. The key mechanism here is recruitment in which good resources (or more generically, good "solutions") attract more individuals to an area and thus a more intensive search of a promising area often at the expense of forager density at some other less productive site. (It should be stressed that although similar to route choice using pheromone trails, this problem is far less constrained because of the spatial nature of the search, and that recruitment is individual-individual interaction rather than chemical based.) In some cases, ants will move their whole colony to a more productive area and that site will become the new hub for forager operations.

Monmarché, Venturini, and Slimane (2000) emulated such a mechanism in a suite of two to five variable numerical optimization tasks (e.g., minimize $0.5+((\sin^2(x_1^2+x_2^2)^{0.5})-0.5)/(1+0.0001(x_1^2+x_2^2)$ where $x_i \in [-100,100]$). They specifically modeled their search algorithm on a particular ant *Pachycondyla apicalis*, a primitive ponerine ant that recruits on an individual basis through the use of "tandem running." In brief, an ant-agent will choose a random foraging site and explore around that area, transporting any captured prey to the nest and returning to that same site if that site was successful or choosing a different site after a certain number of unsuccessful foraging trips. Each time after all n ants have explored their neighborhood, a random pair of ants is chosen and the ant with the more productive foraging site will recruit the other ant to its site. Finally, and importantly, the nest is periodically moved to the most productive site. This example, with a very strong biological grounding, and which performs well (outcompeting hill climbing and genetic algorithms for many of the test functions) is particularly interesting given its very strong similarities to genetic algorithms (discussed in detail by the authors).

Collective Robotics

Collective robotics is another field that has drawn very heavily on insect societies (and for which there is insufficient space here to do it justice) and represents a substantial shift from the long-held philosophy of building highly "intelligent" individual robots to building a robust swarm of simpler, cheaper, disposable robots that solve problems collectively. Essentially, collective robotics goes a step further than ACO an other similar swarm intelligence methods in that it represents physical ant-mimetic instantiations. Thus, there are robot systems that directly mimic ant recruitment systems through trail laying or "crumb dropping" (Drogoul & Ferber, 1993) for collective search and foraging (Krieger, Billeter, & Keller, 2000), stigmergic robot systems that sort and pile scattered objects in a similar way that termites construct nests and ants form cemetery piles of dead bodies (Holland & Melhuish, 1999; Bonabeau et al., 1999), and even hope that biomemetic robots can interact with real ani-

mals in groups themselves and shape collective behavior (Caprari, Estier, & Siegwart, 2000; Gautrais, Jost, Jeanson, & Theraulaz, 2004; see also Cao, Fukunaga, & Khang's 1997 comprehensive review).

DISCUSSION

This has been a very brief review of distributed, swarm-like systems that have been directly inspired by the remarkable collective abilities of the insect societies. These complex adaptive systems—both the ants themselves and their virtual or silicon mimics—are composed of relatively simple individuals that tend to respond to local cues with simple heuristics. However, together they form systems that are decentralized, extremely robust, and highly adaptive without forecasting and make full use of emergent properties. This is a relatively young subfield with a lot of potential; it will be fascinating to follow future directions.

REFERENCES

Anderson, C. (2002). Self-organization in relation to several similar concepts: Are the boundaries to self-organization indistinct? *Biological Bulletin, 202,* 247-255.

Anderson, C. (2006). Creation of desirable complexity: Strategies for designing self-organized systems. In D. Braha, A. Minai, & Y. Bar-Yam (Eds.), *Complex engineering systems* (volume in the *New England Complex Systems Institute Series on Complexity*). New York: Springer-Verlag.

Anderson, C., & McShea, D. W. (2001). Individual versus social complexity, with particular reference to ant colonies. *Biological Reviews, 76*(2), 161-209.

Bonabeau, E. (1998). Social insect colonies as complex adaptive systems. *Ecosystems, 1,* 437-443.

Bonabeau, E., & Theraulaz, G. (1999). Role and variability of response thresholds in the regulation of division of labor in insect societies. In C. Detrain, J.L. Deneubourg, & J.M. Pasteels (Eds.), *Information processing in social insects* (pp. 141-164). Berlin: Birkhäuser.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence.* New York: Oxford University Press.

Bruckstein, A. M. (1993). Why the ant trails look so straight and nice. *The Mathematical Intelligencer, 15*(2), 59-62.

Camazine, S., Deneubourg, J. L., Franks, N. R., Sneyd, J., Theraulaz, G., & Bonabeau, E. (2001). *Self-organization in biological systems*. Princeton: Princeton University Press.

Campos, M., Bonabeau, E., Theraulaz, G., & Deneubourg, J.-L. (2001). Dynamic scheduling and division of labor in social insects. *Adaptive Behavior, 8*(2), 83-92.

Cao, Y. U., Fukunaga, A. S., & Khang, A. B. (1997). Cooperative mobile robotics. *Autonomous Robots, 4,* 1-23.

Caprari, G., Estier, T., & Siegwart, R. (2000, April 24-28). Mobile micro-robots. *Proceedings of the IEEE International Conference on Robotics and Automation* (ICRA'2000), San Francisco.

Cicirello, A. C., & Smith, S. F. (2004). Wasp-like agents for distributed factory coordination. *Journal of Autonomous Agents and Multi-Agent Systems, 8*(3), 237-266.

Cicirello, V. A., & Smith, S. F. (2001, December). *Wasp-like agents for distributed fac-*

tory coordination. Technical Report CMU-RI-TR-01-39, Robotics Institute, Carnegie Mellon University, USA.

Cicirello, A.C., & Smith, S.F. (2002a). Distributed coordination of resources via wasp-like agents. In W. Truszkowski, C. Rouff, & M. Hinchey (Eds.), *Proceedings of Innovative Concepts for Agent-Based Systems: First International Workshop on Radical Agent Concepts* (WRAC-2002, pp. 71-80) (LNAI 2564). Berlin: Springer-Verlag .

Cicirello, C.A., & Smith, S.F. (2002b). Distributed coordination of resources via wasp-like agents. *Proceedings of the 1st Goddard/JPL Workshop on Radical Agent Concepts* (WRAC'02), NASA.

Deneubourg, J. L., Aron, S., Goss, S., Pasteels, J. M., & Duernick, G. (1986). Random behavior, amplification processes and number of participants: How they contribute to the foraging properties of ants. *Physica D, 22,* 176-186.

Drogoul, A., & Ferber, J. (1993). From Tom Thumb to the Dockers: Some experiments with foraging robots. In J. A. Meyer, H. L. Roitblat, & S.W. Wilson (Eds.), *From animals to animats* (pp. 451-459). London: MIT Press.

Ewers, H. H. (1927). *The ant people.* New York: Dodd, Mead & Company.

Gautrais, J., Jost, C., Jeanson, R., & Theraulaz, G. (2004). How individual interactions control aggregation patterns in gregarious arthropods. *Interaction Studies, 5,* 245-269.

Grassé, P. P. (1959). La reconstruction du nid et les coordinations inter-individuelles chez Belicositermes natalensis et Cubitermes sp. La théorie de la Stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux, 6,* 41-80.

Hölldobler, B., & Wilson, E. O. (1990). *The ants.* Cambridge: Harvard University Press.

Hofstadter, D. R. (1980). *Gödel, Escher, Bach: An eternal golden braid.* New York: Vintage Books.

Holland, O., & Melhuish, C. (1999). Stigmergy, self-organization, and sorting in collective robotics. *Artificial Life, 5,* 173-202.

Kittithreerapronchai, O., & Anderson, C. (2003, December 8-12). Do ants paint trucks better than chickens? Market versus response thresholds for distributed dynamic scheduling. *Proceedings of the 2003 IEEE Congress on Evolutionary Computation,* Canberra, Australia (pp. 1431-1439).

Krieger, M.J.B., Billeter, J.-B., & Keller, L. (2000). Ant-like tasks allocation and recruitment in cooperative robots. *Nature, 406,* 992-995.

Langton, C.G. (1986). Studying artificial life with cellular automata, *Physica*, 22D, 120-149.

Levy, S. (1992). *Artificial life.* London: Penguin Books.

Maeterlinck, M. (1927). *The life of the white ant.* London: George Allen and Unwin.

Monmarché, N., Venturini, G., & Slimane, M. (2000). On how *Pachycondyla apicalis* ants suggest a new search algorithm. *Future Generation Computer Systems, 16*(8), 937-946.

Moritz, R. F. A., & Southwick, E.E. (1992). *Bees as superorganisms.* Berlin: Springer-Verlag.

Morley, R. (1996). Painting trucks at General Motors: The effectiveness of a complexity-based approach. In *Embracing complexity: Exploring the application of complex adap-*

tive systems to business (pp. 53-58). Boston: Ernst & Young Center for Business Innovation.

Moyson, F., & Manderick, B. (1988). The collective behavior of ants: An example of self-organization in massive parallelism. *Proceedings of the AAAI Spring Symposium on Parallel Models of Intelligence*, Stanford, CA.

Nakrani, S., & Tovey, C. (2003, December 15-17). On honey bees and dynamic allocation in an Internet server colony. In C. Anderson & T. Balch (Eds.), *Proceedings of the 2nd International Workshop of the Mathematics and Algorithms of Social Insects* (pp. 115-122). Atlanta: Georgia Institute of Technology.

Schneirla, T. C. (1971). *Army ants: A study in social organization.* H. R. Topoff (Ed.). San Francisco: W. H. Freeman and Company.

Shao, C., & Hristu-Varsakelis, D. (2005). *A local pursuit strategy for bio-inspired optimal control with partially constrained final state.* Technical Research Report, TR2005-76, Institute for Systems Research, University of Maryland, USA.

Step, E. (1924). *Go to the ant.* London: Hutchinson & Co.

Sober, E., & Wilson, D. S. (1998). *Unto others.* Cambridge, MA: Harvard University Press.

Weidenmüller, A. (2004). The control of nest climate in bumblebee (*Bombus terrestris*) colonies: Interindividual variability and self-reinforcement in fanning response. *Behavioral Ecology, 15,* 120-128.

Weisstein, E. W. (2005). *Langton's ant.* Retrieved from http://mathworld.wolfram.com/LangtonsAnt.html

Wilson, E.O. (1984). The relation between caste ratios and division of labor in the ant genus *Pheidole* (Hymenoptera: Formicidae). *Behavioral Ecology and Sociobiology, 16,* 89-98.

Chapter III
Stochastic Optimization Algorithms

Pierre Collet
Université du Littoral Côte d'Opale, France

Jean-Philippe Rennard
Grenoble Graduate School of Business, France

ABSTRACT

When looking for a solution, deterministic methods have the enormous advantage that they do find global optima. Unfortunately, they are very CPU intensive, and are useless on untractable NP-hard problems that would require thousands of years for cutting-edge computers to explore. In order to get a result, one needs to revert to stochastic algorithms that sample the search space without exploring it thoroughly. Such algorithms can find very good results, without any guarantee that the global optimum has been reached; but there is often no other choice than using them. This chapter is a short introduction to the main methods used in stochastic optimization.

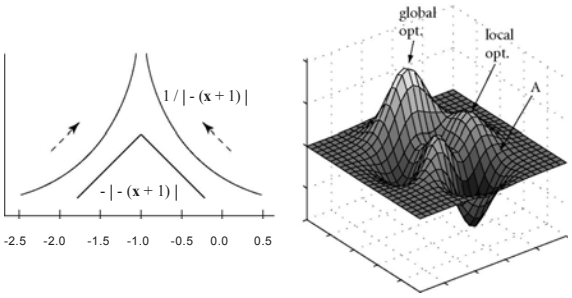
INTRODUCTION

The never-ending search for productivity has made optimization a core concern for engineers. Quick process, low-energy consumption, short and economical supply chains are now key success factors.

Given a space Ω of individual solutions $\omega \in R^n$ and an objective function f, $f(\omega) \rightarrow R$, optimizing is the process of finding the solution ω^* which minimizes (maximizes) f.

For hard problems, optimization is often described as a walk in a *fitness landscape*. First proposed by biologist S. Wright (1932), fitness landscapes aimed at representing the fitness of a living organism according to the genotype space. While optimizing, *fitness* measures the quality of a solution, and fitness landscapes plot solutions and corresponding goodness (fitness). If one wishes to optimize the function $x+1=0$, then depending on the choice of the error measure, fitness can for example be defined as $-|-(x+1)|$ or as $1/|-(x+1)|$. The optimization process then tries to find the peak of the fitness landscape (see Figure 1(a)).

Figure 1. (a) Fitness landscapes for $x + 1 = 0$. *(b) A multimodal fitness landscape.*



(a) *(b)*

This example is trivial and the optimum is easy to find. Real problems are often multimodal, meaning that their fitness landscapes contain several local optima (i.e., points whose neighbors all have a lower fitness; see Figure 1(b)). This is particularly true when variables interact with one another (*epistasis*).

Usual analytical methods, like gradient descent, are often unable to find a global optimum, since they are unable to deal with such functions. Moreover, companies mostly deal with combinatorial problems like quadratic assignment, timetabling, or scheduling problems. These problems using discrete states generate non-continuous objective functions that are unreachable through analytical methods.

Stochastic optimization algorithms were designed to deal with highly complex optimization problems. This chapter will first introduce the notion of complexity and then present the main stochastic optimization algorithms.

NP-Complete Problems and Combinatorial Explosion

In December, Santa Claus must prepare the millions of presents he has to distribute for Christmas. Since the capacity of his sleigh is finite, and he prefers to minimize the number of runs, he would like to find the best way to organize the packs. Despite the apparent triviality of the task, Santa Claus is facing a very hard problem. Its simplest formulation is the *one-dimensional bin packing problem*. Given a list $L = (a_1, a_2, ..., a_n)$ of *items* with sizes $0 < s(a_i) \leq 1$, what is the minimum number m of unit-capacity bins B_j such that $\sum_{a_i \in B_j} s(a_i) \leq 1, 1 \leq j \leq m$? This problem is known to be *NP-hard* (Coffman, Garey, & Johnson, 1996).

Various forms of the *bin packing problem* are very common. The transportation industry must optimize truck packing given weight limits, the press has to organize advertisements minimizing the space, and the sheet metal industry must solve the *cutting-stock problem* (how to minimize waste when cutting a metal sheet).

Such problems are very tough because we do not know how to build algorithms that can solve them in *polynomial-time*; they are said to be *intractable problems*. The only algorithms we know for them need an *exponential-time*. Table 1 illustrates the evolution of time algo-

Table 1. Polynomial vs. non-polynomial functions complexity growth

O(N)	N=17	N=18	N=19	N=20
N	17×10^{-9} s	18×10^{-9} s	19×10^{-9} s	20×10^{-9} s
N^2	289×10^{-9} s	324×10^{-9} s	361×10^{-9} s	400×10^{-9} s
N^5	1.4×10^{-3} s	1.8×10^{-3} s	2.4×10^{-3} s	3.2×10^{-3} s
2^N	131×10^{-6} s	262×10^{-6} s	524×10^{-6} s	1×10^{-3} s
5^N	12.7 mn	1 h	5.29 h	26.4h
TSP	2.9 h	2 days	37 days	2 years !
$N!$	4 days	74 days	4 years	77 years !

Considering 10^9 *operations per second, evolution of the algorithm time according to its complexity. TSP stands for Traveling Salesman Problem, with complexity* $\frac{(N-1)!}{2}$ *for N towns (see following sections).*

rithms for polynomial-time problems vs. non-polynomial. Improving the speed of computers or algorithms is not the solution, since if the speed is multiplied, the gain of time is only additive for exponential functions (Papadimitriou & Steiglitz, 1982).

The consequences of the computational complexity for a great many real-world problems are fundamental. Exact methods for scheduling problems "become computationally impracticable for problems of realistic size, either because the model grows too large, or because the solution procedures are too lengthy, or both, and heuristics provide the only viable scheduling techniques for large projects" (Cooper, 1976).

Heuristics and Meta-Heuristics

Since many real-world combinatorial problems are NP-hard, it is not possible to guarantee the discovery of the optimum. Instead of exact methods, one usually uses *heuristics*, which are approximate methods using iterative trial-and-error processes to approach the best solution. Many of them are nature inspired, and their latest development is to use *metaheuristics*.

A metaheuristic is an iterative master process that guides and modifies the operations of subordinate heuristics to efficiently produce high-quality solutions. It may manipulate a complete (or incomplete) single solution or a collection of solutions at each iteration. The subordinate heuristics may be high (or low) level procedures, or a simple local search, or just a construction method. (Voss, Martello, Osman, & Roucairol, 1999)

Metaheuristics are high-level methods guiding classical heuristics. They deal with a dynamic balance between *diversification* (exploration of the solution space) and *intensification* (exploitation of the accumulated knowledge) (Blum & Roli, 2003).

Stochastic Algorithms

Random Search

Random search is what it says it is. In essence, it simply consists of picking up random potential solutions and evaluating them. The best solution over a number of samples is the result of random search.

Many people do not realize that a stochastic algorithm is nothing more than a random search, with hints by a chosen heuristics (or meta-heuristics) to guide the next potential solution to evaluate. People who realize this feel uneasy about stochastic algorithms, because there is not a guarantee that such an algorithm (based on random choices) will *always* find the global optimum.

The only answer to this problem is a probabilistic one:

- if, for a particular problem, one already knows the best solution for different instances of this problem, and
- if, over a significant number of runs, the proposed stochastic algorithm finds a solution that in average is 99% as good as the known optimum for the tested instances of the problem, then,
- one can hope that on a new instance of the problem for which the solution is not known, the solution found by the stochastic algorithm will be 99% as good as the unknown optimum over a significant number of runs.

This claim is not very strong, but there are not many other options available: if one absolutely wants to get *the* global optimum for a

large NP-hard problem, the only way is to let the computer run for several hundred years (cf. Table 1). The stochastic way is therefore a pragmatic one.

Computational Effort

As can be seen above, it is difficult to evaluate the performance of stochastic algorithms, because, as Koza (1994) explains for genetic programming:

Since genetic programming is a probabilistic algorithm, not all runs are successful at yielding a solution to the problem by generation G. When a particular run of genetic programming is not successful after the prespecified number of generations G, there is no way to know whether or when the run would ever be successful. When a successful outcome cannot be guaranteed for every run, there is no knowable value for the number of generations that will yield a solution….

Koza therefore proposes a metrics to measure what he calls the *computational effort* required to solve a problem that can be extended to any stochastic algorithm where evaluations consume a significant fraction of the computer resources:

One first calculates $P(n)$, the cumulative probability of success by dividing the number of runs that succeeded on or before the n^{th} evaluation, by the number of conducted runs.

The computational effort $I(n,z)$ can then be defined as the number of evaluations that must be computed to produce a satisfactory solution with probability greater than z (where z is usually 99%), using the formula:

$$n * \left\lceil \frac{\ln(1-z)}{\ln(1-P(n))} \right\rceil .$$

No Free Lunch Theorem

Random search is also important because it serves as a reference on which one can judge stochastic algorithms. A very important theorem is that of the *No Free Lunch* (Wolpert & Macready, 1995). This theorem states that no search algorithm is better than a random search on the space of all possible problems—in other words, if a particular algorithm does better than a random search on a particular type of problem, it will not perform as well on another type of problem, so that all in all, its global performance on the space of all possible problems is equivalent to a random search.

The overall implication is very interesting, as it means that an off-the-shelf stochastic optimizer cannot be expected to give good results on any kind of problem (no free lunch): a stochastic optimizer is not a black box: to perform well, such algorithms must be expertly tailored for each specific problem.

Hill-Climbing

Hill-climbing is the basis of most *local search* methods. It is based on:

- A set of feasible solutions $\Omega = \{\omega; \omega \in R^n\}$.
- An objective function $f(\omega)$ that can measure the quality of a candidate solution.
- A neighborhood function $N(\omega) = \{\omega_n \in \Omega \mid dist(\omega_n, \omega) \leq \delta\}$ able to map any candidate solution to a set of close candidate solutions.

The optimization algorithm has to find a solution $\omega^*, \forall \omega \in \Omega, f(\omega^*) \leq f(\omega)$. The basic hill-climbing algorithm is trivial:

1. Build a candidate solution $\omega \in \Omega$.
2. Evaluate ω by computing $f(\omega)$ and set $\omega^* \leftarrow \omega$.

3. Select a neighbor $\omega_n \in N(\omega)$ and set $\omega \leftarrow \omega_n$.
4. If $f(\omega) \leq f(\omega^*)$ set $\omega^* \leftarrow \omega$.
5. If some stopping criterion is met, exit else go to 3.

One example is the famous Traveling Salesman Problem (TSP: given a collection of cities, finding the shortest way of visiting them all and returning back to the starting point), which is an NP-hard problem (cf. Table 1 for complexity). The candidate solution is a list of cities, for example *F-D-B-A-E-C,* and the objective function is the length of this journey. There are many different ways to build a neighborhood function. *2-opt* (Lin, 1965) is one of the simplest since it just reverses a sequence. Applying *2-opt* could lead to *F-E-A-B-D*-C. The new tour will be selected if it is shorter than the previous one, otherwise one will evaluate another neighbor tour.

More advanced hill-climbing methods look for the best neighbor:

1. Build a candidate solution $\omega \in \Omega$.
2. Evaluate ω by computing $f(\omega)$.
3. For each neighbor $\omega_n \in N(\omega)$, evaluate $f(\omega_n)$.
4. If all $f(\omega_n)$ are $\geq f(\omega)$ (local optimum) then exit.
5. Else select $\omega^*, \forall \omega_n \in N(\omega), f(\omega^*) < f(\omega_n)$ as the current candidate solution and set $\omega \leftarrow \omega^*$.
6. Go to 3.

The main advantage of hill-climbing is its simplicity, the core difficulty usually being the design of the neighborhood function. The price for this simplicity is a relative inefficiency. It is trivial to see that hill-climbing is easily trapped in local minima. If one starts from point A (see Figure 1b), it will not be able to reach the global optimum, since once on top of the first peak, it will not find any better point and will get stuck there.

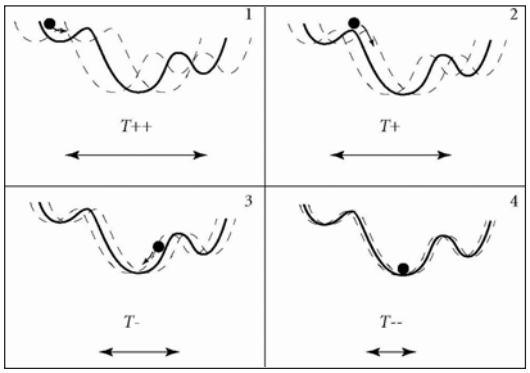
Even though many advanced forms of hill-climbing have been developed, these methods are limited to smooth and unimodal landscapes. "A question for debate in medieval theology was whether God could create two hills without an intervening valley…unfortunately, when optimizing functions, the answer seems to be no" (Anderson & Rosenfeld, 1988, p. 551). This is why search rules based on local topography usually cannot reach the highest point.

Simulated Annealing

Simulated annealing (Kirkpatrick, Gellat, & Vecchi, 1983) is an advanced form of hill-climbing. It originates in metallurgy. While annealing a piece of metal, quickly lowering the temperature leads to a defective crystal structure, far from the minimum energy level. Starting from a high temperature, cooling must be progressive when approaching the freezing point in order to obtain a nearly perfect crystal, which is a crystal close to the minimum energy level. Knowing that the probability for a system to be at the energy level E_0 is $p(E_0)=\exp(-E_0/k_BT)/Z(T)$, where k_B is the *Boltzmann constant*, T the temperature, and $Z(T)$ a normalizing function, Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1955) proposed a simple algorithm to simulate the behavior of a collection of atoms at a given temperature At each iteration, a small random move is applied to an atom and the difference of energy ΔE is computed. If $\Delta E \leq 0$ the new state is always accepted. If $\Delta E > 0$ the new state is accepted according to a probability $p(\Delta E) = \exp(-\Delta E/k_BT)$.

Simulated annealing is based on a series of Metropolis algorithms with a decreasing temperature. It can shortly be described this way:

Figure 2. Simulated annealing



1. Build a candidate solution $\omega \in \Omega$.
2. Evaluate ω by computing $f(\omega)$.
3. Select a neighbor candidate solution $\omega_n \in N(\omega)$.
4. If $f(\omega_n) \leq f(\omega)$ then set $\omega \leftarrow \omega_n$ and exit if the evaluation is good enough.
5. Else select ω_n ($\omega \leftarrow \omega_n$) according to the probability: $p = \exp(-(f(\omega_n) - f(\omega))/T_i)$ where T_i is the current temperature which decreases over time.
6. Go to 3.

Uphill moves (step 5) allow overcoming local minima. One can illustrate the difference between hill-climbing and simulated annealing with the rolling ball metaphor (see Figure 2). Imagine a ball on a bumpy surface. The ball will roll down and stop at the first point of minimum elevation which usually is a local optimum. By tolerating uphill moves, simulated annealing somehow "shakes" the surface pushing the ball beyond the local minimum. At the beginning of the process, the surface is brutally shaken—the temperature is high—allowing a large exploration. The reduction of the temperature progressively decreases the shaking to prevent the ball from leaving the global optimum.

Simulated annealing is efficient, but slow. Many improvements have been proposed, like the *rescaled simulated annealing* which limits the transitions in a band of energy centered around a target energy level by using $\Delta E_{ij} = (\sqrt{E_j} - \sqrt{E_t})^2 - (\sqrt{E_i} - \sqrt{E_t})^2$, with typically $E_t = \alpha T^2, \alpha > 0$ (Hérault, 2000). This method "flattens" the error surface at the beginning of the process, minimizing the tendency of the algorithm to jump among local minima.

Tabu Search

"Tabu search may be viewed as 'meta-heuristic' superimposed on another heuristic. The approach undertakes to transcend local optimality by a strategy of forbidding certain moves" (Glover, 1986—this is the first appearance of the term 'meta-heuristic'). Like simulated annealing, it is as an advanced form of hill-climbing, based on a set of feasible solutions Ω, an objective function $f(\omega)$, and a neighborhood function $N(\omega)$. Tabu search tries to overcome local minima by allowing the selection of non-improving solutions and by using a procedure which avoids cycling moves. Unlike simulated annealing, the probability of selection of a non-improving move is not applied to a given neighbor, but to the set of neighbors. To avoid cycles, tabu search implements a list T of tabu moves which in the basic form contains the t last moves. The simple tabu search works as follows (Glover, 1989):

1. Select a potential solution $\omega \in \Omega$ and let $\omega^* \leftarrow \omega$. Initialize the iteration counter $k = 0$ and let $T = \varnothing$.
2. If $N(\omega) - T = \varnothing$ go to 4. Otherwise, increment k and select $\omega_b \in N(\omega) - T$ the "best" available move.
3. Let $\omega \leftarrow \omega_b$. If $f(\omega) < f(\omega^*)$, let $\omega^* \leftarrow \omega$.
4. If ω^* is equal to the desired minimum or if $N(\omega) - T = \varnothing$ from 2, stop. Otherwise update T and go to 2.

We did not define the "best" available move at step 2. The simplest—nevertheless powerful—way is to select ω_b such that $\forall \omega_n \in N(\omega) - T, f(\omega_b) < f(\omega_n)$. This means that the algorithm can select a non-improving move since $f(\omega_b)$ can be greater than $f(\omega^*)$.

The definition of the tabu list (step 4) is also a central one. This list aims at escaping local minima and avoiding cycles. It should then consider as tabu any return to a previous *solution state.* If s^{-1} is the reverse move of s, the tabu list can be defined such that $T = \{s_h^{-1} : h > k - t\}$, where k is the iteration index and t defines the size of the time window. Practically, this method is hard to implement, especially because of memory requirement. One usually stores only partial ranges of the moves attributes, which can be shared by other moves. The tabu list then contains collections C_h of moves sharing common attributes: $T = \cup C_h; h > k - t$, where $s_h^{-1} \in C_h$ (Glover, 1989).

Since the tabu list manages moves and not solutions, unvisited solutions can have tabu status. In order to add flexibility to the research process, tabu search uses *aspiration levels.* In its simplest form, the aspiration level will allow tabu moves whose evaluation has been the best so far.

Two extra features are usually added to tabu search (Glover, 1990): *intensification* and *diversification.* These terms can be added to the objective function $\tilde{f} = f + intensification + diversification$.

Intensification aims at closely examining "interesting" areas. The intensification function will favor solutions close to the current best. The simplest way is to get back to a close-to-the-best solution and to reduce the size of the tabu list for some iterations. More sophisticated methods use *long-term memory* memorizing the good components of good solutions.

Diversification aims at avoiding a too local search. The diversification function gives more weight to solutions far from the current one. The simplest way to implement it is to perform random restarts. One can also penalize the most frequent solutions components.

Let us examine a simple example to illustrate tabu search.

The cube (see Figure 3) shows the cost and the neighborhood of an eight-configurations problem. The random initial configuration is, for example, 10. We will simply define the tabu movements as the reverse movements in each of the three directions, that is if we move along $x+$, the movement $x-$ will be tabu.

First iteration:

- Neighborhood of 10 is 15, 8, and 12.
- The best move is $z+$ which selects 8.
- $z-$ is added to the tabu list.

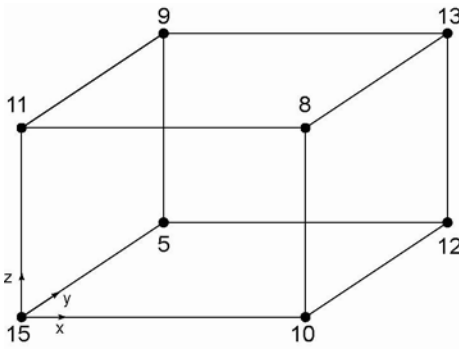
Second iteration:

- Neighborhood is 11, 13, and 10.
- The best move is $z-$, but it is tabu. The second best move is $x-$ which selects 11.
- $x+$ is added to the tabu list.

Third iteration:

- Neighborhood is 9, 8, and 15.
- The best move is $x-$, but it is tabu. The second best move is $y+$ which selects 9.

Figure 3. An illustration of tabu search



- *y*– is added to the tabu list.

Fourth iteration:

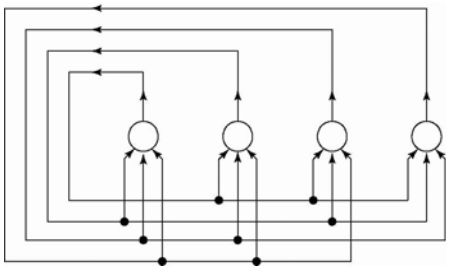
- Neighborhood is 11, 13, and 5.
- The best move is *z*– which is tabu, but its evaluation is 5 which is lower than the best evaluation so far (8). The aspiration criterion overrides the tabu restriction and the 5 is selected.
- 5 is the global minimum, the research is over.

Despite its simplicity, tabu search is a highly efficient algorithm. It is known to be one of the most effective meta-heuristics for solving the job-shop scheduling problem (Taillard, 1994; Watson, Whitley, & Howe, 2003). It is used in many different fields like resources planning, financial analysis, logistics, and flexible manufacturing.

Neural Networks

A neural network is a set of processing units linked by "learnable connections." They are well known in the field of artificial intelligence where they notably provide powerful generalization and clustering tools. Some recurrent neural networks are also useful for optimization. The optimization process is usually based on the minimization of an energy function defined as: $E(x) = E_c(x) + \sum_k a_k E_k(x)$ where E_c is the cost function, $E_k(x)$ are the penalties associated to constraint violations, and a_k are the associated weighting parameters. For many optimization problems, the cost function is expressed in a quadratic form $E(x) = -1/2 \sum_{i,j} T_{ij} s_i s_j - \sum_i I_i s_i$, where s_i is the signal of the neuron i, $T_{ij} = \partial^2 E / \partial s_i \partial s_j$, and $I_i = \partial E / \partial s_i$ (Dreyfus et al., 2002).

Figure 4. Hopfield network



Hopfield networks (Hopfield, 1982) are the most famous neural networks used for optimization. They are asynchronous (one randomly selected neuron is updated at each step), fully connected—except self-connection—neural networks (see Figure 4).

The binary version uses a sign function; the output signal of a neuron is computed as: $s_i = 1$ if $\sum_j w_{ji} s_j - \theta_i \geq 0$, $s_i = 0$ otherwise; where w_{ji} is the weight of the connection between neurons j and i, s_j is the signal of the neuron j and θ_i is the bias of neuron i (a constant, usually negative, signal).

Such a network is a dynamic system whose attractors are defined by the minima of the energy function defined as:

$$E = -1/2 \sum_{i,j} w_{ij} s_i s_j - \sum_j \theta_j s_j$$

Originally, Hopfield designed his networks as associative memories. Data are stored in the attractors where the network converges, starting from partial or noisy data providing a *content-addressable memory*. In 1985, Hopfield demonstrated the optimizing capabilities of his network, applying it to the TSP problem (Hopfield & Tank, 1985).

Table 2. Hopfield matrix for the TSP

	1	2	3	4	5
A	0	1	0	0	0
B	0	0	0	1	0
C	1	0	0	0	0
D	0	0	0	0	1
E	0	0	1	0	0

While using Hopfield networks for optimization, the main difficulty is the representation of the problem and the definition of the objective function as the energy of the network. For n cities, Hopfield and Tank used n^2 neurons. A set of n neurons was assigned to each city, and the rank of the firing neuron designed the rank of the city during the travel.

Table 2 represents the tour *C-A-E-B-D*. The energy function depends on constraints and cost. For the TSP, the constraints define the validity of the tour that is the fact that each city is visited once. Hopfield and Tank defined the corresponding function as:

$$A/2\sum_x\sum_i\sum_{j\neq i}V_{xi}V_{xj} + B/2\sum_i\sum_x\sum_{x\neq y}V_{xi}V_{yi} + C/2(\sum_x\sum_i V_{xi} - n)^2$$

where V_{xi} is the binary signal of the neuron representing the city x at the position i ($V_i = 0$ or $V_i = 1$) and A, B, C are constant. The first term is zero when each row contains one "1" (cities are visited once), the second is zero when each column contains one "1" (there is one city per position), and the third term is zero when the matrix contains exactly n "1."

The cost function depends on the length of the tour. It is defined as:

$$D/2\sum_x\sum_{y\neq x}\sum_i d_{xy}V_{xi}(V_{y,i+1} + V_{y,i-1})$$

where d_{xy} is distance between cities x and y and where $V_{x,n+j} = V_{x,j}$. The energy function is the sum of the four terms. If the constants are large enough ($A = B = 500$, $C = 200$, $D = 500$ in the initial tests), low energy states will correspond to valid tours. The matrix of the connection weights becomes:

$$w_{xi,yj} = -A\delta_{xy}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{xy}) - C - Dd_{xy}(\delta_{j,i+1} + \delta_{j,i-1})$$

where $w_{xi,yj}$ is the weight of the connection between the neurons representing the city x at position i and the city y at position j and $\delta_{ij} = 1$ if $i = j$, $\delta_{ij} = 0$ otherwise.

The original model of Hopfield-Tank has been quite controversial since their results have proved to be very difficult to reproduce. Thanks to posterior improvements (e.g., *Boltzmann machine* which tries to overcome local minima by using a stochastic activation function (Ackley, Hinton, & Sejnowski, 1985)), the Hopfield-Tank model demonstrated its usefulness. It is notably used to solve general (e.g., Gong, Gen, Yamazaki, & Xu, 1995) and quadratic (e.g., Smith, Krishnamoorthy, & Palaniswami, 1996) assignment problems; Cutting stock problems (e.g., Dai, Cha, Guo, & Wang, 1994), or Job-Shop scheduling (e.g., Foo & Takefuji, 1988).

Apart from Hopfield networks, T. Kohonen Self-Organizing Maps (SOM) (Kohonen,1997), initially designed to solve clustering problems, are also used for optimization (Smith, 1999), notably since the presentation of the *Elastic Net Method* (Durbin & Willshaw, 1987).They are especially used to solve quadratic assignment problems (Smith, 1995) and vehicle routing problems (e.g., Ramanujam & Sadayappan, 1995).

Evolutionary Algorithms and Genetic Programming

Evolutionary algorithms and genetic programming have chapters devoted to them in this book, so this section will remain small and general. Evolutionary algorithms provide a way to solve the following interesting question. Given: (1) a very difficult problem for which no way of finding a good solution is known and where a solution is represented as a set of parameters, and (2) a number of previous trials that have all been evaluated, how can one use the accumulated knowledge to choose a new set of parameters to try out (and therefore do

better than a random search)? One could store all the trials in a database and perform statistics on the different parameters characterizing the trials, to try to deduce some traits that will lead to better results. However, in real life, parameters are often interdependent (*epistasis*), so drawing conclusions may not be that easy, even on a large amount of data.

Evolutionary algorithms (EAs) rely on artificial Darwinism to do just that: exploit each and every trial to try out new potential solutions that will hopefully be better than the previous ones: given an initial set of evaluated potential solutions (called a population of individuals), "parents" are *selected* to "give birth" to "children" thanks to "genetic" operators, such as "crossover" and "mutation." "Children" are then evaluated and form the pool of "parents" and "children," a *replacement* operator selects those that will make it to the new "generation."

As can be seen in the previous paragraph, the biological inspiration for this paradigm led to borrowing vocabulary specific to this field.

The *selection* and *replacement* operators are the driving forces behind artificial evolution. They are biased towards good individuals, meaning that (all in all) the population is getting better along as the generations evolve. A too strong selection pressure will lead to a premature convergence (the population of individuals will converge towards a local optimum) while a too weak selection pressure will prevent any convergence.

Evolutionary algorithms can be used to optimize virtually any kind of problem, even some that cannot be formalized. This makes them usable for interactive problems where the fitness of an individual is given by a human operator (see Chapter XXVII in this handbook). They are also very efficient on multi-objective problems, thanks to the fact that they evolve a whole population of individuals at once (see Chapter VI in this book). Proper techniques (such as NSGA-II (Deb, Agrawal, Pratab, & Meyarivan, 2000)) can be used to create a full Pareto-front in only one run (something impossible to do with simulated annealing or tabu search, for instance).

If EAs can be used for virtually anything, why not try to evolve programs? This is what genetic programming (detailed in Chapter V) is about. Individuals are not merely a set of parameters that need to be optimized, but full programs that are run for evaluation. The main difference between genetic programming and standard EAs is that individuals are executed to be evaluated (rather than used in an evaluation function).

Data-Level Parallelism

The powerful data-level parallelism trend appeared in the mid-1980s with many seminal works (Wolfram, 1984; Rumelhart & McClelland, 1986; Minsky, 1986; Hillis & Steele, 1986; Thinking Machines Corporation, 1986). Object-oriented programming allows one to embed intelligent behavior at the data level. The idea is then used to put together many small intelligent entities, possibly on parallel machines or, even better, a connection machine (cf. CM-1 by D. Hillis of *Thinking Machines*, with 64K processors, 1986).

One of the first applications of the Connexion Machine was to implement particles simulating perfect ball bearings that could move at a single speed in one of six directions, and only connected to their nearest neighbors. The flow of these particles on a large enough scale was very similar to the flow of natural fluids. Thanks to data-level parallelism, the behavior of a complex turbulent fluid—which would have used hours of computation to simulate using Navier-Stokes equations—could be much simply obtained thanks to the parallel evolution of the elementary particles.

The notion of turbulence was not specifically implemented in the behavior of the elementary particles: it just *emerged* when they were put together and set in motion.

Still in 1986, Reynolds implemented a computer model to simulate coordinated animal motion such as bird flocks and fish schools. He called the creatures *boids* (Reynolds, 1987). What he discovered was that one could obtain a flocking or schooling *emergent* behavior by implementing the very simple three following rules into individual boids:

1. **Separation:** Steer to avoid getting too close from local flockmates.
2. **Alignment:** Steer towards the average heading of local flockmates.
3. **Cohesion:** Steer to move towards the average position (center of gravity) of local flockmates.

Note that there is no global knowledge of the position or shape of the flock of boids. Boids only know about the position of their neighbors (local flockmates).

Particle Swarm Optimization

In 1975, Wilson (a sociobiologist) wrote, in reference to fish schooling: "In theory at least, individual members of the school can profit from the discoveries and previous experience of all other members of the school during the search for food" (Wilson, 1975). In 1995, Kennedy and Eberhart, who were aware of the work of Reynolds, saw in this statement that swarms (or fish schools) could possibly be used to find specific places (such as optima) in a search space.

Their first tests on known problems were quite positive, so they refined the model. They quickly saw that rules number 1 and 2 were not needed, so they used Occam's razor to remove them, but visually, this changed the flock behavior into a swarm behavior, hence the *particle swarm optimization* (PSO) name (Kennedy & Eberhart, 1995).

Particles collaborate as a population to find the best possible solution to a problem. Each particle (a potential solution to the problem made of an n-dimensions array of parameters) knows of the position of the best solution ever found by the swarm (called *gbest*) and of the best solution it ever found (called *pbest*). However, going directly to either *pbest* or *gbest* is pointless because these points have already been visited. The idea behind PSO is to have the particles go towards both *pbest* and *gbest* with inertia: it is their speed that is modified rather than directly their position.

In the original algorithm, each particle p has access to:

- its current position in each of the dimensions *i* of the problem at hand: p.pos[i];
- the best solution it has personally found, that is, p.pBestVal (value of the best found solution) and for each dimension *i*: p.pBestPos[i];
- a velocity for each dimension *i*: p.Veloc[i]; and
- the best solution found by the particle swarm: gBest.Val and for each dimension *i*: gBest.pos[i].

Then, the algorithm runs as follows:

1. Initialize randomly all particles of the population, evaluate them, and set their pBestVal field to 0.
2. Clone the best particle and copy it into gBest, and for all particles, set their pBest positions to their current positions if their current value is greater than their pBestVal value.

3. Calculate new velocities for all particles and for all dimensions:

   p.Veloc[i]=p.Veloc[i]
   +pIncrement*rand()*(p.pBest[i]-p.Pos[i])
   +gIncrement*rand()*(gBest.pos[i]p.Pos[i]

   where rand() returns a random value between 0 and 1.

4. Move all particles using the calculated velocities, that is, for all dimensions of all particles: p.pos[i]=p.pos[i]+p.Veloc[i].
5. Evaluate each particle and go back to step 2 until a stopping criterion is met.

In this algorithm, particles are attracted by two locations: the place where they found their personal best result pBest and some kind of public knowledge of where the best spot found by the swarm lies gBest. Two increments are associated to adjust the velocity towards these particular locations. In their original paper, Kennedy and Eberhart called pIncrement pressure for "simple nostalgia," while gIncrement represents pressure towards group knowledge.

Simulations showed that a high value of pIncrement relative to gIncrement results in excessive wandering of isolated individuals through the problem space, while the reverse (high gIncrement vs. pIncrement) results in premature convergence towards a local optimum. Approximately equal values seem to give the best result. A value of 2 was given by Kennedy and Eberhart to give rand() a mean of 1, meaning that agents would have equal chances of moving too fast or too slow towards their aim.

Particle Swarm Optimization uses an elegant and simple algorithm to implement a swarm of individuals evolving in a fitness landscape thanks to which an optimal solution emerges. To quote Kennedy and Eberhart, "much of the success of particle swarms seems to lie in the agents' tendency to hurtle past their target." The model was later on generalized in Shi and Eberhart (1998).

Ant Colony Optimization

Ant colony optimization is another nature-inspired algorithm, based on data-level parallelism and the concept of emergence. The idea comes from a biology paper (Deneubourg, Pasteels, & Verhaeghe, 1983) which describes the very simple mechanism that ants use to establish an optimal path between a food source and their ant-hill, without central supervision.

This paper came out in 1983, while the data-level parallelism trend was blooming. In 1988, Manderick and Moyson saw the emergent process lying in this description and wrote a seminal paper (Moyson & Manderick, 1988) in which they describe the implementation of virtual ants on a computer. The appendix of the paper contains the equations that govern the behavior of ants, allowing computation of the critical mass of ants above which self-organization emerges at the macroscopic level.

Colorni, Dorigo, and Maniezzo (1991) describe a *Distributed Optimization by Ant Colonies,* and Dorigo's (1992) PhD thesis uses virtual ants to solve the Traveling Salesman Problem (TSP). Many other papers followed, describing ant colony optimization (Stützle & Dorigo, 2002; Maniezzo, Gambardella, & Luigi, 2004; Dorigo & Caro, 1999).

Ant colony optimization is based on stigmergy, evaporation, and errors. Real or artificial ants communicate by leaving global information in their environment (*stigmergy*) under the form of *pheromones* (odors) that evaporate with time:

- Foraging ants leave the ant-hill with the aim of bringing back food for the commu-

nity. If there are no pheromones around, they walk randomly. Otherwise, they tend to follow pheromone trails proportionally to the amount of pheromones on the trail.

- If ants find a food source, they detach and carry a bit of the food, and continue walking either randomly, or following a pheromone trail. In either case, ants carrying food leave behind them a trail of pheromones. If they do not follow a pheromone trail and walk randomly, they create a new pheromone trail. If they follow a pheromone trail, they will reinforce it.

The described algorithm is really simple. Robustness and adaptability reside in the volatility of stigmergic information (evaporating pheromones) and the stochastic following of existing paths (error in trail following).

If the food source depletes, ants will not be able to carry food anymore, and will not reinforce the trail leading to the food source. The trail leading to a depleted food source will therefore fade away and disappear automatically, thanks to pheromone evaporation.

If an existing trail is interrupted because of an external cause, ants coming from the food source will start walking randomly around the obstacle, until by chance, the trail leading to the ant-hill is found again. If two alternative solutions are found, the traffic on the shortest one will be greater, meaning that the pheromone scent will be stronger. The shorter trail will therefore appear more attractive to ants coming to the point where they must choose which way they want to go. After a while, the longest path disappears in favor of the shortest.

Solving the TSP with Ant Colony Optimization

An implementation of a TSP solver using ant colony optimization is well described in Stützle and Dorigo (1999): Cities are placed on a graph, with edges bearing the distance between the two connected cities. A number of artificial ants are placed on random cities and move to other cities until they complete a tour. A fitness is computed for each edge of the graph, which is a weighted sum between the amount of pheromones already borne by the edge and the inverse of the distance to the town pointed to by the edge.

At each time step, artificial ants probabilistically choose to follow an edge depending on its fitness value, and deposit an amount of pheromone, to indicate that the edge has been chosen (local updating). This goes on until all ants have completed a tour. When this is done, the ant that has found the shortest tour deposits pheromones on the edges of its tour proportionally to the inverse of the total distance (global updating), and all ants are restarted again from random towns.

This algorithm is very efficient: *HAS-SOP* (Hybridized Ant System for the Sequential Ordering Problem) is one of the fastest known methods to solve Sequential Ordering Problems (a kind of asymmetric TSP with multiple precedence constraints) (Gambardella & Dorigo, 2000).

CONCLUSION

Stochastic optimization has known several eras, where different algorithm techniques have blossomed as they were discovered and used. Simulated annealing and neural networks were certainly among the first. They are therefore very well described and widely used in industry and applied sciences.

However, other algorithms (sometimes as ancient as the previously quoted ones) have come of age, thanks to the computation power of modern computers, and they should not be

ignored since they bring many advantages over the old ones. Evolutionary algorithms, for instance, are very good at tackling multi-objective problems, since they can provide a complete Pareto-front in only one run. Genetic programming can be used for symbolic regression problems or to design analog circuits that are human competitive. More recent algorithms, based on the concept of data-level parallelism, allow the crossing of levels of abstraction in order to find emergent solutions, possibly leading to "a new kind of science" (Wolfram, 2002).

One important thing to keep in mind though is the *No Free Lunch* theorem, which states that no black-box will ever be able to solve any kind of problem better than a random search. The conclusion is that for a specific class of problem, some algorithms will work better than others, and that all these heuristics need to be tailored for a particular application if one really wants to obtain outstanding results.

REFERENCES

Ackley, D., Hinton, G., & Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science, 9,* 147-169.

Anderson, J. A., & Rosenfeld, E. (Eds.). (1988). *Neurocomputing*. Cambridge, MA: MIT Press.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys, 35*(3), 268-308.

Coffman, E. G., Garey, M. R., & Johnson, D. S. (1996). Approximation algorithms for bin packing: A survey. In D. Hochbaum (Ed.), *Approximation algorithms for NP-hard problems* (pp. 46-93). Boston: PWS Publishing.

Colorni, A., Dorigo, M., & Maniezzo, V. (1991). Distributed optimization by ant colonies. In B. F. Varela (Ed.), *Proceedings of the 1st European Conference on Artificial Life* (pp. 134-142). Paris: Elsevier.

Cooper, D. F. (1976). Heuristics for scheduling resource-constrained projects: An experimental investigation. *Management Science,* (22), 1186-1194.

Dai, Z., Cha, W., Guo, W., & Wang, F. (1994). A heuristic-based neural network for packing problems. *Proceedings of the International Conference on Data and Knowledge Systems for Manufacturing and Engineering 2* (pp. 698-703).

Deb, K., Agrawal, S., Pratab, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *Proceedings of the 6th Conference on Parallel Problem Solving from Nature* (pp. 849-858). Berlin: Springer-Verlag (LNCS 1917).

Deneubourg, J.-L., Pasteels, J.-M., & Verhaeghe, J.-C. (1983). Probabilistic behavior in ants: A strategy of errors? *Journal of Theoretical Biology,* 105.

Dorigo, M. (1992). *Optimization, learning and natural algorithms*. Dipartimento di Electronica, Politecnico di Milano, Italy.

Dorigo, M., & Caro, G.D. (1999). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimization* (pp.101-117). London: McGraw-Hill.

Dreyfus, G., Martinez, J.-M., Samuelides, M., Gordon, M. B., Badran, F., Thiria, S. et al. (2002). *Réseaux de neurones*. Paris: Eyrolles.

Durbin, R., & Willshaw, D. (1987). An analogue approach to the traveling salesman problem using an elastic net method. *Nature, 326,* 689-691.

Foo, Y., & Takefuji, Y. (1988). Stochastic neural networks for job-shop scheduling: Parts 1 and 2. *Proceedings of the IEEE International Conference on Neural Networks 2* (pp. 275-290).

Gambardella, L., & Dorigo, M. (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing, 12*(3), 237-255.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research, 13*(5), 533-549.

Glover, F. (1989). Tabu search—Part I. *ORSA Journal on Computing, 1*(3), 190-206.

Glover, F. (1990). Tabu search—Part II. *ORSA Journal on Computing, 2*(3), 4-32.

Gong, D., Gen, M., Yamazaki, G., & Xu, W. (1995). Neural network approach for general assignment problem. *Proceedings of the International Conference on Neural Networks 4* (pp. 1861-1866), Perth.

Hérault, L. (2000). Rescaled simulated annealing—Accelerating convergence of simulated annealing by rescaling the states energy. *Journal of Heuristics, 6,* 215-252.

Hillis, W. D., & Steele, G. L. (1986). Data parallel algorithms. *Communications of the ACM, 29*(12).

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences,* (79), 2554-2558.

Hopfield, J. J., & Tank, D. W. (1985). "Neural" computation of decisions in optimization problems. *Biological Cybernetics, 52,* 141-152.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of the IEEE International Conference on Neural Networks* (vol. 4, pp. 1942-1948).

Kirkpatrick, S., Gellat, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*(4598), 671-680.

Kohonen, T. (1997). *Self-organizing maps.* Heidelberg: Springer.

Koza, J. R. (1994). *Genetic programming II: Automatic discovery of reusable programs.* Cambridge, MA: MIT Press.

Lin, S. (1965). Computer solutions to the traveling salesman problem. *Bell System Technical Journal, 44,* 2245-2269.

Maniezzo, V., Gambardella, L. M., & Luigi, F. D. (2004). Ant colony optimization. In G.C. Onwubolu & B.V. Babu (Eds.), *New optimization techniques in engineering* (pp.101-117). Berlin, Heidelberg: Springer-Verlag.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1955). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics, 21*(6), 1087-1092.

Minsky, M. (1986). *The society of mind.* New York: Basic Books.

Moyson, F., & Manderick, B. (1988). The collective behavior of ants: An example of self-organization in massive parallelism. *Proceedings of the AAAI Spring Symposium on Parallel Models of Intelligence,* Stanford, CA.

Papadimitriou, C. H., & Steiglitz, K. (1982). *Combinatorial optimization: Algorithms and complexity.* Englewood Cliffs, NJ: Prentice-Hall.

Ramanujam, J., & Sadayappan, P. (1995). Mapping combinatorial optimization problems onto neural networks. *Information Sciences, 82,* 239-255.

Reynolds, C. (1987). Flocks, herds and schools: A distributed behavioral model. *Computer Graphics, 21*(4), 25-34.

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel distributed processing.* Cambridge, MA: MIT Press.

Shi, Y., & Eberhart, R. C. (1998). A modified particle swarm optimizer. *Proceedings of the IEEE International Conference on Evolutionary Computation,* Anchorage, AK.

Smith, K. (1995). Solving the generalized quadratic assignment problem using a self-organizing process. *Proceedings of the IEEE International Conference on Neural Networks 4* (pp. 1876-1879), Perth.

Smith, K. (1999). Neural networks for combinatorial optimization: A review of more than a decade of research. *INFORMS Journal on Computing, 11,* 15-34.

Smith, K., Krishnamoorthy, M., & Palaniswami, M. (1996). Neural versus traditional approaches to the location of interacting hub facilities. *Location Science, 4,* 155-171.

Stützle, T., & Dorigo, M. (1999). ACO algorithms for the traveling salesman problem. In K. Miettinen et al. (Eds.), *Evolutionary algorithms in engineering and computer science*. (pp. 163-183). Chichester: John Wiley & Sons.

Stützle, T., & Dorigo, M. (2002). A short convergence proof for a class of ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation, 6,* 358-365.

Taillard, D. (1994). Parallel tabu search techniques for the job shop scheduling problem. *ORSA Journal on Computing, 6,* 106-117.

Thinking Machines Corporation. (1986). *Introduction to data level parallelism.* Technical Report No. 86.14, Thinking Machines Corporation, Cambridge, MA.

Voss, S., Martello, S., Osman, I. H., & Roucairol, C. (Eds.). (1999). *Meta-heuristics—advances and trends in local search paradigms for optimization.* Dordrecht: Kluwer.

Watson, J.-P., Whitley, D. L., & Howe, A. E. (2003). A dynamic model of tabu search for the job-shop scheduling problem. *Proceedings of the 1st Multidisciplinary International Conference on Scheduling: Theory and Application* (MISTA 2003).

Wilson, E. O. (1975). *Sociobiology: The new synthesis.* Cambridge, MA: Belknap Press.

Wolfram, S. (1984). Universality and complexity in cellular automata. *Physica D, 10,* 1-35.

Wolfram, S. (2002). *A new kind of science.* Wolfram Media.

Wolpert, D. H., & Macready, W. G. (1995). *No Free Lunch theorems for search.* Technical Report No. SFI-TR-95-02-010, Santa-Fe Institute, USA.

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proceedings of the 6th International Congress of Genetrics 1* (pp. 356-366).

KEY TERMS

Emergent Behavior: A behavior that was not directly hardcoded into an algorithm.

Fitness Landscape: If one plots the fitness of all possible solutions to a problem, one obtains a "landscape" that an optimization algorithm will explore, usually to find a global optimum.

Global Search: Global search methods will implement ways to avoid getting trapped in local optima.

Heuristic: Non-exact method used to find a good solution, usually based on trial and error.

Local Search: Local search methods are usually deterministic. They can find a local optimum very quickly, but will get stuck there.

Multimodal: A problem with several local optima. Local search algorithms will get trapped in the first local optimum they find. If one is unlucky, the found local optimum is not the global optimum. Local search algorithms cannot be used on multimodal problems.

NP-Hard Problem: A problem for which no polynomial time solution has yet been found. This means that in order to find a global optimum, no other solution is known than to test all possible solutions.

Stigmergy: Basic inter-individual communication based on information left in the environment for others to use.

Unimodal: A problem with a single local optimum (that is *de facto* also the global optimum). Local search algorithms will find the global optimum of a unimodal problem.

Chapter IV
Evolutionary Algorithms

Pierre Collet
Université du Littoral Côte d'Opale, France

ABSTRACT

Evolutionary computation is an old field of computer science that started in the end of the 1960s nearly simultaneously in different parts of the world. Each paradigm has evolved separately, apparently without knowledge of what was happening elsewhere, until people finally got together and shared their experience. This resulted in strong trends that still survive, even though it is now possible to outline a generic structure for an evolutionary algorithm that is described in this chapter.

INTRODUCTION AND HISTORY

The development of evolutionary algorithms almost dates back to the dark ages of computers. To put back everything in perspective, Computer Science really started when John von Neumann designed the EDVAC (electronic discrete variable automatic computer) in 1945, but the first prototype was actually implemented in 1949 with Wilkes' EDSAC (electronic delay storage automatic calculator). Then, for a while, the only commercially available machines used valves and were therefore not that reliable (IBM 650 in 1953). A quantum leap was made when transistors became available around the 1960s, and finally, Integrated Circuits in 1964.

By that time, evolutionary computation had about ten independent beginnings in Australia, the United States, and Europe, starting in 1953, traced by David Fogel's excellent *Fossil Record* (Fogel, 1998): Alex Fraser had evolved binary strings using crossovers (Fraser, 1957), Friedberg had already thought of self-programming computers through mutations (Friedberg, 1958; Friedberg, Dunham, & North, 1958), and Friedman of how evolution could be digitally simulated (Friedman, 1959). However, the main evolutionary trends that survived are as follows:

- **Evolutionary Strategies:** By Rechenberg and Schwefel, best described in Rechenberg (1973) and Schwefel (1995).

- **Genetic Algorithms:** By Holland, later popularized by Goldberg on the U.S. East Coast (Michigan) (Holland, 1975; Goldberg, 1989).
- **Evolutionary Programming:** By Lawrence Fogel and later David Fogel on the U.S. West Coast (Fogel, Owens, & Walsh, 1966; Fogel, 1992).
- **Genetic Programming:** By Cramer (1985) and later developed by Koza (1992) (see Chapter V).

Evolutionary computation cannot, therefore, be seen as a recent development of computer science, or even classified as artificial intelligence, which is a different concept that also started back in the mid-1950s with John McCarthy and many others.

However, until the principles of evolutionary computation were clearly understood, these techniques necessitated a larger amount of computer power than was available until the beginning of the 1990s.

Thus although evolutionary computation really started in the late 1960s, it only came of age when computers had enough power to make them a technique competitive with other (posterior) stochastic optimization paradigms such as simulated annealing (Kirkpatrick, Gellat, & Vecchi, 1983) or tabu search (Glover, 1977, 1989, 1990) (see Chapter III).

SHORT PRESENTATION OF THE EVOLUTIONARY COMPUTATION PARADIGM

The general idea comes from the observation that animals and plants are very well adapted to their environment. Back in 1859, Charles Darwin came with an explanation for this called *natural selection,* which is now widely accepted (Darwin, 1859). The rationale is that *individuals* that are not well adapted to their environment do not survive long enough to reproduce, or have less chances to reproduce than other individuals of the same species that have acquired beneficial traits through *variation* during *reproduction.* Adaptation to the environment is also called *fitness*.

Artificial evolution grossly copies these natural mechanisms in order to optimize solutions to difficult problems. All optimization techniques based on Darwinian principles are *de facto* members of the evolutionary computation paradigm.

A UNIFIED EVOLUTIONARY ALGORITHM

Kenneth DeJong has been giving a GECCO tutorial on the unification of evolutionary algorithms for several years now and has come up with a recent book on the subject (DeJong, 2005). Indeed, the previously quoted currents (evolutionary strategies, genetic algorithms, evolutionary programming, genetic programming) all share the same principles copied from natural selection.

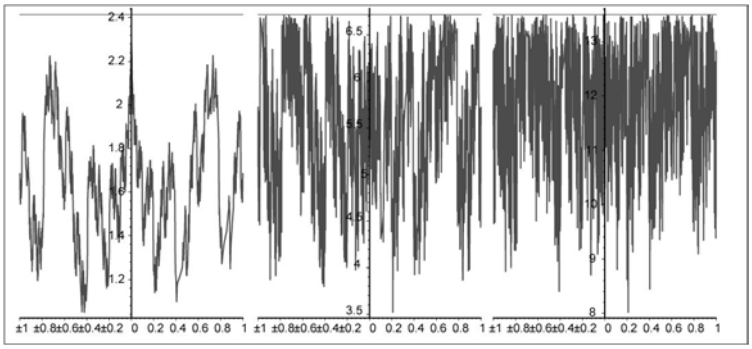
Rather than describing each algorithm, this chapter will describe a generic and complete version that can emulate virtually any paradigm, depending on chosen parameters.

Representation of Individuals

Due to the similarities between artificial evolution and natural evolution that was the source of its inspiration, a good part of the vocabulary was borrowed from biologists. In artificial evolution, a potential solution to a problem is called an *individual*.

Using a correct representation to implement individuals is a very essential step that is trivial for some kinds of problems and much less trivial for others. The American trend (genetic algorithms) advocates using a representation that is as generic as possible—for example, a bit string

Figure 1. Mandelbrot-Weierstrass test functions with increasing irregularities



(even to code real values). The German trend (evolutionary strategies) that was designed to optimize continuous problems advocates using real variables.

Although using bitstrings makes sense for combinatorial problems or for theoretical studies, representing real values with bits, while feasible, has many drawbacks (Hinterding, Gielewski, & Peachey, 1995). It seems much more reasonable to use an appropriate representation tailored to the problem at hand.

If one tries to optimize a recipe for pancakes that uses flour, sugar, baking powder, and eggs, a reasonable encoding for an individual can be four "genes":

```
(float cpFlour, float tbspSugar, float
tspBakingPowder, int nEggs)
```

Note that this is the *genotype* of an individual. The *phenotype* will be the resulting pancake. This example makes it easy to understand that individuals are not evaluated on their genotype (ingredients) but on their phenotype (pancake). In this case, the fitness function will consist of measuring the width of the smile of the person who tastes the pancake.

In many problems, the relationship between genotype and phenotype is not so simple, and there can be intercorrelation between genes. (There is not a single gene coding for blue eyes.) The biology term to describe this correlation *epistasis*.

Choosing the good representation for the genotype is important, because it materializes the search space in which the best solution is sought. Note that evolutionary algorithms are not limited to bits, integers, or real numbers. Genetic programming uses binary trees, or a linear representation for a tree, or a grammar, or even stranger concepts (Cartesian GP—Miller, 2000)

To each his own. The conclusion is that common sense should prevail: one should use the representation that is best suited to the problem at hand.

Evaluation (or Fitness) Function

As for individual representation, the fitness function is problem dependent. The fitness function guides the evolution. It usually implements or simulates the problem to be solved, and should be able to rate the phenotype of the individuals proposed by the evolutionary engine. Most fitness functions requiring evolutionary algorithms are multimodal (with several local optima). The fitness function determines a *landscape* made of hills and valleys that may be more or less erratic (cf. Figure 1).

If an airfoil profile is to be optimized, an individual will encode an airfoil, and the fitness function will compute the Navier-Stokes equations that will yield the lift/drag ratio of the airfoil under precise conditions. The aim of the evolutionary engine will be, for instance, to maximize this ratio.

If the problem is a timetabling problem (constraints satisfaction problem), individuals will encode a timetable, and the fitness function will usually add penalties whenever a soft or a hard constraint is violated. The aim of the evolutionary engine will therefore be to minimize the penalties (the selection operators need to be reversed).

Fitness functions may be very diverse. Some may take hours to compute (Navier-Stokes equations), while others may take only milliseconds (checking violated constraints in a timetable). In the first case, 99% of the CPU time will be spent in the fitness function, so it is of the utmost importance to use a refined evolutionary engine so as to minimize the number of evaluations.

In the second case, which may use millions of millisecond evaluations, the hot-spot may be the evolutionary engine and not the fitness function! In such a case, it is usually best to keep genetic operators as simple (and fast) as possible, and let artificial evolution do its job. What is the point of using intelligent operators to reduce the number of evaluations if it is faster to find equivalent results by letting artificial evolution operate with simpler but much quicker operators (Collet, Louchet, & Lutton, 2002).

Finally, one last point to take into account is that some selection schemes (roulette wheel, for instance) require that the fitness function return always positive values.

Individual Initialization

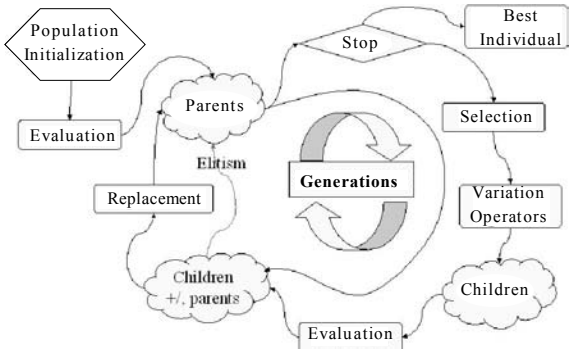
Initialization is another very important step. Standard evolutionary algorithms advocate starting with a population of random individuals, in order to sample the fitness landscape as uniformly as possible. One important reason for this is that "helping" the evolutionary algorithm by initializing individuals with "good" values may actually prevent it from finding very good solutions if these are unconventional: if one wants to optimize boat propelling systems by initializing all individuals of the population with what seems the most reasonable—for example, paddle wheels—the evolutionary algorithm will only optimize paddle wheels. If, by pure chance, an individual is mutated into a very coarse propeller shape, it will perform so poorly compared to even the worst paddle wheels that it will not be selected for reproduction, and will be immediately discarded from the population.

Some individuals may be evidently bad or even nonfeasible. In this case (and if the fitness function takes a long time to compute), removing them will speed up evolution; however, one should be very cautious and very conservative with killing of "bad" individuals: history is full of examples where progress was slowed down because some really good ideas had been rejected by people who thought they would never work (flying without flapping wings, wave/particle duality, Darwinism, etc.).

In order to start the algorithm, the initialization function is called repeatedly to create new individuals until the required initial population size is reached.

Each individual must then be evaluated by the fitness function in order to begin the evolutionary loop (it is necessary to know the fitness of all individuals in order to choose the parents that will create the first generation). If, by pure

Figure 2. Generic evolutionary algorithm flowchart



chance, a stopping criterion is met by an individual (error below 1%, for instance), the evaluation can stop at this point and yield back the good individual.

A Generic Evolutionary Loop (cf. Figure 2)

At this point, a certain number of necessary parameters (discussed below) must have been chosen, among which the number of children per generation (which may go from 1 (steady state replacement) to the population size (generational replacement) to any number n.

While (Stopping criterion not met):

1. While (NbChildren<NbChildrenPerGeneration):
   a. Select a variation operator (following Darwin's vocabulary), usually either unary or n-ary. The n-ary variation operator is often called with a 80-100% probability and is also often a binary crossover, yielding two children. The unary operator (called with a 20-0% probability is a cloning operator (which also preserves the fitness value without needing to recalculate it in static environment systems).
   b. Pick up the correct number of parents, using an appropriate *selection operator*. To the opposite of a *replacement operator* (cf. below), picked up individuals are put back in the parents pool and can be selected more than once.
   c. Call the variation operator, thereby creating one or several children (generally a number of children equal to the arity of the variation operator).
   d. Call a mutation operator on created children with a p probability (usually 100%). Please note that calling the mutation operator with a 100% probability does not mean that all children are mutated: mutation operators usually go through all the genes of the individual, and mutate each gene with a q probability. If the genotype of an individual contains 10 genes, and $q = 0.01$, then if this operator is called on 100% of the children ($p = 1$), in average, one children out of ten will undergo a mutation. Authors are generally not clear on the p and q values they use, making it difficult to reproduce their results.
   e. Variation operators can be followed by a *validation operator* that makes sure that newly created children are valid. Invalid individuals can be either deleted (in which case another individual needs to be created), "repaired," or given a very low fitness without even being evaluated by the fitness function. This last method is very interesting in constraints satisfaction problems (such as timetables) because it is very quick: in very constrained problems, one can spend a

lot of time finding 100 individuals that do not violate a single hard constraint. It is also very fast because it does not call the evaluation function to give a fitness to the child. In problems with fast fitness functions, it is often more efficient to give a bad mark and let artificial evolution deal with the problem.

2. Call the evaluation function on all children who do not already have a fitness. Children may already have a fitness if they are unmutated clones or if they were given a bad mark by the *validation operator*.
3. The algorithm now has to deal with two populations: a population of already evaluated parents and a population of evaluated children. In algorithms with a constant population size (the overwhelming majority), the number of individuals now needs to be reduced back to the original population size to constitute the next generation. There remains however a last step before replacement: Holland's original Genetic Algorithm uses a generational replacement, meaning that the population of children brutally replaces the population of parents to create the next generation. In this process, it is possible to lose a very good solution if, for instance, none of the children did better than the best of their parents. The solution that was found to overcome this problem was *elitism*. Generational GAs with elitism simply take the best parent and put it in the new generation. This method is called *strong elitism* since it is not intelligent. A more subtle form of elitism is *weak elitism,* which moves into the next generation the best individual from both the parent and children populations. One must be aware that elitism may lead to premature convergence, as the best found individual will always make it to the next generation. If it is much better than the other individuals, it will be often selected as a parent, and its genes will spread into the population along with the generations, preventing other potentially good solutions to develop (cf. paddle wheels vs. propellers). Elitism should therefore be used sparingly, especially if premature convergence occurs. If it is not used, it is well advised to keep track of the best found individual, so as to be sure that it does not get lost.
   Finally, usually, elitism only concerns the best individual. However, some paradigms such as the Parisian approach may need to use elitism on 40 or more percent of the population (Collet, Lutton, Raynal, & Schoenauer, 2000; Louchet, 2001).
4. The final step of the evolutionary loop is the replacement operator. If elitism was used, one or several individuals are already part of the new generation. The replacement operator will pick up other individuals among parents and children until the new generation is complete (e.g., for constant size population algorithm) until a number of individuals equal to the original population size are selected.
   The difference between the *selection* and the *replacement* operators is that the latter cannot choose individuals more than once. Any selection method can be used, as for parents selection (cf. the following).

What is very nice with evolutionary algorithms is that they parallelize extremely well: most of the time, evaluation is the big CPU consumer, so once all children are created (which usually takes very little time), one can distribute the evaluation of the population over a network of machines. If there are 10 machines and 100 children per generation, each

machine can be given 10 children to evaluate. Usually the speed-up ratio is nearly linear with the number of machines because in most evolutionary algorithms (but genetic programming), evaluation time is identical for all individuals.

VARIATION OPERATORS

In *The Origin of Species,* Charles Darwin suggested that individuals evolved thanks to the fact that individuals inherited traits from their parents with variations. Biological and physical constraints do not apply in evolutionary computation, meaning that virtually any kind of variation operators can be imagined, from unary operators (mutation), binary operators (crossover), to n-ary operators. Orgies are also possible, where all individuals of a population can share their genes to create a new individual (Mühlenbein & Paass, 1996).

The most traditional way to create children is the one described above in the evolutionary loop, even though a more generic way is to have a number of variation operators, each associated with a probability to be applied to the parent population until the right number of children is generated (Keijzer, Merelo, Romero, & Schonauer, 2002).

Crossover: A (Usually) Binary Variation Operator

The way two (or n) parents' genotypes are mixed in order to create one or two children is highly dependent on the problem being solved. Genetic algorithms usually use a bitstring representation, while evolutionary strategies use a vector of real values and genetic programming uses a tree representation.

The guidelines for a good crossover operator are therefore quite difficult to establish, although it is important to bear in mind that

Figure 3. Multi-point crossover between parents p1 and p2, resulting in children c1 and c2

crossover is considered as an exploitation operator (that can use hints from a possibly deterministic method in meta-heuristic "memetic" algorithms (Hart, Krasnogor, & Smith, 2005)).

If the genotype contains real values, some problems may benefit from a *barycentric* crossover where genes of the resulting child are the mean of the parents' genes.

A more standard way of creating children is to use a single or multi-point combinatorial crossover: two parents are selected, as well as one or several crossover points (*loci* in biological parlance). Two children are created by swapping the parent's genotype between each point (cf. Figure 3).

If individuals are made of n genes, an $n-1$ points crossover is called a uniform crossover (Syswerda, 1987). However, genes are then usually swapped randomly (with a probability $p = 0.5$ rather than 1).

One should avoid uniform crossover in problems that show a high degree of epistasis because interrelated genes (called building blocks) will be disrupted. On the other hand, it has been shown that a single point crossover could be problematic, as it may not allow creation of some combinations of features encoded on chromosomes. In cases where this problem may occur, a two point crossover is usually preferred (Spears and De Jong, 1990).

Mutation

Mutation also depends on the problem to be solved. As an exploration operator, it should

ideally be *ergodic* in a strong sense, meaning that the probability of reaching any point of the search space from the current position through a single mutation should be greater than 0.

If the construction of such a mutation operator is not feasible, it should be ergodic in a weaker sense, meaning that it should be possible to reach any point of the search space in a *finite* number of mutations.

On a bitstring genome, mutation is simple: it merely consists in flipping a chosen bit. On a real genome, mutations can be done in several ways, the simplest being to add some Gaussian noise to a selected real value.

The most evolved (and efficient) way of mutating a real value is *self-adaptive mutation,* described in evolutionary strategies (Schwefel, 1995; Beyer, 1995; Bäck, 1995; Bäck, Hammel, & Schwefel, 1997). The idea (inspired from nature) is that to each real value should be associated a variance value σ that is subject to mutation and recombination, just as are the other genes of the genome.

This self-adaptive mutation is comparable to what happens with repair enzymes and mutator genes that are coded onto the DNA, thus providing partial control by the DNA of its own mutation probability.

In the first generation, σ values are initialized with random values between 0 and 0.5. If a mutation occurs on a gene, one starts by updating the σ value associated with this gene along a log-normal distribution—that is, by multiplying it by $\exp(\frac{1}{\sqrt{n}}G)$ —where n is the number of genes in the genome and G is a Gaussian normally distributed random value with variance 1 and mean 0. One then adds to the real gene a Gaussian value multiplied by the updated σ value associated to the gene.

Self-adaptive mutation uses a bit of CPU time and resource, but allows one to achieve comparable results in fewer evaluations (Collet et al., 2002).

Selection and Replacement Operators

It has often been said that the force driving biological evolution was *natural selection.* In evolutionary computation, selection algorithms are also extremely important, as they can lead to premature or very slow convergence, depending on selection pressure.

Selection occurs at two stages, when choosing parents for breeding and when choosing survivors for the next generation. In this chapter, *selection* chooses parents, and *replacement* chooses survivors. The main difference between *selection* and *replacement* is that the first operator allows a same individual to be chosen several times, while the latter removes the chosen individual from the pool of candidates.

In his original description of genetic algorithms, Holland chose to use *Roulette Wheel* as a selection scheme for a theoretical pupose (Holland, 1975). Unfortunately, in practical problems, this is probably the worst choice to be made. This operator selects individuals proportionately to their fitness, which has several important drawbacks:

1. Selection pressure totally depends on the fitness landscape, which is usually unknown. It is not translation invariant: in a population of 10 individuals, if the best has a fitness of 11 and the worst a fitness of 1, the probability for the best individual to be chosen is 16.6% and 1.5% for the worst. If one adds 100 to all fitness values, the best and worst individuals have nearly identical probabilities to be chosen (10.4% and 9.5%)!
2. Things can be partially improved thanks to *linear scaling* of fitness values, or a *sigma truncation,* but to the cost of increased complexity (additional parameters

to adjust). Roulette wheel is CPU-consuming, as the simulation of the spin of a weighted roulette requires $O(n)$ operations that need to be performed n times, resulting in an $O(n^2)$ algorithm.

3. Roulette requires the sum of the fitness values of all the individuals. This is problematic if the evolutionary computation is distributed over several machines. (This is the case for all other selection algorithms but Tournament selection.)
4. The fitness function needs to yield positive values (which is not really problematic, but due to the fact that Roulette is not translation invariant, shifting the values so that they are all positive has consequences).

Other selection methods have been devised in order to mainly circumvent problem number 1:

- **Ranking:** Selection is based on rank, not fitness. One therefore needs to sort the population, leading to an $O(n\log n)$ complexity. Problem 1 is solved, but the others remain (Baker, 1985).
- **Stochastic Universal Sampling:** Individuals are assigned slots of a weighted roulette wheel, as for the Roulette selection. n markers are then placed equally around the wheel and the wheel is spun once. The complexity of this algorithm is also $O(n\log n)$ and it requires the sum of the fitness values of all the individuals (Baker, 1987).
- **Selection in Genitor:** Genitor is an evolutionary paradigm that is of the *Steady State* kind, in which only one individual is created per "generation." The population is initially ranked, after which each new child is inserted at its place and the worst individual of the population is discarded. This requires $O(\log)$ steps that need to be repeated n times in order to simulate the creation of a whole population, so the complexity of the algorithm is $O(n\log n)$. The Genitor selection and replacement scheme may lead to premature convergence, which is why large population sizes are suggested (Whitley, 1989).
- **Truncation Selection:** This is the selection method used by breeders. Only the T best individuals are considered, and all of them have the same selection probability (random selection among T individuals). The population needs to be sorted first, so complexity is $O(n\log n)$. Bad individuals (below threshold T) cannot be selected, so loss of diversity can be important (Mühlenbein & Schlierkamp-Voosen, 1993).
- **Deterministic Selection:** Only the n best individuals are selected. This method requires sorting the individuals. Loss of diversity is important (as for Truncation selection), and this selection method may lead to premature convergence.
- **Random:** Quick, but no selection pressure.

Next, there is n-ary *tournament* selection (Brindle, 1981; Blickle & Thiele, 1995). Unless there is a good reason for using any other method, tournament selection is most certainly the best of all. Binary tournament consists of picking two individuals at random and comparing their fitness. The individual with the highest fitness wins the tournament and is selected. Selection pressure can be increased by organizing a tournament between three or more individuals. In contrast, if premature convergence occurs with a binary tournament, it is possible to decrease selection pressure by using a *stochastic tournament* that uses a variable p. A stochastic tournament is a binary tournament where the best of the two individu-

als is chosen with a probability p. If 0.5, then this is equivalent to a random selection. If p = 1, the stochastic tournament is not stochastic anymore and becomes equivalent to a binary tournament.

Therefore, in comparison with Roulette wheel selection:

1. Selection pressure does not depend on the fitness landscape and can be very finely adjusted. This is a very important parameter to tune in a genetic algorithm: all other parameters being equal, if the algorithm does not converge rapidly enough (the best fitness curve was still increasing when a stopping criterion was met), one can increase the selection pressure *ad libitum* by increasing the number of participants in the tournament (n-ary tournament). Alternately, if premature convergence occurs (plateau on the best fitness many generations before the stopping criterion is met), it is possible to decrease pressure by reducing the number of participants. If it is already as low as two, one can switch to stochastic tournament which allows the decrease of selection pressure further, possibly down to a purely random selection.

   Loss of diversity represents the proportion of individuals of a population that are not selected. Here are some computed values for different tournament sizes: 25% loss for size 2, 40% loss for size 3, 47% for size 4, 53% for size 5, 60% size 7, 70% size 10, and 80% for size 20.
2. Tournament complexity is simply $O(n)$, as one only needs n tournaments to create a population of n individuals. This method is therefore one of the fastest.
3. Tournament is the method of choice for parallel evolutionary algorithms, as it does not require any global knowledge of individuals' fitness. Tournament selection can be implemented locally on parallel machines, with pairwise or s -wise communication between different processors being the only requirement (Mühlenbein, 1989; Harvey, 1993).
4. The fitness function needs not yield only positive values, and no scaling or post-processing of any kind is needed.

A good study on selection schemes can be found in Blickle and Thiele (1997) and Goldberg and Deb (1991).

STOPPING CRITERIA

Most users choose to stop evolution after n generations and use this number as an evaluation of CPU-consumption. This only makes sense for generational replacement algorithms, where the number of individuals created per generation is equivalent to the population size.

Unfortunately, this is not the case for evolutionary strategies (see the following paragraphs) that use a $(\mu + \lambda)$ replacement scheme, where the number of created children is not correlated to the population size. A much better metric is therefore the number of *evaluations* and not the number of *generations*.

Runtime can also be a stopping criterion (stop after one hour). When such fixed criteria (duration or number of evaluations) are used, parameters of the algorithm should be tweaked so that the algorithm converges at the end of the run, and not before or after. If the algorithm converges before the stopping criterion is met (plateau on the best fitness individual), one can either reduce selection pressure or increase population size and do the opposite if the algorithm had not converged yet.

This triggers another idea: why not use fitness convergence as stopping criterion? One

way of doing this is to stop if a plateau is too long: if F_k is the fitness of the best individual at generation k and F_c is the current best fitness, one can stop if $F_c - (\sum_1^p F_k)/p \leq \varepsilon$, with p the length of the plateau in number of generations.

Generally, a population that has converged is stuck in a local optimum and does not evolve anymore (which is why it is very important not to let it converge and implement diversity preserving schemes to this effect). If a metrics is available that can determine the distance between two individuals' genotype, one can use loss of diversity over the population as a stopping criterion.

Finally, the ultimate stopping criterion is fitness value. If the problem is to find an individual with a fitness beyond 1,000, the algorithm can stop once this value is met (possibly on the first generation if one is very lucky!).

PARAMETERS

The full list of parameters for an evolutionary algorithm is the following:

- **Population Size/Number of Generations:** These two parameters go together: for a same number of evaluations, one can use a small population and evolve it for a large number of generations, or use a large population and evolve it for a smaller number of generations. Common sense says that using a larger population will preserve diversity and help fight against premature convergence.
- **Crossover and Mutation Probabilities:** Usage of unary or n-ary variation operators depends on people and paradigms (evolution strategies use nearly exclusively mutations, and genetic programming nearly exclusively crossovers, for instance). In fact, this is very problem dependent. Without prior experience and a good reason for putting forward crossover or mutation, the most standard choice is to create offspring thanks to a binary crossover called with a probability of 80 to 90%, followed by a mutation function called on each child, with a mutation rate that will change a gene once in a while (one mutation for every 10 children is a good starting base). Too many mutations (exploration operator) will lead to non-converging algorithms. A high low/mutation rate is therefore usually associated with a strong/weak selection pressure, respectively.
- **Number of Children per Generation:** If the population size is n, many people create n children, although there is no real reason behind this choice, other than the fact that this is how Holland's genetic algorithms were working. Evolutionary strategies use a $(\mu + \lambda)$ or (μ, λ) replacement scheme. In the first strategy, the replacement operator picks n individuals for the new generation from a pool of individuals made of μ parents and λ children, while the second strategy only picks individuals of the new generation in the λ children population (with $\lambda \geq \nu$). Finally, steady-state algorithms create only one child per generation! Choosing the number of children per generation generally depends on how fast one wants the algorithm to converge. Allowing parents to compete with children is a powerful form of elitism that can be countered by creating many children per generation.

CONCLUSION

The different evolutionary paradigms have not been described because it was not possible to do so within a single chapter. Instead, a generic

algorithm was presented that can emulate any of the different historical paradigms, should anyone wish to do so.

These paradigms correspond to different trends. It seems however that parameter choice should be made in order to solve a particular problem, rather than to follow a particular trend. Evolutionary computation has now come of age, with very impressive achievements, so it is high time that this domain be unified in a pragmatic way. Books such as DeJong (2005) are certainly going in the right direction.

REFERENCES

Bäck, T. (1995). *Evolutionary algorithms in theory and practice.* New York: Oxford University Press.

Baker, J. E. (1985). Adaptive selection methods for genetic algorithms. *Proceedings of the International Conference on Genetic Algorithms and Their Applications* (pp. 100-111). Mahwah, NJ: Lawrence Erlbaum Associates.

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette (Ed.), *Proceedings of the 2nd International Conference on Genetic Algorithms* (pp. 14-21). San Francisco: Morgan Kaufmann.

Blickle, T., & Thiele, L. (1995). A mathematical analysis of tournament selection. In L. J. Eshelman (Ed.), *Proceedings of the 6th International Conference on Genetic Algorithms* (pp. 9-16). San Francisco: Morgan Kaufmann.

Blickle, T., & Thiele, L. (1997). A comparison of selection schemes used in genetic algorithms. *Evolutionary Computation, 11,* 361-394.

Brindle, A. (1981). *Genetic algorithms in search, optimization.* Technical Report No. TR81-2, Department of Computer Science, University of Alberta, Canada.

Collet, P., Louchet, J., & Lutton, E. (2002). Issues on the optimization of evolutionary algorithms code. In D. B. Fogel & M. A. El-Sharkawi (Eds.), *Proceedings of the 2002 Congress on Evolutionary Computation* (pp. 1103-1108). Piscataway, NJ: IEEE Press.

Cramer, N. L. (1985). A representation for the adaptive generation of simple sequential programs. In J. J. Greffenstette (Ed.), *Proceedings of the International Conference on Genetic Algorithms and Their Applications* (pp.183-187). Carnegie Mellon University.

Darwin, C. (1859). *On the origin of species by means of natural selection or the preservation of favored races in the struggle for life.* London: John Murray.

DeJong, K. (2005). *Evolutionary computation: A unified approach.* Cambridge, MA: MIT Press.

Fogel, D. B. (1992). An analysis of evolutionary programming. In D. B. Fogel & W. Atmar (Eds.), *Proceedings of the 1st Annual Conference on Evolutionary Programming* (pp. 43-51).

Fogel, D. B.(1998). *Evolutionary computation: The fossil record.* Wiley-IEEE Press.

Fogel, L. J., Owens, A. J., & Walsh, M. J.(1966). *Artificial intelligence through simulated evolution.* New York: John Wiley & Sons.

Fraser, A. S. (1957). Simulation of genetic systems by automatic digital computers. *Australian Journal of Biological Sciences, 10,* 484-491.

Friedberg, R., Dunham, B., & North, J. (1958). A learning machine: Part II. *IBM Research Journal, 3*(3), 282-287.

Friedman, G. (1959). Digital simulation of an evolutionary process. *General Systems Yearbook, 4,* 171-184.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision Science, 8,* 156-166.

Glover, F. (1989). Tabu search—part I. *ORSA Journal on Computing, 1*(3), 190-206.

Glover, F. (1990). Tabu search—part II. *ORSA Journal on Computing, 2*(3), 4-32.

Goldberg, D., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. *Foundations of Genetic Algorithms,* 416-421.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Boston: Addison-Wesley.

Hart, W. E., Krasnogor, N., & Smith, J. E. (2005). *Recent advances in memetic algorithms*. Springer.

Harvey, I. (1993). Evolutionary robotics and saga: The case for hill crawling and tournament selection. *Artificial Life III, Santa Fe Institute Studies in the Sciences of Complexity, XVI,* 299-326.

Holland, J. H. (1975). *Adaptation in natural and artificial systems.* Ann Arbor: University of Michigan Press.

Keijzer, M., Merelo, J.J., Romero, G., & Schoenauer, M. (2002). Evolving objects: A general purpose evolutionary computation library. In P. Collet, E. Lutton, M. Schoenauer, C. Fonlupt, & J.-K. Hao (Eds.), *Artificial evolution '01* (pp. 229-241). Berlin: Springer Verlag (LNCS 2310).

Kirkpatrick, S., Gellat, C. D., & Vecchi, M. P.(1983). Optimization by simulated annealing. *Science, 220*(4598), 671-680.

Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural evolution.* Cambridge, MA: MIT Press.

Louchet, J. (2001). Using an individual evolution strategy for stereovision. *Genetic Programming and Evolvable Machines, 2*(2), 101-109.

Miller, P. T. (2000). Cartesian genetic programming. In R. Poli, W. Banzhaf, W. B. Langdon, J. F. Miller, P. Nordin, & T. C. Fogarty (Eds.), *Proceedings of EUROGP'00* (pp. 121-131). Edinburgh: Springer.

Mühlenbein, H., & Paass, G. (1996). From recombination of genes to the estimation of distributions. *Parallel Problem Solving from Nature, 1411,* 178-187.

Mühlenbein, H. (1989). Parallel genetic algorithms, population genetics and combinatorial optimization. *Proceedings of the 3rd International Conference on Genetic Algorithms* (pp. 416-421).

Mühlenbein, H., & Schlierkamp-Voosen, D. (1993). The science of breeding and its application to the breeder genetic algorithm (BGA). *Evolutionary Computation, 1*(4), 335-360.

Rechenberg, I. (1973). *Evolutionstrategie: Optimierung technisher systeme nach prinzipien des biologischen evolution*. Stuttgart: Fromman-Hozlboog Verlag.

Schwefel, H.-P. (1995). *Numerical optimization of computer models* (2nd ed.). New-York: John Wiley & Sons.

Spears, W. M., & De Jong, K. A. (1990). An analysis of multi-point crossover. In G. J. E. Rawlins (Ed.), *Proceedings of the Foundations of Genetic Algorithms Workshop.* San Mateo, CA: Morgan Kaufmann.

Syswerda, G. (1987). Uniform crossover in genetic algorithms. In J. Schaffer (Ed.), *Pro-*

ceedings of the 3rd International Conference on Genetic Algorithms (pp. 2-9). San Mateo: Morgan Kaufmann.

Whitley, D. (1989). The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In J.D. Schaffer (Ed.), *Proceedings of the 3rd International Conference on Genetic Algorithms* (pp. 116-121). San Francisco: Morgan Kaufmann.

KEY TERMS

Crossover: Function capable of recombining several potential solutions (parents) into new solutions (children). Crossover is usually seen as an exploitation operator, equivalent to a local search.

Elitism: Elitist algorithms make sure that the best individuals are not discarded, by transferring them directly into the next generation.

Epistasis: Amount of correlation between different genes (or parameters).

Ergodicity: A function is ergodic if the probability to jump from any point *a* of the search space to any other point *b* is not null. A mutation operator randomly flipping a single bit of a bitstring is therefore not strictly ergodic. However, it is ergodic in a loose sense, in that it is possible to obtain any other individual in a finite number of mutations.

Genotype: Set of parameters allowing representation of a solution to a problem.

Individual: Potential solution to a problem.

Mutation: Function that can possibly alter the genotype of an individual. Mutation is usually seen as an exploration operator, allowing the escape from local optima. A good mutation function should be ergodic.

N-ary Operator: Operator needing *n* operands.

Phenotype: Expression of the genotype. The fitness of an individual is evaluated on its phenotype, not on its genotype. In some problems there is no simple correlation between phenotype and genotype. For instance, there is not a single gene coding for blue eyes. Blue eyes is a phenotype.

Replacement: Refers to the operator that will create a new generation out of a pool of children and/or parents. Only one instance of an individual can make it to the new generation.

Representation: Structure of the genome of an individual (array of bits, integers, doubles, etc.).

Selection: An algorithm that picks up an individual in a population with a bias (usually towards good individuals). Selection usually refers to choosing parents for mating. Selection is different from replacement, as an individual can be chosen several times for reproduction.

Unary Operator: Operator needing only one operand.

Chapter V
Genetic Programming

Pierre Collet
Université du Littoral Côte d'Opale, France

ABSTRACT

The aim of genetic programming is to evolve programs or functions (symbolic regression) thanks to artificial evolution. This technique is now mature and can routinely yield results on par with (or even better than) human intelligence. This chapter sums up the basics of genetic programming and outlines the main subtleties one should be aware of in order to obtain good results.

INTRODUCTION

Genetic programming (GP) is still rather unknown, even though it has recently obtained spectacular results: John Koza showed in his latest book (Koza et al., 2003) that genetic programming can routinely produce solutions that are competitive with human intelligence, without requiring one to be an expert in the domain of the problem to be solved.

A Bit of History

The idea of evolving computer programs dates back to the dawn of computing. Back in 1958, Friedberg made several attempts to have a computer program itself (Friedberg, 1958; Friedberg, Dunham, & North, 1958) using what would now be called mutations. He started with a "population" of random programs, and modified the contents stochastically, trying to improve the results.

Later on, Smith (1980), who was working on learning classifier systems, introduced small programs in the rules he was evolving. However, the modern vision of genetic programming starts with a small but seminal paper by Cramer (1985), who uses a tree-like variable size structure to represent a program. Programs are not written in LISP (as suggested by Koza later on) but in TB (a tree version of the JB language). Along with mutation, Cramer also uses a standard subtree crossover, introduces as well a mono-parental crossover, and insists on the necessity to create closed genetic operators. Above all, he evolves his population of programs with an evolutionary engine.

All the seeds were therefore present for the domain to grow, but as for Manderick and Moyson and Ant Colony Optimization (cf. Chap-

ter III), Cramer somehow failed to promote his work enough, and nothing major happened in this domain for several years. In fact, another major problem was that computers of this era were not powerful enough to obtain really good results with GP.

Genetic Programming à la Koza

By the beginning of the 1990s, genetic programming made its comeback thanks to Koza (1989, 1992, 1994; Koza et al., 1999, 2003) who put in a lot of energy (and computer power) to develop the ideas introduced by Cramer and his predecessors.

Genetic programming is nothing else than standard evolutionary techniques (described in Chapter IV) applied to individuals that implement programs. Standard evolutionary algorithms evolve potential solutions to a problem to be optimized. Most of the time, EA individuals are made of a list of parameters that are passed over to a "fitness function" used to evaluate the individual.

In genetic programming, an individual is a program, or more often a function. The main (and only?) difference with EAs is that GP *executes the individuals to evaluate them.* Another difference is that in most of the cases, GP uses variable length individuals where standard EAs use fixed size individuals.

This chapter presents standard genetic programming *à la* Koza, including hints, suggestions, and pointers to state-of-the-art papers that will hopefully allow newcomers to obtain good results with this delicate technique.

STANDARD GENETIC PROGRAMMING

Representation of an Individual

As is the case in EAs, using the good representation for a particular problem is quite essential, because the chosen representation more or less determines the search space in which the individuals will evolve: it will be very difficult to obtain an iterative program with a representation that does not allow loops or recursive calls. On the contrary, if everything is allowed, the search space may be so large that finding compiling programs that stop correctly or simply do not hang will be already very difficult.

The individual representation described below corresponds to the most common one¾that is, the representation that Koza used for the development of what is now standard genetic programming. In order to reduce the search space and find an individual structure adapted to the representation of a program, Koza naturally chose a functional language for several reasons:

- Syntactically speaking, with a purely functional representation, there is no need to define a grammar recognizing valid programs, provided the set of functions is closed.
- Functional languages limit side effects, which, as a side effect, minimizes bug occurrence.
- Above all, a functional program can very easily be implemented as a tree: nodes are operators that have as many children that they need operands and that return to their parent the result of their evaluation.

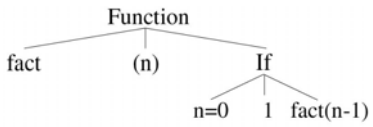
For instance, a function calculating a factorial:

```
Function fact(n) {
    If n=0 return 1
    Else return fact(n-1)
}
```

can be simply represented by the tree in Figure 1.

Moreover, of all different possible representations, a tree structure can naturally implement variable size individuals on which cross-

Figure 1. Representation of a function using a tree structure



over operators can be very easily applied: one only needs to swap two subtrees (cf. section below on genetic operators).

Of all available functional languages, LISP is certainly the most usable. It was quite popular at the end of the 1980s, and some computers were developed around its specificities. LISP was therefore Koza's choice to implement GP individuals, and he used a Texas Instrument LISP machine to speed up evaluation of the evolved programs.

Whether one wants to evolve a true program or simply a function approximating a curve, standard genetic programming represents individuals with a tree structure. Other interesting representations have emerged since then (linear GP, Grammatical Evolution, Cartesian GP, Stack-based GP, etc.), but each would need a complete chapter for a correct description.

Terminals and Functions

In a tree representation, functions *(or operators) are nodes and* terminals *are leaves.*

Terminals Set

Terminals are constants or variables. Independent variables (x, for instance) are typically inputs of the program to be optimized. Their values are passed over as parameters to the individual for evaluation. In robotics, inputs can come from sensors.

Constants are called ERCs (for *ephemeral random floating-point constants*). The standard way of using them is to create them when needed, by choosing randomly a value inside [–1.0, 1.0]. In standard GP, the values of ERCs are never modified.

Functions Set

In order to minimize the search space, one must choose a minimal set of functions or operators, allowing genetic programming to find a solution to the problem to be solved (approximating $1/x$ with only {+, –, *} may prove difficult, for example).

Wang and Soule (2004) showed that some functions could be equivalent for a given problem. If, for instance, similar results are obtained with the set of functions {+, –, *, /, sin} and {+, –, *, /, cos}, one can conclude that sine and cosine functions are equivalent. Then, further experiments suggest that using a functions set containing several equivalent functions (such as {+, –, *, /, sin, cos}) yields worse results than either of the two previous sets, probably because the added function does not bring an advantage that compensates the widened search space.

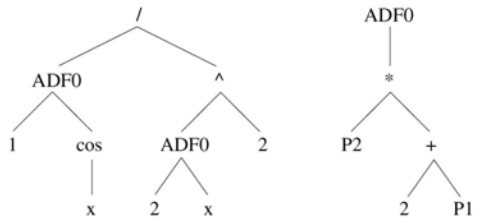
If functions of different arity are allowed, genetic operators such as crossover and mutation must take that into account in order to always provide syntactically correct functions (closure).

Automatically Defined Functions (ADFs)

Restricting individuals to a simple tree would be quite limiting, as there would be no way to implement new functions: if the problem to be solved necessitated four calls to the same subfunction, genetic programming would have to evolve four identical functions!

Koza (1994) has therefore introduced what he calls ADFs (*automatically defined functions*) that make it possible to evolve one or more independent functions that can be called

Figure 2. A single individual containing both a main function tree and an Automatically Defined Function ADF0. P1 and P2 are the parameters (operands) of the ADF function.



by the main program. If the problem to be solved suggests that it could use two different subroutines, individuals should implement two additional trees (called ADF0 and ADF1, for example). Figure 2 shows an individual with a single ADF.

If the original functions set was {+, –, *, /, sin}, one will simply add to it ADF0 and ADF1 so that the main program can call these new functions. The functions set of the ADFs is either the same or a different one, depending on what one thinks the ADF might need, but in any case would not include ADF0 or ADF1, to prevent recursivity that genetic programming would not handle well.

The set of terminals of an ADF function must also contain the arguments needed by the function (P1 and P2 for instance). If ADF0 (P_1, P_2) = $P_2(2 + P_1)$, the individual of Figure 2 will code:

$$\frac{\text{ADF0}(1, \cos x)}{\text{ADF0}(2, x)^2}, \textit{ that is}:$$

$$\frac{\cos x(2+1)}{(x(2+2))^2}.$$

Fitness Function

The difference between evolutionary algorithms and genetic programming is that genetic programming executes individuals to evaluate them. The fitness of an individual is therefore determined as the result of its execution. Here are some examples:

- If GP is used to evolve artificial ants seeking food (classical Santa-Fe trail benchmark (Koza, 1994)), the fitness will be represented by the number of food spots found by an ant (being maximized).
- If one seeks an Iterated Functions System approximating a given target (Collet, Lutton, Raynal, & Schoenauer, 2000), one can count the number of pixels generated by the evolved IFS that are within the target, minus those that are without.
- If one uses GP for *symbolic regression* (cf. Figure 4), the usual evaluation function will typically be either a root *mean square error (MSE)* or a *relative MSE (RMSE)*:

$$\text{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f(x_i) - s_i)^2} \qquad (1)$$

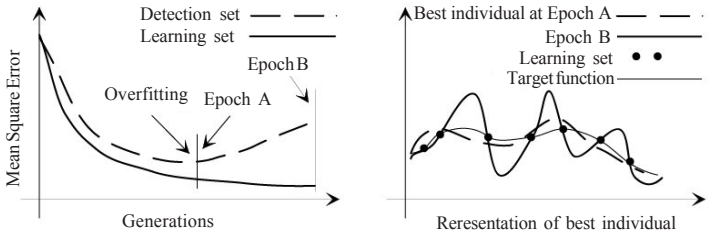
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{f(x_i) - s_i}{s_i}\right)^2} \qquad (2)$$

where f is the function to be evaluated on a *learning set* made of an array of n couples (x_i, s_i) where s_i is the known solution for x_i.

Evaluation of Evolved Individuals

It is important to know that a phenomenon called *overfitting* often occurs when evolving

Figure 3. Consequences of overfitting: At Epoch B (left figure), the same best individual will have a very good fitness on the training set and a bad fitness on the detection set. This individual, pictured with a continuous line on the right-hand side, matches points of the training set nearly perfectly, at the expense of generalization. In contrast, on the training set, the fitness of the best individual of Epoch A is lower than the fitness of the best individual of Epoch B; however, the best individual of Epoch A reveals a better generalization ability (right figure).



a program with GP. Overfitting is characterized by the fact that beyond a certain point, individuals only get better by overspecializing on the learning set.

In order to at the same time correctly evaluate evolved individuals and prevent the occurrence of overfitting, one should decompose the set of available values (called *training set*) into three subsets: "... a learning set, a detection set, and an evaluation set" (Gagné et al., 2006):

1. The *learning set* is used to guide the evolution of individuals. Fitness of individuals is determined with reference to the *learning set* only.
2. The *detection set* is only used to detect overfitting. Every *n* generations, the best individual found thanks to the *learning set* is tested on the *detection set*. In the first stages of evolution, the fitness of the same individual on the *learning set* and the *detection set* will follow the same curve until a certain point (cf. Figure 3a, Epoch A) where the performance of the best individual will decrease over the *detection set* while the same individual will keep doing better on the *learning set* (cf. Figure 3a, Epoch B). Overfitting is a stopping criterion.
3. Finally, the real performance of the best individual must be evaluated over an *evaluation set,* different from the two other subsets of the same *training set.*

Overfitting is probably caused by the very high driving force of artificial evolution. At the beginning of the run, creating individuals that will perfectly match the different points of the learning set is very difficult, since the first individuals are usually created randomly. It is difficult for very large unfit individuals to improve quickly because they contain a lot of genetic material. On the contrary, smaller individuals can evolve faster, meaning that the best individuals found in the first generations are usually smaller. Their fitness is not very good, but they generalize well. As evolution goes on, some individuals manage to improve, and it becomes harder and harder for small generic individuals to obtain a good evaluation. If the evolutionary process is not stopped, individuals will grow specific pieces of code in order to

exactly match the points of the *learning set.* Unfortunately, in order to do so, they will also become less generic (cf. Figure 3b). The fitness of the same individual over the *learning set* and over the *detection set* will become different, even though the *learning set* and *detection set* were subsets of the same original *training set.*

This overfitting problem appears not only on symbolic regression problems. The idea is to stop evolution when the fitness of the best individual over the *detection set* starts to decrease. One cannot use the fitness found on the *detection set* to give a good evaluation of the best found individual because the *detection set* has played a role in finding this best individual: the detection set is part of the evolution as it has been used as a stopping criterion.

Since the detection set is not independent of the evolution of individuals, one must use a third set (the *evaluation set*) to evaluate the evolved individual in a totally independent way.

Initialization Operator

Initialization is probably more important in genetic programming than in other evolutionary algorithms, due to the potential problem of bloat (which may be linked to overfitting).

In 1992, Koza introduced the *full, grow,* and *ramped half and half* algorithms to initialize tree functions (cf. Koza, 1992, p. 92).

Many other initialization methods have appeared ever since, such as *uniform* (Bohm & Geyer-Schultz, 1996), *random-branch* (Chellapilla, 1997), *ramped uniform* (Langdon, 2000), *rand-tree* (Iba, 1996), *PCT1* and *PCT2* (Luke, 2000b) and so on, taking at least one parameter: the maximum allowed depth. The aim of all these methods is to create the best possible random trees able to avoid premature convergence and apparition of bloat. However, a comparison between *ramped half and half, PCT1, PCT2, random branch,* and *uniform* by Luke and Panait (2001) did not show a real winner.

Even though it has been shown that Koza's algorithms were not flawless (Burke, Gustafson, & Kendall, 2002; Langdon, 2000), they are still the most widely used:

- **Full:** Takes as a parameter the maximum depth *T* of the tree to be created. Until this depth is reached, *full* chooses nodes among the functions set. When the tree is of depth *T,* nodes are chosen among terminals only.
- **Grow:** Takes as a parameter the maximum depth *T* of the tree to be created. Until this depth is reached, *grow* chooses nodes among functions and terminals. When the tree is of depth *T, grow* chooses its nodes among terminals only.
- **Ramped Half and Half:** Takes as a parameter the maximum depth *T* of the tree to be created. This method is proposed by Koza to create the most diverse trees. He suggests the creation of an equal number of trees of depth 2 to *T* (ramped) while alternating between *full* and *grow* (half and half).

Functions and terminals are picked up at random in their respective sets. The initialization method should respect the arity of operators in order to guarantee that created individuals are feasible (closing condition).

Parents Selection

All selectors used in evolutionary algorithms can be used in genetic programming (tournament, stochastic universal sampling, ranking, etc.). As in other evolutionary algorithms, one chooses the method best adapted to the fitness landscape and to the desired selection pres-

sure. If such information is not available, a binary tournament will probably be the best choice (cf. Chapter IV). If premature convergence occurs (flat fitness well before the stopping criterion is met), a good idea is to use a lower selection pressure with a stochastic tournament (cf. section on selection operators in Chapter IV). On the contrary, if the derivative of the fitness curve is still positive when the stopping criterion is met, this means that selection could have been harsher. An easy way to do this is to use a tournament between three or more individuals.

Finally, it is possible to select a same parent several times. This is the main difference between the selection operator and the replacement operator (cf. the following).

Genetic Operators

Here again, genetic operators are the same as those used in standard evolutionary algorithms—that is, mainly crossover and mutation. They are however adapted to the tree structure of individuals.

In his first book on GP, Koza (1992) suggests the use of mainly crossover rather than mutation, based on experiments on the Boolean multiplexer. Mutation is therefore usually set aside in GP, even though it has been shown that mutation may not be that inefficient compared to crossover (Luke & Spector, 1998).

- **Binary Genetic Operator (Crossover):** This is the operator of choice in GP, which attempts to combine already evolved subtrees (read building blocks) coming from selected parents. In Koza's work, if individuals use ADFs, crossovers do not occur between main programs and ADFs, but only between main programs or between equivalent ADFs.

Many different crossovers are described in the literature. Here are some of them:

- **Standard Crossover:** The operator described by Koza is the simplest. It consists of exchanging two subtrees randomly selected in the two parents. However, since in binary trees there are roughly as many leaves as there are nodes, a purely random choice would select a leaf in 50% of the cases. Koza therefore suggests the application of a bias of 90% towards nodes rather than leaves so that exchanged nodes are subtrees more often than leaves.
- **Height-Fair Crossover:** O'Reilly and Oppacher (1994) remark that crossovers have a greater probability to occur lower in the trees than higher. They therefore propose to first select uniformly a depth, and then, in the selected depth, choose a node to exchange with the other parent.
- **GP-Std/Same:** Based on the previous idea, Harries and Smith (1997) wonder whether it would be wise to choose an identical crossover depth for the two parents, hoping to exchange subtrees with a similar role. However, their tests show that *GP-Same* (same depth for the two parents) does not work as well as a standard crossover! However results are better than standard crossover if *GP-Same* is applied in 50% of the cases (hence the name *GP-Std/Same*).
- **Homologous Crossover:** Another idea is to find subtrees with similar structure and position in the two parents. In the first parent, a subtree is chosen (with 90% bias towards a node). In the second parent, one first explores the tree to find subtrees of similar structure. Among the matching subtrees, the one whose position corresponds best to the subtree of the first parent is selected.

Unfortunately, this homologous crossover does not seem to produce better results than a standard crossover, which seems to confirm that two subtrees with similar structure and position in two different individuals may not address the same problem.

However, homologous crossover fights bloat efficiently (cf. *Bloat* section that follows), which is a very good reason to use it, since the results are no worse than a standard crossover (Langdon, 2000).

All crossover methods must make sure that resulting individuals are feasible.

- **Unary Genetic Operator (Mutation):** Due to Koza's guiderules, mutation is generally set aside in genetic programming. When it is used, it is usually applied with a probability between 0 and 1%, however many different types of mutations have been studied. Mutation is applied either on children that have just been created through crossover or directly on selected parents. Standard operators are the following:
  - **Standard Mutation:** The idea is to select a node with a 90% probability (rather than a leaf, see above), delete its subtree, and grow it again randomly.
  - **Small Mutation:** The *standard mutation* is quite rough. The *small mutation* tries to operate a minimal mutation: if an operator is mutated, *small mutation* replaces it by another of the same arity. If a terminal was selected, it is replaced by another terminal. If the terminal is a numerical value, some Gaussian noise is added to it to change its value. The amount of perturbation can become smaller along the generations, thanks for instance to the auto-adaptive mutation used in evolution strategies (Schwefel, 1995).
  - **Mono-Parental "Crossover":** A mono-parental crossover is possible in genetic programming! This type of crossover (Kinnear, 1994), which is also called "inversion," consists of exchanging two subtrees of the same individual. This "crossover" is in fact a mutation because it uses only one individual.

All mutation methods must make sure that resulting individuals are feasible.

The last two mutations are considered as slightly too smooth and would not allow escape from a local maximum. Harries and Smith (1997) study the efficiency of these different mutations in a GP algorithm using mutation only. The *small mutation* and the *mono-parental crossover* are the worst methods, although they bring a visible improvement if they are used along with *standard mutation.*

A total of 58,000 tests of 14 different strategies on four different problems (Luke & Spector, 1998) shows that using mutation only in GP gives very good results, compared to a standard GP algorithm using mostly crossover. Therefore, Koza's bias towards crossover does not seem to be justified, and using mutation with a probability similar to that used in EAs seems fine.

Notion of Intron

In 1977, biologists discovered that 90% of the genetic code contained in ADN is not expressed. Genes are in fact made of coding regions (*exons*) separated by numerous and very long non-coding regions (*introns*) that are suppressed by enzymes, before messenger-ARN does its job. The role of introns is not well known. The sequence of nucleotides that compose introns is not random. The structure of the

introns suggests that they contain information of which the origin and function is yet unknown.

When programs are evolved using a genetic algorithm, one quickly wonders what should be done with superfluous code if during mutations, one allows a unary operator to replace a binary operator.

Eliminating the subtree is not satisfactory, because if the mutation occurs after many generations, the code contained in the subtree is no longer random: the subtree is the result of an evolution process that represents a significant CPU-cost that would be a pity to lose. Generally, this code is left where it is, and the unary operator will only use one of the two subtree operands of the initial binary operator. Would this mechanism be similar to what creates introns in biologic ADN (Nordin, Banzhaf, & Francone, 1997)? A subtree that is an intron is not totally lost, if a subsequent mutation of the unary operator turns it into a binary operator again. Another possibility is that this unexpressed code may be used later on by a crossover operator.

Interval Arithmetics

Since individuals must be executed to be evaluated, one must be cautious about operators or functions that may not be defined for tested values. Division, for instance, cannot have zero for denominator. For a long time, the adopted solution was to *protect* the operator by testing the denominator before executing the division. If the denominator was null, either 1 or a very large value was returned.

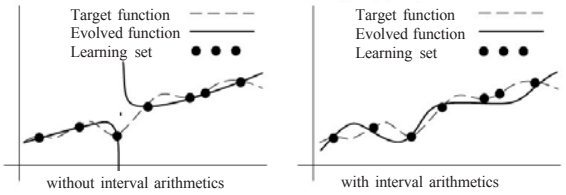
The problem with these "intelligent" operators is that they do not remove the vertical asymptote (in the case of the division). It is therefore possible to obtain a function that approximates very well a learning set that may contain such an asymptote (cf. Figure 4), with the consequence that when the evolved function is used in real conditions (outside of the learning set), it may return extravagant and unrealistic values.

The best solution is to make sure, when a child is generated, that all its operators will be used on an interval they can naturally handle (Keijzer, 2003): this is done by calculating the bounds of a function given the bounds of the operands, while excluding singular values.

New parameters are added: each node n is accompanied with its lower and upper bounds n_l and n_u. With this information, the bounds of the operator are evaluated using basic mathematical knowledge for all the different operators of the functions set:

- If node n is an addition between its operators a and b: the lower bound for node n is $n_l = a_l + b_l$ and the upper bound is $n_u = a_u + b_u$.
- If node n represents $a_l \times b_l$: $n_l = \min(a_l \times b_l, a_l \times b_u, a_u \times b_l, a_u \times b_u)$ and $n_u = \min(a_l \times b_l, u_l \times b_u, a_u \times b_l, a_u \times b_u)$.
- …

Figure 4. Protecting operators by making them "intelligent" can be very dangerous! A protected division will make it possible for GP to evolve the function on the left which can lead to big trouble if it is used for real. A much better solution is to only allow division if the interval of the denominator does not contain zero, even if the result is not as accurate (figure on the right).



It is possible to update these boundaries from bottom to top every time a modification is made, and make sure that all operators are valid on the calculated intervals of their operands. With *interval arithmetics,* one can make sure that the second operand of/never contains zero, for instance.

If such a case appears, Keijzer suggests that the individual be either assigned the worst possible fitness value or be deleted. Another option is to mutate the operator of the offending operand into another one that can accept the operand's interval.

Replacement Algorithm

Once children (created thanks to crossover or mutation) are evaluated, the total population is now made of parents and children. One must, therefore, reduce the number of individuals to come back to the original population size for the next generation.

In traditional genetic programming, replacement is either:

- *generational* (meaning that children replace parents), implying that for each generation, one creates as many children as there are parents in the initial population; or
- *steady-state*-like (i.e., only one child is created per generation that replaces the worst parent; cf. Chapter IV).

Studies have compared both methods (Syswerda, 1991), but there is no reason to limit oneself to these two replacement operators.

The $(\mu + \lambda)$ or (μ, λ) operators of the *evolution strategies* and their improvements (Beyer & Schwefel, 2002) can be used as well. As usual in all evolutionary algorithms, the problem to be solved dictates which method is the best.

The difference between the replacement operator and the parents selection operator is that for replacement, individuals cannot be selected more than once.

EXECUTION

Population Size vs. Number of Generations

The same question always arises in artificial evolution: for an identical number of evaluations, is it better to have many individuals and few generations, or few individuals and many generations?

The intuitive answer to this question is the same as for evolutionary algorithms. Using many individuals will postpone convergence. Since GP is supposed to be easily subject to premature convergence, a large population is generally used (Koza uses between 500 and 1,000 individuals) in order to widely explore the search space, with a number of generations usually lower than 100 (for a generational replacement, of course).

However, many studies show that there is no real reason behind these choices: in their paper comparing crossover and mutation, Luke and Spector (1998) also examine results when changing the population size between 4 and 2,048, and the number of generations between 512 and 1. No clear conclusion can be drawn: the winning combination is different depending on the problem.

Bloat

A mysterious *bloat* problem has already been evoked a couple of times in this chapter. As the number of generations increases, the general fitness improves, often due to individuals getting more complex.

Fitness usually follows a logarithmic curve, and there comes a time where a strong effort is required in order to obtain a very small improvement. A phenomenon can then occur that is called *bloat*: the size of all the individuals in the population increases beyond reason. The individuals being coded as a tree, the amount of memory they occupy and the time needed to execute them suddenly increases drastically, consuming all the resources of the computer within a small number of generations, without any visible improvement of their fitness.

Much work has been done to understand the source of the problem (Langdon & Poli, 1997; Luke, 2000a; Smith, 2000; Banzhaf & Langdon, 2002), as well as all the papers by Soule (1998, 2002; Soule & Foster, 1998; Soule, Foster, & Dickinson, 1996) on code growth analysis, and also Streeter (2003). Even though the problem was revealed by genetic programming, it looks like bloat can appear in any evolutionary algorithm using variable sized individuals (McPhee & Miller, 1995; Soule, 1998).

A current explanation is that larger individuals are protected against the destructive effects of genetic operators, making them more robust in the sense that their children have a larger chance than average to obtain a fitness similar to that of their parents (Blickle & Thiele, 1994; McPhee & Miller, 1995; Nordin & Banzhaf, 1995): as the exons/introns ratio decreases, code changes due to genetic operators will have more chances of affecting introns (without any effects on the phenotype) than exons. (Is it for this same reason that 90% of the human genome is made of introns?)

Soule therefore suggests that bloat (and robustness) is a byproduct of the execution of genetic operators (Soule, 2003). As a consequence, he suggests application of all genetic operators (including crossover) with a probability depending linearly on the number of nodes (usually the case with the mutation operator). In this case, larger individuals (with a low exons/introns ratio) would not be favored, as they would undergo more modifications of their code (Soule & Heckendorn, 2002; Stevens, et al., 2005). Rather than penalizing larger individuals directly by giving them an arbitrarily low fitness value (like in Poli's Tarpeian GP (Poli, 2003)), performing more genetic modifications is also some kind of penalization, but much more constructive as it can lead to the emergence of better individuals.

Overfitting and Bloat

Overfitting has already been evoked in the section on fitness function above. Most papers on bloat suggest that it may be a consequence of overfitting, as observation reveals that smaller individuals are usually more generic than larger ones. The idea would therefore be that at the beginning of evolution, it is easier for simple and small individuals to improve (because they do not contain a lot of code to optimize), but only up to a certain point.

As evolution goes on, unless one is very lucky, modifying simple individuals is no longer sufficient to obtain better results, because they only contain enough material to grossly approximate the learning set. If evolution continues with a high selection pressure, the only way to survive is to grow larger and larger, so that destructive genetic operators cannot corrupt exon code.

It is noticeable that controlling bloat also often prevents genetic programming from finding good individuals (Mahler, Robilliard, & Fonlupt, 2005), so there is no simple solution. The study of the relationship between bloat and overfitting thus remains a hot topic.

CONCLUSION

Koza uses an A/I ratio to evaluate the quality of Artificial Intelligence techniques. This ratio represents the amount of intelligence brought in by an *A*rtificial device, with reference to the amount of human *I*ntelligence that was needed to program it.

When Deep(er) Blue was able to beat Kasparov at Chess back in 1997, with a 3.5-2.5 score:

- The machine was capable of analyzing 200 million positions per second thanks to a 30-node RS/6000-based super-computer using 480 VLSI hardware chips designed for this purpose only.
- It used a huge database of 700,000 human grandmaster games.
- It was programmed with the help of four human grandmasters.
- The code was modified between each game by human experts and grandmasters so as to take into account Kasparov's current playstyle.

The A/I rate for Deep Blue is therefore quite low, because the computer only brought in brute force to a program that used a huge amount of human intelligence in its design (Hsu, 2002).

Conversely, genetic programming constantly proves to be a very high A/I technique, with human-competitive results obtained by computer programmers in domains where they are far from being experts, as is repeatedly shown in Koza's last book (Koza et al., 2003; Al-Sakran, Koza, & Jones, 2005), where he insists on how he routinely obtains excellent results on different problems from the same setup.

Thus, genetic programming is a very promising field that one should not overlook when faced with really complex problems to solve.

REFERENCES

Al-Sakran, S. H., Koza, J. R., & Jones, L. W. (2005, March 30-April 1). Automated re-invention of a previously patented optical lens system using genetic programming. In M. Keijzer, A. Tettamanzi, P. Collet, J. I. van Hemert, & M. Tomassini (Eds.), *Proceedings of the 8th European Conference on Genetic Programming* (Vol. 3447, pp. 25-37). Lausanne, Switzerland: Springer.

Banzhaf, W., & Langdon, W.B. (2002). Some considerations on the reason for bloat. *Genetic Programming and Evolvable Machines, 3*(1), 81-91.

Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies: A comprehensive introduction. *Natural Computing: An International Journal, 1*(1), 3-52.

Blickle, T., & Thiele, L. (1994, Saarbrücken, Germany). Genetic programming and redundancy. In J. Hopf (Ed.), "Genetic Algorithms Within the Framework of Evolutionary Computation," *Proceedings of KI-94* (pp. 33-38). Germany: Max-Planck-Institut fur Informatik.

Bohm, W., & Geyer-Schulz, A. (1996). Exact uniform initialization for genetic programming. In K. Richard & M. Vose (Eds.), *Foundations of genetic algorithms IV* (pp. 379-407). San Francisco: Morgan Kaufman.

Burke, E., Gustafson, S., & Kendall, G. (2002, July 9-13). A survey and analysis of diversity measures in genetic programming. In W. B. Langdon (Ed.), *Gecco 2002: Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 716-723). New York: Morgan Kaufmann.

Chellapilla, K. (1997). Evolving computer programs without subtree crossover. *IEEE Trans-*

actions on Evolutionary Computation, 1(3), 209-216.

Collet, P., Lutton, E., Raynal, F., & Schoenauer, M. (2000). Polar ifs + parisian gp = efficient inverse ifs problem solving. *Genetic Programming and Evolvable Machines, 1*(4), 339-361.

Cramer, N. L. (1985). A representation for the adaptive generation of simple sequential programs. *Proceedings of the International Conference on Genetic Algorithms and Their Applications* (pp. 183-187).

Friedberg, R. (1958). A learning machine: Part I. *IBM Research Journal, 2*(1).

Friedberg, R., Dunham, B., & North, J. (1958). A learning machine: Part II. *IBM Research Journal, 3*(3), 282-287.

Gagné, C., & Schoenauer, M. (2006). Genetic programming, validation sets, and parsimony pressure. In P. Collet, M. Tomassini, M. Ebner, S. Gustafson, & A. Ekárt (Eds.), *Proceedings of the 9th European Conference on Genetic Programming* (LNCS 3905, pp. 109-120). Budapest: Springer.

Harries, K., & Smith, P. (1997). Exploring alternative operators and search strategies in genetic programming. In J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, et al. (Eds.), *Genetic Programming 1997: Proceedings of the 2nd Annual Conference* (pp. 147-155). San Francisco: Morgan Kaufmann.

Hsu, F.-H. (2002). *Behind Deep Blue: Building the computer that defeated the world chess champion.* Princeton, NJ: Princeton University Press.

Iba, H. (1996). Random tree generation for genetic programming. In H.-M. Voigt, et al. (Eds.), *PPSN IV* (pp. 144-153). Berlin: Springer-Verlag (LNCS 1141).

Keijzer, M. (2003). Improving symbolic regression with interval arithmetic and linear scaling. In C. Ryan, T. Soule, M. Keijzer, E. P. K. Tsang, R. Poli, & E. Costa (Eds.), *EUROGP* (Vol. 2610, pp. 70-82). Berlin: Springer-Verlag.

Kinnear, K. E. (1994). Alternatives to Automatically Defined Functions. In *Advances in genetic programming*. (pp. 119-141). Cambridge, MA: MIT Press.

Koza, J. R. (1989). Hierarchical genetic algorithms operating on populations of computer programs. *Proceedings of the 11th International Joint Conference on Artificial Intelligence* (vol. I, pp. 768-774). San Francisco: Morgan Kaufmann.

Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural evolution.* Cambridge, MA: MIT Press.

Koza, J. R. (1994). *Genetic programming II: Automatic discovery of reusable programs.* Cambridge, MA: MIT Press.

Koza, J. R., Bennett, F. H., III, Andre, D., & Keane, M. A. (1999). *Genetic programming III: Automatic synthesis of analog circuits.* Cambridge, MA: MIT Press.

Koza, J. R., Keane, M. A, Streeter, M. J. Mydlowec, W., Yu, J., & Lanza, G. (2003). *Genetic programming IV: Routine human-competitive machine intelligence.* Kluwer Academic.

Langdon, W. B. (2000). Size fair and homologous tree crossovers for tree genetic programming. *Genetic Programming and Evolvable Machines, 1,* 95-119.

Langdon, W. B., & Poli, R. (1997, 24 February). *Fitness causes bloat.* Technical Report No. CSRP-97-09, School of Computer Science, University of Birmingham, UK.

Luke, S. (2000a, July 8). Code growth is not caused by introns. In D. Whitley (Ed.), *Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference* (pp. 228-235), Las Vegas, NV.

Luke, S. (2000b). Two fast tree-creation algorithms for genetic programming. *IEEE Transactions in Evolutionary Computation, 4*(3), 274-283.

Luke, S., & Panait, L. (2001). A survey and comparison of tree generation algorithms. In L. Spector (Ed.), *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-2001) (pp.81-88). San Francisco: Morgan Kaufmann.

Luke, S., & Spector, L. (1998, July 22-25). A revised comparison of crossover and mutation in genetic programming. In J. R. Koza et al. (Eds.), *Genetic Programming 1998: Proceedings of the 3rd Annual Conference,* Madison, WI (pp. 208-213). San Francisco: Morgan Kaufmann.

Mahler, S., Robilliard, D., & Fonlupt, C. (2005). Tarpeian bloat control and generalization accuracy. In M. Keijzer, A. Tettamanzi, P. Collet, J. I. van Hemert, & M. Tomassini (Eds.), *EUROGP* (Vol. 3447, pp. 203-214). Berlin: Springer-Verlag.

McPhee, N. F., & Miller, J. D. (1995, July 15-19). Accurate replication in genetic programming. In L. Eshelman (Ed.), *Genetic algorithms: Proceedings of the 6th International Conference* (ICGA95), Pittsburgh, PA (pp. 303-309). San Francisco: Morgan Kaufmann.

Nordin, P., & Banzhaf, W. (1995, July 15-19). Complexity compression and evolution. In L. Eshelman (Ed.), *Genetic Algorithms: Proceedings of the 6th International Conference* (ICGA95), Pittsburgh, PA (pp. 310-317). San Francisco: Morgan Kaufmann.

Nordin, P., Banzhaf, W., & Francone, F. D. (1997). Introns in nature and in simulated structure evolution. In D. Lundh, B. Olsson, & A. Narayanan, (Eds.), *Bio-computing and emergent computation.* Berlin: Springer-Verlag.

O'Reilly, U.-M., & Oppacher, F. (1994). *Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing.* Technical Report No. 94-04-021, Santa Fe Institute, USA.

Poli, R. (2003, April 14-16). A simple but theoretically motivated method to control bloat in genetic programming. In C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, & E. Costa (Eds.), *Genetic Programming, Proceedings of EUROGP'2003* (Vol. 2610, pp. 200-210). Essex: Springer-Verlag.

Schwefel, H.-P. (1995). *Numerical optimization of computer models* (2nd ed.). New-York: John Wiley & Sons.

Smith, P.W.H. (2000). Controlling code growth in genetic programming. In R. John & R. Birkenhead (Eds.), *Advances in soft computing* (pp. 166-171). Leicester, UK: Physica-Verlag.

Smith, S. F. (1980). *A learning system based on genetic adaptive algorithms.* Unpublished doctoral dissertation, University of Pittsburgh, USA.

Soule, T. (1998). *Code growth in genetic programming.* Unpublished doctoral dissertation, University of Idaho, USA.

Soule, T. (2002, April 3-5). Exons and code growth in genetic programming. In J. A. Foster, E. Lutton, J. Miller, C. Ryan, & A.G.B. Tettamanzi (Eds.), *Genetic Programming, Proceedings of the 5th European Conference* (EUROGP 2002), Kinsale, Ireland (Vol. 2278, pp. 142-151). Berlin: Springer-Verlag.

Soule, T. (2003). Operator choice and the evolution of robust solutions. In R. L. Riolo & B. Worzel (Eds.), *Genetic programming theory and practice* (pp. 257-270). Kluwer.

Soule, T., & Foster, J. A. (1998, May 5-9). Removal bias: A new cause of code growth in tree based evolutionary programming. *Proceedings of the IEEE International Conference on Evolutionary Computation*, Anchorage, Alaska (pp. 781-186). IEEE Press.

Soule, T., Foster, J. A., & Dickinson, J. (1996, July 28-31). Code growth in genetic programming. In J. R. Koza, D.E. Goldberg, D. B. Fogel, & R. L. Riolo (Eds.), *Genetic Programming 1996: Proceedings of the 1st Annual Conference* (pp. 215-223), Stanford University, CA. Cambridge, MA: MIT Press.

Soule, T., & Heckendorn, R.B. (2002, September). An analysis of the causes of code growth in genetic programming. *Genetic Programming and Evolvable Machines, 3*(3), 283-309.

Stevens, J., Heckendorn, R. B. , Soule, T. (2005). Exploiting disruption aversion to control code bloat. In H.-G. Beyer, U.-M. O'Reilly, D. V. Arnold, W. Banzhaf, C. Blum, E. W. Bonabeau, et al. (Eds.), *Proceedings of the 2005 conference on genetic and evolutionary computation*. Washington, DC: ACM Press.

Streeter, M. J. (2003, April 14-16). The root causes of code growth in genetic programming. In C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, & E. Costa (Eds.), *Genetic Programming*, *Proceedings of EUROGP'2003* (vol. 2610, pp. 449-458). Essex: Springer-Verlag.

Syswerda, G. (1991, July 15-18). A study of reproduction in generational and steady state genetic algorithms. In G. J. E. Rawlins (Ed.), *Proceedings of the 1st Workshop on Foundations of Genetic Algorithms* (pp. 94-101). San Mateo: Morgan Kaufmann.

Wang, G., & Soule, T. (2004). How to choose appropriate function sets for genetic programming. In M. Keijzer, U.-M. O'Reilly, S. M. Lucas, E. Costa, & T. Soule (Eds.), *EUROGP* (Vol. 3003, pp. 198-207). Berlin: Springer-Verlag.

KEY TERMS

A/I Ratio: The *artificial to intelligence ratio* is defined by John Koza as the ratio of that which is delivered by the automated operation of the artificial method to the amount of intelligence that is supplied by the human applying the method to a particular problem.

Bloat: In some cases, the size of individuals will suddenly grow beyond reason, consuming all the resources of the host computer within a handful of generations. It is believed that bloat is related to overfitting.

Exon: Coding region of the ADN. Exons represent less than 10% of the human genome.

Intron: Non-coding region of the ADN. Introns represent more than 90% of the human genome.

Overfitting: Genetic programming evolves individuals over a training set, hopefully representative of the function to be approximated. However, the evolution driving force is usually so strong that individuals will develop specific code to match the training set as well as possible. An "overfit" solution will have a very good fitness on the training set, but will perform poorly on real data.

Chapter VI
Evolutionary Multi-Objective Optimization in Finance

Carlos Artemio Coello Coello
Evolutionary Computation Group (EVOCINV), Mexico

ABSTRACT

This chapter provides a brief introduction of the use of evolutionary algorithms in the solution of multi-objective optimization problems (an area now called "evolutionary multi-objective optimization"). Besides providing some basic concepts and a brief description of the approaches that are more commonly used nowadays, the chapter also provides some of the current and future research trends in the area. In the final part of the chapter, we provide a short description of the sort of applications that multi-objective evolutionary algorithms have found in finance, identifying some possible paths for future research.

INTRODUCTION

Many real-world problems have two or more objective functions that we aim to minimize. Such problems are called multi-objective optimization problems and require an alternative definition of "optimality." The most common notion of optimality normally adopted is the so-called *Pareto optimality,* which indicates that the best possible solutions are those representing the best trade-offs among the objective functions. In other words, the desirable solutions are those in which one objective cannot be improved without worsening another objective.

Evolutionary algorithms (EAs) are techniques based on the emulation of the mechanism of natural selection, which have been successfully used to solve problems during several years (Fogel, 1999; Goldberg, 1989). One of the problem domains in which EAs have been found to be particularly useful is in multi-objective optimization (Coello Coello, Van Veldhuizen, & Lamont, 2002). EAs are particularly suitable for solving multi-objective optimization problems because they deal simultaneously with a set of possible solutions (the so-called population) which allows us to find several members of the Pareto optimal set (i.e., the

best possible trade-offs found) in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques. Additionally, EAs do not require the derivatives of the objective functions and are less susceptible to any features of the problem (e.g., discontinuities either in decision variable space or in objective function space).

The first documented attempt to solve a multi-objective optimization problem using an evolutionary algorithm dates back to the mid-1980s (Schaffer, 1984, 1985). Since then, a considerable amount of research has been done in this area, now known as evolutionary multi-objective optimization (EMO, for short). The growing importance of this field is reflected by a significant increment (mainly during the last ten years) of technical papers in international conferences and peer-reviewed journals, special sessions in international conferences, and interest groups on the Internet.[1]

BASIC CONCEPTS

Definition 1 (Global Minimum): Given a function $f : \Omega \subseteq R^n \to R$, $\Omega \neq \varnothing$, for $\vec{x} \in \Omega$ the value $f^* f(\vec{x}^*) > -\infty$ is called a global minimum if and only if

$$\forall \vec{x} \in \Omega : \; f(\vec{x}^*) \leq f(\vec{x}) . \tag{1}$$

Then, $\vec{x}^*$ is the global minimum solution, f is the objective function, and the set Ω is the feasible region ($\Omega \in S$), where S represents the whole search space.

Definition 2 (General Multi-Objective Optimization Problem (MOP)): Find the vector $\vec{x}^* = \left[x_1^*, x_2^*, ..., x_n^*\right]^T$ which will satisfy the *m* inequality constraints:

$$g_i(\vec{x}) \leq 0 i = 1, 2, ..., m \tag{2}$$

the *p* equality constraints

$$h_i(\vec{x}) = 0 i = 1, 2, ..., p \tag{3}$$

and will optimize the vector function

$$\vec{f}(\vec{x}) = \left[f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})\right]^T \tag{4}$$

where $\vec{x} = \left[x_1, x_2, ..., x_n\right]^T$ is the vector of decision variables.

Definition 3 (Pareto Optimality): *A point $\vec{x}^* \in \Omega$ is **Pareto optimal** if for every $\vec{x} \in \Omega$ and I = {1, 2,..., k} either,*

$$\forall_{i \in I} (f_i(\vec{x}^*) \leq f_i(\vec{x})) \tag{5}$$

and, there is at least one $i \in I$ such that

$$f_i(\vec{x}^*) < f_i(\vec{x}) \tag{6}$$

In words, this definition says that $\vec{x}^*$ is Pareto optimal if there exists no feasible vector $\vec{x}$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. The phrase "Pareto optimal" is considered to mean with respect to the entire decision variable space unless otherwise specified.

Definition 4 (Pareto Dominance): *A vector $\vec{u} = (u_1, ..., u_k)$ is said to dominate $\vec{v} = (v_1, ..., v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if u is partially less than v, that is,*

$$\forall i \in \{1, ..., k\}, u_i \leq v_i \wedge \exists i \in \{1, ..., k\} : u_i < v_i .$$

Definition 5 (Pareto Optimal Set): For a given MOP $\vec{f}(x)$, the Pareto optimal set (P^*) is defined as:

$$P^* := \{x \in \Omega \mid \neg\exists\, x' \in \Omega \;\; \vec{f}(x') \preceq \vec{f}(x)\}. \qquad (7)$$

Pareto optimal solutions are also termed non-inferior, admissible, or efficient solutions (Horn, 1997); their corresponding vectors are termed non-dominated.

Definition 6 (Pareto Front): For a given MOP $\vec{f}(x)$ and Pareto optimal set P^*, the Pareto front (PF^*) is defined as:

$$PF^* := \{\vec{u} = \vec{f} = (f_1(x), ..., f_k(x)) \mid x \in P^*\}. \qquad (8)$$

In the general case, it is impossible to find an analytical expression of the line or surface that contains these points. The normal procedure to generate the Pareto front is to compute the feasible points Ω and their corresponding $f(\Omega)$. When there is a sufficient number of these, it is then possible to determine the non-dominated points and to produce the Pareto front.

ORIGINS OF EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION

Traditional evolutionary algorithms cannot properly deal with multi-objective optimization problems because of two main reasons:

1. Due to stochastic noise, evolutionary algorithms tend to converge to a single solution if run for a sufficiently large number of iterations. Thus, it is necessary to block the selection mechanism so that different solutions (which are all non-dominated) are preserved in the population of an evolutionary algorithm.
2. It is desirable that all non-dominated solutions are sampled at the same rate during the selection stage (i.e., that they all are considered with the same survival probability), since all non-dominated solutions are equally good among themselves.

Thus, it is necessary to introduce certain modifications into an evolutionary algorithm in order to make it suitable to solve multi-objective optimization problems.

Over the years, there have been many different proposals to extend evolutionary algorithms to solve multi-objective optimization problems. Historically, it is possible to consider three periods:

1. Origins
2. First generation
3. Second generation

In this section, we will focus in the first period, and in the two further sections, we will discuss the others.

The first actual implementation of what it is now called a multi-objective evolutionary algorithm (or *MOEA,* for short) was Schaffer's vector evaluation genetic algorithm (VEGA), which was introduced in the mid-1980s, mainly aimed for solving problems in machine learning (Schaffer, 1984, 1985; Schaffer & Grefenstette, 1985). So, this work is considered as the origin of research in this area.

VEGA basically consisted of a simple genetic algorithm (GA) with a modified selection mechanism. At each generation, a number of sub-populations are generated by performing proportional selection according to each objective function in turn. Thus, for a problem with k objectives, k sub-populations of size N/k each are generated (assuming a total population size of N). These sub-populations are then shuffled together to obtain a new population of size N, on which the GA applies the crossover and mutation operators in the usual way. Schaffer realized that the solutions generated by his approach were non-dominated in a local sense,

because their non-dominance was limited to the current population, which was obviously not appropriate. Also, he noted something that was called "middling" performance.[2] An individual that had this problem in all the objectives was perhaps a good compromise solution, but could not survive under the selection scheme of VEGA, because it was not the best in any particular objective. Thus, this problem opposes the goal of finding Pareto optimal solutions. Although the middling problem can be dealt with using heuristics or other additional mechanisms, it remained as the main drawback of VEGA.

From the second half of the 1980s up to the first half of the 1990s, few other researchers developed MOEAs. Additionally, most of these approaches were very naive and relied on aggregating functions (linear in most cases) (Syswerda & Palmucci, 1991), lexicographic ordering (Fourman, 1985), and target-vector approaches (Wienke, Lucasius, & Kateman, 1992). All of these approaches were strongly influenced by the work done in the operations research community and in most cases did not require any major modifications to the evolutionary algorithm adopted (except for the definition of the fitness function).

Although most of these early MOEAs are rarely referenced in the current literature, this historical period is of great importance because it provided the first insights into the possibility of using evolutionary algorithms for multi-objective optimization. Over the years, researchers would design more sophisticated MOEAs, giving rise to the two generations that are discussed in the two further sections.

THE FIRST GENERATION

The major step towards the first actual generation of MOEAs was given by David E. Goldberg on pages 199 to 201 of his famous book on genetic algorithms published in 1989 (Goldberg, 1989). In his book, Goldberg analyzes VEGA and proposes a selection scheme based on the concept of Pareto optimality. Goldberg not only suggested what would become the standard first-generation MOEA, but also indicated that stochastic noise would make such algorithm useless unless some special mechanism was adopted to block convergence. First-generation MOEAs typically adopt niching or fitness sharing (Deb & Goldberg, 1989) for that sake. Three are the most representative algorithms from the first generation:

1. **Non-Dominated Sorting Genetic Algorithm (NSGA):** This algorithm was proposed by Srinivas and Deb (1994). The NSGA is based on several layers of classifications of the individuals as suggested in Goldberg (1989). Before selection is performed, the population is ranked on the basis of non-domination: all non-dominated individuals are classified into one category (with a dummy fitness value, which is proportional to the population size, to provide an equal reproductive potential for these individuals). To maintain the diversity of the population, these classified individuals are shared with their dummy fitness values. Then this group of classified individuals is ignored and another layer of non-dominated individuals is considered. The process continues until all individuals in the population are classified. Stochastic remainder proportionate selection is adopted for this technique. Since individuals in the first front have the maximum fitness value, they always get more copies than the rest of the population. This allows the search for non-dominated regions, and results in convergence of the population toward such regions.

Sharing, by its part, helps to distribute the population over this region (i.e., the Pareto front of the problem).

2. **Niched-Pareto Genetic Algorithm (NPGA):** Proposed in Horn, Nafpliotis, and Goldberg (1994). The NPGA uses a tournament selection scheme based on Pareto dominance. The basic idea of the algorithm is the following: Two individuals are randomly chosen and compared against a subset from the entire population (typically, around 10% of the population). If one of them is dominated (by the individuals randomly chosen from the population) and the other is not, then the non-dominated individual wins. When both competitors are either dominated or non-dominated (i.e., there is a tie), the result of the tournament is decided through fitness sharing (Goldberg & Richardson, 1987). More recently (Erickson, Mayer, & Horn, 2001), NPGA 2 was proposed. This algorithm relies on a traditional Pareto ranking approach (similar to Fonseca and Fleming's (1993) MOGA), but it keeps its tournament selection scheme. Ties are solved through fitness sharing as in its predecessor. However, the niche count of NPGA 2 is computed using individuals from the next partially filled generation using a technique called "continuously updated fitness sharing" (Oei, Goldberg, & Chang, 1991).
3. **Multi-Objective Genetic Algorithm (MOGA):** Proposed in Fonseca and Fleming (1993). In MOGA, the rank of a certain individual corresponds to the number of chromosomes in the current population by which it is dominated. Consider, for example, an individual x_i at generation t, which is dominated by $p_i^{(t)}$ individuals in the current generation.

The rank of an individual is given by (Fonseca & Fleming, 1993):

$$\text{rank}(x_i, t) = 1 + p_i^{(t)} \quad (9)$$

All non-dominated individuals are assigned rank 1, while dominated ones are penalized according to the population density of the corresponding region of the trade-off surface. Fitness assignment is performed in the following way (Fonseca & Fleming, 1993):

1. Sort population according to rank.
2. Assign fitness to individuals by interpolating from the best (rank 1) to the worst (rank $n \leq M$) in the way proposed by Goldberg (1989), according to some function, usually linear, but not necessarily.
3. Average the fitnesses of individuals with the same rank, so that all of them are sampled at the same rate. This procedure keeps the global population fitness constant while maintaining appropriate selective pressure, as defined by the function used.

From these three algorithms, a few comparative studies undertaken during the mid- and late 1990s indicated that MOGA was the most effective and efficient approach, followed by NPGA and by NSGA (in a distant third place) (Coello Coello, 1996; Van Veldhuizen, 1999). This period was characterized by the use of selection mechanisms based on Pareto ranking and by the use of fitness sharing to maintain diversity. The papers of this period normally rely on visual comparisons of results (little work was done regarding the use of performance measures to allow quantitative comparisons of results before the mid-1990s), and normally incorporate very simple test functions.

THE SECOND GENERATION

The second generation of MOEAs was born with the introduction of the notion of elitism.[3] In the context of multi-objective optimization, elitism usually (although not necessarily) refers to the use of an external population (also called secondary population) to retain the non-dominated individuals. The use of this external file raises several questions:

- How does the external file interact with the main population (e.g., do we select the union of the main population and the external file)?
- What do we do when the external file is full (assuming that the capacity of the external file is bounded)?
- Do we impose additional criteria to enter the file instead of just using Pareto dominance (e.g., use the distribution of solutions as an additional criterion)?

Besides the use of an external file, elitism can also be introduced through the use of a $(\mu + \lambda)$-selection in which parents compete with their children, and those which are non-dominated (and possibly comply with some additional criterion such as providing a better distribution of solutions) are selected for the following generation. Besides the notion of elitism, efficiency (both at an algorithmic level and at the data structures level) has become a concern for researchers in this area (e.g., Jensen, 2003b; Coello Coello & Toscano Pulido, 2001; Mostaghim, Teich, & Tyagi, 2002). The second generation is also characterized by the use of performance measures to provide a quantitative (rather than only a qualitative) comparison of results (Zitzler, Deb, & Thiele, 2000; Van Veldhuizen & Lamont, 2000b; Fonseca & Fleming, 1996). However, the several drawbacks of many performance measures developed during the second generation (e.g., Knowles & Corne, 2002; Zitzler, Laumanns, Thiele, Fonseca, & Grunert da Fonseca, 2002) have (ironically) brought back many researchers to adopt visual comparisons as in the origins of the field.

Some MOEAs that are representative of the research trends of the second generation are the following:

1. **Strength Pareto Evolutionary Algorithm (SPEA):** This algorithm was introduced in Zitzler and Thiele (1999). This approach was conceived as a way of integrating different MOEAs. SPEA uses an archive containing non-dominated solutions previously found (the so-called external non-dominated set). At each generation, non-dominated individuals are copied to the external non-dominated set. For each individual in this external set, a strength value is computed. This strength is similar to the ranking value of MOGA (Fonseca & Fleming, 1993), since it is proportional to the number of solutions to which a certain individual dominates. In SPEA, the fitness of each member of the current population is computed according to the strengths of all external non-dominated solutions that dominate it. The fitness assignment process of SPEA considers both closeness to the true Pareto front and even distribution of solutions at the same time. Thus, instead of using niches based on distance, Pareto dominance is used to ensure that the solutions are properly distributed along the Pareto front. Although this approach does not require a niche radius, its effectiveness relies on the size of the external non-dominated set. In fact, since the external non-dominated set participates in the selection process of SPEA, if its size grows too large, it might

reduce the selection pressure, thus slowing down the search. Because of this, the authors decided to adopt a technique that prunes the contents of the external non-dominated set so that its size remains below a certain threshold. The approach adopted for this sake was a clustering technique called "average linkage method" (Morse, 1980).

2. **Strength Pareto Evolutionary Algorithm 2 (SPEA2):** SPEA2 has three main differences with respect to its predecessor (Zitzler, Laumanns, & Thiele, 2001): (1) it incorporates a fine-grained fitness assignment strategy which takes into account for each individual the number of individuals that dominate it and the number of individuals by which it is dominated; (2) it uses a nearest neighbor density estimation technique which guides the search more efficiently, and (3) it has an enhanced archive truncation method that guarantees the preservation of boundary solutions.
3. **Pareto Archived Evolution Strategy (PAES):** This algorithm was introduced in Knowles and Corne (2000). PAES consists of a (1+1) evolution strategy (i.e., a single parent that generates a single offspring) in combination with a historical archive that records the non-dominated solutions previously found. This archive is used as a reference set against which each mutated individual is being compared. Such a historical archive is the elitist mechanism adopted in PAES. However, an interesting aspect of this algorithm is the procedure used to maintain diversity which consists of a crowding procedure that divides objective space in a recursive manner. Each solution is placed in a certain grid location based on the values of its objectives (which are used as its "coordinates" or "geographical location"). A map of such grid is maintained, indicating the number of solutions that reside in each grid location. Since the procedure is adaptive, no extra parameters are required (except for the number of divisions of the objective space).
4. **Non-Dominated Sorting Genetic Algorithm II (NSGA-II):** This approach was introduced in Deb, Agrawal, Pratab, and Meyarivan (2000) and in Deb, Pratap, Agarwal, and Meyarivan (2002) as an improved version of the NSGA (Srinivas & Deb, 1994).[4] In the NSGA-II, for each solution one has to determine how many solutions dominate it and the set of solutions to which it dominates. The NSGA-II estimates the density of solutions surrounding a particular solution in the population by computing the average distance of two points on either side of this point along each of the objectives of the problem. This value is the so-called crowding distance. During selection, the NSGA-II uses a crowded-comparison operator which takes into consideration both the non-domination rank of an individual in the population and its crowding distance (i.e., non-dominated solutions are preferred over dominated solutions, but between two solutions with the same non-domination rank, the one that resides in the less crowded region is preferred). The NSGA-II does not use an external memory as the other MOEAs previously discussed. Instead, the elitist mechanism of the NSGA-II consists of combining the best parents with the best offspring obtained (i.e., a $(\mu+\lambda)$-selection). Due to its clever mechanisms, the NSGA-II is much more efficient (computationally speaking) than its predecessor, and its performance is so good that it has become very popular in the last few years, becoming a landmark

against which other multi-objective evolutionary algorithms have to be compared.

Many other algorithms exist (e.g., Coello Coello & Toscano Pulido, 2001; Corne, Knowles, & Oates, 2000; Corne, Jerram, Knowles, & Oates, 2001; Van Veldhuizen & Lamont, 2000a; Zydallis, Lamont, & Veldhuizen, 2000). The interested reader should consult additional sources for more details (Coello Coello et al., 2002; Coello Coello, 1999; Deb, 2001; Osyczka, 2002; Collette & Siarry, 2003; Tan, Khor, & Lee, 2005).

CURRENT RESEARCH TRENDS

According to the historical view of evolutionary multi-objective optimization presented at the beginning of this chapter, we are currently living the second generation. So far, researchers have not produced a breakthrough that is so significant as to redirect most of the research into a new direction. However, there are several interesting ideas that have certainly influenced some of the work being done these days. Some examples are the following:

- *The use of relaxed forms of Pareto dominance* has become popular as a mechanism to regulate convergence of an MOEA. From these mechanisms, ε-dominance is, with no doubt, the most popular (Laumanns, Thiele, Deb, & Zitzler, 2002). ε-dominance allows control of the granularity of the approximation of the Pareto front obtained. As a consequence, it is possible to accelerate convergence using this mechanism (if we are satisfied with a very coarse approximation of the Pareto front).
- *The transformation of single-objective problems into a multi-objective form that somehow facilitates their solution—* For example, some researchers have proposed the handling of the constraints of a problem as objectives (Coello Coello, 2002), and others have proposed the so-called "multi-objectivization" by which a single-objective optimization problem is decomposed into several subcomponents considering a multi-objective approach (Jensen, 2003a; Knowles, Watson, & Corne, 2001). This procedure has been found to be helpful in removing local optima from a problem.
- *The use of alternative bio-inspired heuristics for multi-objective optimization—* The most remarkable examples are particle swarm optimization (Kennedy & Eberhart, 2001) and differential evolution (Price, 1999), whose use has become increasingly popular in multi-objective optimization (e.g., Abbass & Sarker, 2002; Coello Coello, Toscano Pulido, & Salazar Lechuga, 2004). However, other bio-inspired algorithms such as artificial immune systems have also been used to solve multi-objective optimization problems (Coello Coello & Cruz Cortes, 2005).

FUTURE RESEARCH TRENDS

There are several topics that involve challenges that will keep researchers in this area busy for the next few years. Some of these include:

- *How to deal with problems that have "many" objectives:* Some recent studies have shown that traditional Pareto ranking schemes do not behave well in the presence of many objectives (where "many" is normally a number above 3 or 4) (Purshouse, 2003).
- *How to compare (in a quantitative way) the performance of several MOEAs:*

Despite the existence of a considerable number of performance measures that intend to compare (in a quantitative way) the performance of several MOEAs, many of them are not appropriate because their definition is not compliant with Pareto dominance (Zitzler, Thiele, Laumanns, Fonseca, & Fonseca, 2003).

- There are plenty of fundamental questions that remain unanswered. For example: What are the sources of difficulty of a multi-objective optimization problem for a MOEA? What are the dimensionality limitations of current MOEAs? Can we use alternative mechanisms into evolutionary algorithms to generate non-dominated solutions without relying on Pareto ranking?

SOME APPLICATIONS IN FINANCE

Evolutionary algorithms in general and MOEAs in particular can be useful in the solution of complex problems for which no efficient deterministic algorithm exists (i.e., there is no deterministic algorithm that can solve them in polynomial time).

It is well known that in finance there are several NP-complete problems for which the use of a heuristic is clearly justified (Schlottmann & Seese, 2004). However, the specialized literature on MOEAs reports few papers that deal with problems in finance. Some examples are:

- Solution of portfolio optimization problems, particularly using Markowitz models (e.g., Shoaf & Foster, 1996; Vedarajan, Chan, & Goldberg, 1997; Chang, Meade, & Beasley, 2000; Lin, Wang, & Yan, 2001; Streichert, Ulmer, & Zell, 2004; Doerner, Gutjahr, Hartl, Strauss, & Stummer, 2004; Ehrgott, Klamroth, & Schwehm, 2004). This has been, by far, the most popular application of MOEAs in finance.
- Time series prediction (Zwir & Ruspini, 1999; Ruspini & Zwir, 1999).
- Risk-return trade-offs for loans (Mukerjee, Biswas, Deb, & Mathur, 2002; Schlottmann & Seese, 2002).

Evidently, it is necessary to identify other types of problems in finance whose complexity justifies the use of an MOEA (e.g., Chen, 2002). Even the financial problems that have been tackled so far normally require special treatment and a proper tailoring of the current MOEAs (e.g., regarding the encoding, since portfolio selection problems can be modeled as knapsack problems (Streichert et al., 2004)). Additionally, the decision-making process involved in financial applications is normally very complex and difficult to automate. This presents challenges for the (few) models for incorporation of preferences in current use with MOEAs (Cvetkovic & Parmee, 2002; Coello Coello, 2000a; Coello Coello et al., 2002). Thus, financial applications present research opportunities both for experts in finance and for researchers working exclusively in the development of multi-objective evolutionary algorithms and associated techniques.

CONCLUSION

In this chapter, we have presented a brief introduction to evolutionary multi-objective optimization. We have provided some basic concepts and a historical perspective of the research that has been done in this area. We have also presented short descriptions of some algorithms that are representative of each historical period under consideration including the current one.

In the last part of the chapter, we have presented some of the current and future re-

search trends in the area, as well as a brief description of the sort of financial applications that have been developed using multi-objective evolutionary algorithms.

The main aim of this chapter is to provide a general overview of the area, identifying some opportunity areas mainly related to financial applications.

ACKNOWLEDGMENT

The author acknowledges support from NSF-CONACyT Project No. 42435-Y.

ENDNOTES

1 The author maintains an EMO repository with over 2,100 bibliographical entries at http://delta.cs.cinvestav.mx/~ccoello/EMOO

2 By "middling," Schaffer meant an individual with acceptable performance, perhaps above average, but not outstanding for any of the objective functions.

3 Although there were some early studies that considered the notion of elitism in a multi-objective evolutionary algorithm (e.g., Husbands, 1994; Osyczka & Kundu, 1995), most authors credit Zitzler with the formal introduction of this concept in a multi-objective evolutionary algorithm, mainly because his SPEA was published in a specialized journal (*IEEE Transactions on Evolutionary Computation*—Zitzler & Thiele, 1999) which made it a landmark in the field. Needless to say, after the publication of this paper, most researchers in the field started to incorporate external populations in their multi-objective evolutionary algorithms.

4 Note however that the differences between NSGA-II and NSGA are so significant that they are considered as two completely different algorithms by several researchers.

REFERENCES

Abbass, H., & Sarker, R. (2002). The Pareto differential evolution algorithm. *International Journal on Artificial Intelligence Tools, 11*(4), 531-552.

Chang, T. J., Meade, N., & Beasley, J. E. (2000). Heuristics for cardinality constrained portfolio optimization. *Computers and Operations Research, 27*(13), 1271-1302.

Chen, S.-H. (Ed.). (2002). *Evolutionary computation in economics and finance.* Heidelberg, Germany: Physica-Verlag.

Coello Coello, C.A. (1996). *An empirical study of evolutionary techniques for multiobjective optimization in engineering design.* Department of Computer Science, Tulane University, USA.

Coello Coello, C. A. (1999, August). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems. An International Journal, 1*(3), 269-308.

Coello Coello, C. A. (2001, July). Handling preferences in evolutionary multiobjective optimization: A survey. *Proceedings of the 2000 Congress on Evolutionary Computation* (Vol. 1, pp. 30-37). Piscataway, NJ: IEEE Service Center.

Coello Coello, C. A. (2002). Treating constraints as objectives for single-objective evolutionary optimization. *Engineering Optimization, 32*(3), 275-308.

Coello Coello, C. A., & Cruz Cortés, N. (2005, June). Solving multiobjective optimization prob-

lems using an artificial immune system. *Genetic Programming and Evolvable Machines, 6*(2), 163-190.

Coello Coello, C. A., & Toscano Pulido, G. (2001). Multiobjective optimization using a micro-genetic algorithm. In L. Spector, E. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)* (pp. 274-282). San Francisco: Morgan Kaufmann.

Coello Coello, C. A., Toscano Pulido, G., & Salazar Lechuga, M. (2004, June). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation, 8*(3), 256-279.

Coello Coello, C. A., Van Veldhuizen, D. A., & Lamont, G. B. (2002). *Evolutionary algorithms for solving multi-objective problems.* New York: Kluwer Academic.

Collette, Y., & Siarry, P. (2003). *Multiobjective optimization. Principles and case studies.* New York: Springer.

Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. (2001). PESA-II: Region-based selection in evolutionary multiobjective optimization. In L. Spector, E. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)* (pp. 283-290). San Francisco: Morgan Kaufmann.

Corne, D. W., Knowles, J. D., & Oates, M. J. (2000). The Pareto envelope-based selection algorithm for multiobjective optimization. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, et al. (Eds.), *Proceedings of the Parallel Problem Solving from Nature VI Conference*, Paris, France (LNCS 1917, pp. 839-848). Berlin: Springer-Verlag.

Cvetkoviæ, D., & Parmee, I. C. (2002). Preferences and their application in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation, 6*(1), 42-57.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms.* Chichester, UK: John Wiley & Sons.

Deb, K., Agrawal, S., Pratab, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In M. Schoenauer et al. (Eds.), *Proceedings of the Parallel Problem Solving from Nature VI Conference*, Paris, France (LNCS 1917, pp. 849-858). Berlin: Springer-Verlag.

Deb, K., & Goldberg, D. E. (1989, June). An investigation of niche and species formation in genetic function optimization. In J. D. Schaffer (Ed.), *Proceedings of the 3rd International Conference on Genetic Algorithms* (pp. 42-50). San Mateo, CA: Morgan Kaufmann.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182-197.

Doerner, K., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2004). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research, 131*(1-4), 79-99.

Ehrgott, M., Klamroth, K., & Schwehm, C. (2004). An MCDM approach to portfolio optimization. *European Journal of Operational Research, 155*(3), 752-770.

Erickson, M., Mayer, A., & Horn, J. (2001). The Niched Pareto Genetic Algorithm 2 applied to the design of groundwater remediation systems. In E. Zitzler, K. Deb, L. Thiele, C. A.

Coello Coello, & D. Corne (Eds.), *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization* (LNCS 1993, pp. 681-695). Berlin: Springer-Verlag.

Fogel, L. J. (1999). *Artificial intelligence through simulated evolution. Forty years of evolutionary programming.* New York: John Wiley & Sons.

Fonseca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 416-423). San Mateo, CA: Morgan Kaufmann.

Fonseca, C. M., & Fleming, P. J. (1996, September). On the performance assessment and comparison of stochastic multiobjective optimizers. In H.-M. Voigt, W. Ebeling, I. Rechenberg, & H.-P. Schwefel (Eds.), *Parallel problem solving from nature—PPSN IV* (LNCS 1141, pp. 584-593). Berlin: Springer-Verlag.

Fourman, M. P. (1985). Compaction of symbolic layout using genetic algorithms. *Genetic Algorithms and Their Applications: Proceedings of the 1st International Conference on Genetic Algorithms* (pp.141-153). Hillsdale, NJ: Lawrence Erlbaum.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Reading, MA: Addison-Wesley.

Goldberg, D. E., & Richardson, J. (1987). Genetic algorithm with sharing for multimodal function optimization. In J. J. Grefenstette (Ed.), *Genetic Algorithms and Their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms* (pp. 41-49). Hillsdale, NJ: Lawrence Erlbaum.

Horn, J. (1997). Multicriterion decision making. In T. Bäck, D. Fogel, & Z. Michalewicz (Eds.), *Handbook of evolutionary computation* (Vol. 1, pp. F1.9:1-F1.9:15). UK: IOP Publishing Ltd.; Oxford University Press.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994, June). A Niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence* (Vol. 1, pp. 82-87). Piscataway, NJ: IEEE Service Center.

Husbands, P. (1994). Distributed coevolutionary genetic algorithms for multi-criteria and multi-constraint optimization. In T. Fogarty (Ed.), *Evolutionary computing* (pp. 865, 150-165). Berlin: Springer-Verlag.

Jensen, M. T. (2003a, April). Guiding single-objective optimization using multi-objective methods. In G. Raidl, S. Cagnoni, C. G. Johnson, J. J. Romero Cardalda, E. Marchiori, D. W. Corne, et al. (Eds.), *Applications of evolutionary computing. Evoworkshops 2003: Evobio, evocop, evoiasp, evomusart, evorob, and evostim*, Essex, UK (LNCS 2611, pp.199-210). Berlin: Springer-Verlag.

Jensen, M. T. (2003b). Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms. *IEEE Transactions on Evolutionary Computation, 7*(5), 503-515.

Kennedy, J., & Eberhart, R.C. (2001). *Swarm intelligence.* San Francisco: Morgan Kaufmann.

Knowles, J., & Corne, D. (2002, May). On metrics for comparing nondominated sets. *Proceedings of the Congress on Evolutionary Computation* (CEC'2002) (Vol. 1, pp. 711-716). Piscataway, NJ: IEEE Service Center.

Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the

Pareto archived evolution strategy. *Evolutionary Computation, 8*(2), 149-172.

Knowles, J. D., Watson, R. A., & Corne, D. W. (2001). Reducing local optima in single-objective problems by multi-objectivization. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, & D. Corne (Eds.), *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization*, Zurich, Switzerland (LNCS 1993, pp. 268-282). Berlin: Springer-Verlag.

Laumanns, M., Thiele, L., Deb, K., & Zitzler, E. (2002). Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation, 10*(3), 263-282.

Lin, D., Wang, S., & Yan, H. (2001). *A multiobjective genetic algorithm for portfolio selection.* Working Paper, Institute of Systems Science, Academy of Mathematics and Systems Science Chinese Academy of Sciences, Beijing, China.

Morse, J. (1980). Reducing the size of the nondominated set: Pruning by clustering. *Computers and Operations Research, 7*(1-2), 55-66.

Mostaghim, S., Teich, J., & Tyagi, A. (2002, May). Comparison of data structures for storing Pareto-sets in MOEAs. *Proceedings of the Congress on Evolutionary Computation (CEC'2002)* (Vol. 1, pp. 843-848). Piscataway, NJ: IEEE Service Center.

Mukerjee, A., Biswas, R., Deb, K., & Mathur, A.P. (2002). Multi-objective evolutionary algorithms for the risk-return trade-off in bank-load management. *International Transactions in Operational Research, 9*(5), 583-597.

Oei, C. K., Goldberg, D.E., & Chang, S.-J. (1991, December). *Tournament selection, niching, and the preservation of diversity.* Technical Report No. 91011, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, USA.

Osyczka, A. (2002). *Evolutionary algorithms for single and multicriteria design optimization.* Heidelberg, Germany: Physica Verlag.

Osyczka, A., & Kundu, S. (1995). A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural Optimization, 10,* 94-99.

Price, K. V. (1999). An introduction to differential evolution. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimization* (pp. 79-108). London: McGraw-Hill.

Purshouse, R. C. (2003). *On the evolutionary optimization of many objectives.* Unpublished doctoral dissertation, Department of Automatic Control and Systems Engineering, University of Sheffield, UK.

Ruspini, E. H., & Zwir, I. S. (1999). Automated qualitative description of measurements. *Proceedings of the 16th IEEE Instrumentation and Measurement Technology Conference.* Venice: IEEE Press.

Schaffer, J. D. (1984). *Multiple objective optimization with vector evaluated genetic algorithms.* Unpublished doctoral dissertation, Vanderbilt University, USA.

Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. *Genetic Algorithms and Their Applications: Proceedings of the 1st International Conference on Genetic Algorithms* (pp. 93-100). Hillsdale, NJ: Lawrence Erlbaum.

Schaffer, J. D., & Grefenstette, J. J. (1985). Multiobjective learning via genetic algorithms. *Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI-85)* (pp. 593-595). Los Angeles: AAAI.

Schlottmann, F., & Seese, D. (2002, June). Hybrid multi-objective evolutionary computation of constrained downside risk-return efficient sets for credit portfolio. *Proceedings of the 8th International Conference of the Society for Computational Economics. Computing in Economics and Finance,* Aix-en-Provence, France.

Schlottmann, F., & Seese, D. (2004). Financial applications of multi-objective evolutionary algorithms: Recent developments and future research directions. In C. A. Coello Coello & G.B. Lamont (Eds.), *Applications of multi-objective evolutionary algorithms* (pp. 627-652). Singapore: World Scientific.

Shoaf, J. S., & Foster, J. A. (1996). A genetic algorithm solution to the efficient set problem: A technique for portfolio selection based on the Markowitz model. *Proceedings of the Decision Sciences Institute Annual Meeting* (pp. 571-573), Orlando, Florida.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221-248.

Streichert, F., Ulmer, H., & Zell, A. (2004, June). Comparing discrete and continuous genotypes on the constrained portfolio selection problem. In K. D. et al. (Eds.), *Genetic and Evolutionary Computation—GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part II*, Seattle, WA (LNCS 3103, pp. 1239-1250). Berlin: Springer-Verlag.

Syswerda, G., & Palmucci, J. (1991). The application of genetic algorithms to resource scheduling. In R. K. Belew & L. B. Booker (Eds.), *Proceedings of the 4th International Conference on Genetic Algorithms* (pp. 502-508). San Mateo, CA: Morgan Kaufmann.

Tan, K., Khor, E., & Lee, T. (2005). *Multiobjective evolutionary algorithms and applications.* London: Springer-Verlag.

Van Veldhuizen, D. A. (1999). *Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations.* Unpublished doctoral dissertation, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, USA.

Van Veldhuizen, D. A., & Lamont, G. B. (2000). Multiobjective optimization with messy genetic algorithms. *Proceedings of the 2000 ACM Symposium on Applied Computing*, Como, Italy (pp. 470-476). ACM.

Van Veldhuizen, D. A., & Lamont, G. B. (2000, July). On measuring multiobjective evolutionary algorithm performance. *Proceedings of the 2000 Congress on Evolutionary Computation* (Vol. 1, pp. 204-211). Piscataway, NJ: IEEE Service Center.

Vedarajan, G., Chan, L. C., & Goldberg, D. E. (1997, July). Investment portfolio optimization using genetic algorithms. In J. R. Koza (Ed.), *Late Breaking Papers at the Genetic Programming 1997 Conference* (pp. 255-263). Stanford, CA: Stanford Bookstore.

Wienke, P. B., Lucasius, C., & Kateman, G. (1992). Multicriteria target optimization of analytical procedures using a genetic algorithm. *Analytical Chimica Acta, 265*(2), 211-225.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation, 8*(2), 173-195.

Zitzler, E., Laumanns, M., & Thiele, L. (2001, May). *SPEA2: Improving the strength Pareto evolutionary algorithm.* Technical Report No. 103). Computer Engineering and Networks

Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Switzerland.

Zitzler, E., Laumanns, M., Thiele, L., Fonseca, C. M., & Grunert da Fonseca, V. (2002, July). Why quality assessment of multiobjective optimizers is difficult. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002)* (pp. 666-673). San Francisco: Morgan Kaufmann.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257-271.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation, 7*(2), 117-132.

Zwir, I. S., & Ruspini, E. H. (1999). Qualitative object description: Initial reports of the exploration of the frontier. *Proceedings of the Joint Eurofuse—SIC'99 International Conference,* Budapest, Hungary.

Zydallis, J. B., Lamont, G. B., & Veldhuizen, D. A. V. (2000, September). Messy genetic algorithm based multi-objective optimization: A comparative statistical analysis. *Proceedings of the PPSN/SAB Workshop on Multiobjective Problem Solving from Nature (MPSN)*, Paris, France.

Section II

Social Modeling

Chapter VII
Simulation in the Social Sciences

Robert Axelrod
University of Michigan, USA

ABSTRACT

Advancing the state of the art of simulation in the social sciences requires appreciating the unique value of simulation as a third way of doing science, in contrast to both induction and deduction. Simulation can be an effective tool for discovering surprising consequences of simple assumptions. This chapter offers advice for doing simulation research, focusing on the programming of a simulation model, analyzing the results, sharing the results, and replicating other people's simulations. Finally, suggestions are offered for building a community of social scientists who do simulation.

SIMULATION AS A YOUNG FIELD

Simulation is a young and rapidly growing field in the social sciences.[1] As in most young fields, the promise is greater than the proven accomplishments. The purpose of this chapter is to suggest what it will take for the field to become mature so that the potential contribution of simulation to the social sciences can be realized.

One indication of the youth of the field is the extent to which published work in simulation is very widely dispersed. Consider these observations from the *Social Science Citation Index* of 2002.

1. There were 77 articles with "simulation" in the title.[2] Clearly, simulation is an important field. But these 77 articles were scattered among 55 different journals. Moreover, only two of the 55 journals had more than two of these articles. The full set of journals that published articles with "simulation" in the title come from virtually all disciplines of the social sciences, including anthropology, business, economics, human evolution, environmental planning, law, information, organization theory, political science, and public policy. Searching by a keyword in the title is bound to locate only a fraction of the articles using simulation, but the dispersion of these articles does demonstrate one of the great strengths as well as one of the great weaknesses of this young field. The

strength of simulation is applicability in virtually all of the social sciences. The weakness of simulation is that it has little identity as a field in its own right.

2. To take another example, consider the articles published by the 26 authors of a colloquium on 'agent-based modeling' sponsored by the National Academy of Sciences (USA) and held October 4-6, 2001.[3] In 2002 they published 17 articles that were indexed by the *Social Science Citation Index*. These 17 articles were in 13 different journals. In fact, of the 26 authors, only two published in the same journal. While this dispersion shows how diverse the field really is, it also reinforces the earlier observation that simulation in the social sciences has no natural home.
3. As a final way of looking at the issue, consider citations to one of the classics of social science simulation, Thomas Schelling's *Micro Motives and Macrobehavior* (1978). This book was cited in 21 articles in 2002, but these articles were maximally dispersed among 21 different journals.

In sum, works using social science simulation, works by social scientists interested in simulation, and works citing social science simulation are all very widely dispersed throughout the journals. There is not yet as much concentration of articles in specialist journals as there is in other interdisciplinary fields such as the theory of games or the study of China.[4]

This chapter is organized as follows. The next section discusses the variety of purposes that simulation can serve, giving special emphasis to the discovery of new principles and relationships. After this, advice is offered for how to do research with simulation. Topics include programming a simulation model, analyzing the results, sharing the results with others, and replicating agent-based models. The final section suggests how to advance the art of simulation by fostering a community of social scientists (and others) who use computer simulation in their research.

THE VALUE OF SIMULATION

Let us begin with a definition of simulation: "Simulation means driving a model of a system with suitable inputs and observing the corresponding outputs"(Bratley, Fox, & Schrage 1987, p. ix). While this definition is useful, it does not suggest the diverse purposes to which simulation can be put. These purposes include: prediction, performance, training, entertainment, education, proof, and discovery.

1. **Prediction:** Simulation is able to take complicated inputs, process them by taking hypothesized mechanisms into account, and then generate their consequences as predictions. For example, if the goal is to predict interest rates in the economy three months into the future, simulation can be the best available technique.
2. **Performance:** Simulation can also be used to perform certain tasks. This is typically the domain of artificial intelligence. Tasks to be performed include medical diagnosis, speech recognition, and function optimization. To the extent that the artificial intelligence techniques mimic the way humans deal with these same tasks, the artificial intelligence method can be thought of as simulation of human perception, decision making, or social interaction. To the extent that the artificial intelligence techniques exploit the special strengths of digital computers, simulations of task environments can also help design new techniques.

3. **Training:** Many of the earliest and most successful simulation systems were designed to train people by providing a reasonably accurate and dynamic interactive representation of a given environment. Flight simulators for pilots are an important example of the use of simulation for training.
4. **Entertainment:** From training, it is only a small step to entertainment. Flight simulations on personal computers are fun. So are simulations of completely imaginary worlds.
5. **Education:** From training and entertainment, it is only another small step to the use of simulation for education. A good example is the computer game SimCity. SimCity is an interactive simulation allowing the user to experiment with a hypothetical city by changing many variables, such as tax rates and zoning policy. For educational purposes, a simulation need not be rich enough to suggest a complete real or imaginary world. The main use of simulation in education is to allow the users to learn relationships and principles for themselves.
6. **Proof:** Simulation can be used to provide an existence proof. For example, Conway's *Game of Life* (Poundstone, 1985) demonstrates that extremely complex behavior can result from very simple rules.
7. **Discovery:** As a scientific methodology, simulation's value lies principally in prediction, proof, and discovery. Using simulation for prediction can help validate or improve the model upon which the simulation is based. Prediction is the use that most people think of when they consider simulation as a scientific technique. But the use of simulation for the discovery of new relationships and principles is at least important as proof or prediction. In the social sciences, in particular, even highly complicated simulation models can rarely prove completely accurate. Physicists have accurate simulations of the motion of electrons and planets, but social scientists are not as successful in accurately simulating the movement of workers or armies. Nevertheless, social scientists have been quite successful in using simulation to discover important relationships and principles from very simple models. Indeed, as discussed below, the simpler the model, the easier it may be to discover and understand the subtle effects of its hypothesized mechanisms.

Schelling's (1974, 1978) simulation of residential tipping provides a good example of a simple model that provides an important insight into a general process. The model assumes that a family will move only if more than one-third of its immediate neighbors are of a different type (e.g., race or ethnicity). The result is that very segregated neighborhoods form, even though everyone is initially placed at random and everyone is somewhat tolerant.

To appreciate the value of simulation as a research methodology, it pays to think of it as a new way of conducting scientific research. Simulation as a way of doing science can be contrasted with the two standard methods of induction and deduction. *Induction* is the discovery of patterns in empirical data.[5] For example, in the social sciences, induction is widely used in the analysis of opinion surveys and the macro-economic data. *Deduction,* on the other hand, involves specifying a set of axioms and proving consequences that can be derived from those assumptions. The discovery of equilibrium results in game theory using rational choice axioms is a good example of deduction.

Simulation is a third way of doing science. Like deduction, it starts with a set of explicit assumptions. But unlike deduction, it does not

prove theorems. Instead, a simulation generates data that can be analyzed inductively. Unlike typical induction, however, the simulated data comes from a rigorously specified set of rules rather than direct measurement of the real world. While induction can be used to find patterns in data, and deduction can be used to find consequences of assumptions, simulation modeling can be used as an aid intuition.

Simulation is a way of doing thought experiments. While the assumptions may be simple, the consequences may not be at all obvious. The large-scale effects of locally interacting agents are called "emergent properties" of the system. Emergent properties are often surprising because it can be hard to anticipate the full consequences of even simple forms of interaction.[6]

There are some models, however, in which emergent properties can be formally deduced. Good examples include the neo-classical economic models in which rational agents operating under powerful assumptions about the availability of information and the capability to optimize can achieve an efficient reallocation of resources among themselves through costless trading. But when the agents use adaptive rather than optimizing strategies, deducing the consequences is often impossible; simulation becomes necessary.

Throughout the social sciences today, the dominant form of modeling is based upon the rational choice paradigm. Game theory, in particular, is typically based upon the assumption of rational choice. In my view, the reason for the dominance of the rational choice approach is not that scholars think it is realistic. Nor is game theory used solely because it offers good advice to a decision maker, since its unrealistic assumptions undermine much of its value as a basis for advice. The real advantage of the rational choice assumption is that it often allows deduction.

The main alternative to the assumption of rational choice is some form of adaptive behavior. The adaptation may be at the individual level through learning, or it may be at the population level through differential survival and reproduction of the more successful individuals. Either way, the consequences of adaptive processes are often very hard to deduce when there are many interacting agents following rules that have non-linear effects. Thus, simulation is often the only viable way to study populations of agents who are adaptive rather than fully rational. While people may try to be rational, they can rarely meet the requirement of information or foresight that rational models impose (Simon, 1955; March, 1978). One of the main advantages of simulation is that it allows the analysis of adaptive as well as rational agents.

An important type of simulation in the social sciences is "agent-based modeling." This type of simulation is characterized by the existence of many agents who interact with each other with little or no central direction. The emergent properties of an agent-based model are then the result of "bottom-up" processes, rather than "top-down" direction.

Although agent-based modeling employs simulation, it does not necessarily aim to provide an accurate representation of a particular empirical application. Instead, the goal of agent-based modeling is to enrich our understanding of fundamental processes that may appear in a variety of applications. This requires adhering to the KISS principle, which stands for the army slogan "keep it simple, stupid." The KISS principle is vital because of the character of the research community. Both the researcher and the audience have limited cognitive ability. When a surprising result occurs, it is very helpful to be confident that one can understand everything that went into the model. Simplicity is also helpful in giving other researchers a realistic

chance of replicating one's model and extending the work in new directions. The point is that while the topic being investigated may be complicated, the assumptions underlying the agent-based model should be simple. The complexity of agent-based modeling should be in the simulated results, not in the assumptions of the model.

As pointed out earlier, there are other uses of computer simulation in which the faithful reproduction of a particular setting is important. A simulation of the economy aimed at predicting interest rates three months into the future needs to be as accurate as possible. For this purpose, the assumptions that go into the model may need to be quite complicated. Likewise, if a simulation is used to train the crew of a supertanker or to develop tactics for a new fighter aircraft, accuracy is important and simplicity of the model is not. But if the goal is to deepen our understanding of some fundamental process, then simplicity of the assumptions is important and realistic representation of all the details of a particular setting is not.

DOING SIMULATION RESEARCH

In order to advance the art of simulation in the social sciences, it is necessary to do more than consider the purpose of simulation. It is also necessary to be more self-conscious about the process of doing the research itself. To do so requires looking at three specific aspects of the research process which take place once the conceptual model is developed: the programming of the model, the analysis of the data, and the sharing of the results.

Programming a Simulation Model[7]

The first question people usually ask about programming a simulation model is: "What language should I use?" For experienced programmers, I recommend Java for two reasons. First, it can be run on almost any computer. Second, software packages are available in Java, which are designed to assist simulation.[8] For beginning programmers, I recommend Visual Basic, which is included in the Excel spreadsheet application of Microsoft's Office software package.

The programming of a simulation model should achieve three goals: validity, usability, and extendibility. The goal of validity is for the program to correctly implement the model. This kind of validity is called "internal validity."

Whether or not the model itself is an accurate representation of the real world is another kind of validity that is not considered here. Achieving internal validity is harder than it might seem. The program is knowing whether an unexpected result is a reflection of a mistake in the programming or a surprising consequence of the model itself. For example, in one of my own models, a result was so counterintuitive that I had to spend several weeks to determine whether this result was a consequence of the model or due to a bug in the program (Axelrod, 1997a). As is often the case, confirming that the model was correctly programmed was substantially more work than programming the model in the first place.

The goal of usability is to allow the researcher and those who follow to run the program, interpret its output, and understand how it works. Modeling typically generates a whole series of programs, each version differing from the others in a variety of ways. Versions can differ, for example, in which data is produced, which parameters are adjustable, and even the rules governing agent behavior. Keeping track of all this is not trivial, especially when one tries to compare new results with output of an earlier version of the program to determine exactly what might account for the differences.

The goal of extendibility is to allow a future user to adapt the program for new uses. For example, after writing a paper using the model, the researcher might want to respond to a question about what would happen if a new feature were added. In addition, another researcher might someday want to modify the program to try out a new variant of the model. A program is much more likely to be extendible if it is written and documented with this goal in mind.

Analyzing the Results

Simulation typically generates huge amounts of data. In fact, one of the advantages of simulation is that if there is not enough data, one can always run the simulation again and get some more! Moreover, there are no messy problems of missing data or uncontrolled variables as there are in experimental or observational studies.

Despite the purity and clarity of simulation data, the analysis poses real challenges. Multiple runs of the same model can differ from each other due to differences in initial conditions and stochastic events. A major challenge is that results are often path-dependent, meaning that history matters. To understand the results often means understanding the details of the history of a given run. There are at least three ways in which history can be described.

1. History can be told as "news," following a chronological order. For example, a simulation of international politics might describe the sequence of key events such as alliances and wars. This is the most straightforward type of storytelling, but often offers little in explanatory power.
2. History can be told from the point of view of a single actor. For example, one could select just one of the actors, and do the equivalent of telling the story of the "Rise and Fall of the Roman Empire." This is often the easiest kind of history to understand, and can be very revealing about the ways in which the model's mechanisms have their effects over time.
3. History can also be told from a global point of view. For example, one would describe the distribution of wealth over time to analyze the extent of inequality among the agents. Although the global point of view is often the best for seeing large-scale patterns, the more detailed histories are often needed to determine the explanation for these large patterns.

While the description of data as history is important for discovering and explaining patterns in a particular simulation run, the analysis of simulations all too often stops there. Since virtually all social science simulations include some random elements in their initial conditions and in the operation of their mechanisms for change, the analysis of a single run can be misleading. In order to determine whether the conclusions from a given run are typical, it is necessary to do several dozen simulation runs using identical parameters (using different random number seeds) to determine just which results are typical and which are unusual. While it may be sufficient to describe detailed history from a single run, it is also necessary to do statistical analysis of a whole set of runs to determine whether the inferences being drawn from the illustrative history are really well founded. The ability to do this is yet one more advantage of simulation: the researcher can rerun history to see whether particular patterns observed in a single history are idiosyncratic or typical.

Using simulation, one can do even more than compare multiple histories generated from identical parameters. One can also systematically study the affects of changing the parameters. For example, the agents can be given either equal or unequal initial endowments of

wealth to see what difference this makes over time. Likewise, the differences in mechanisms can be studied by doing systematic comparisons of different versions of the model. For example, in one version agents might interact at random, whereas in another version the agents might be selective in whom they interact with. As in the simple change in parameters, the effects of changes in the mechanisms can be assessed by running controlled experiments with whole sets of simulation runs. Typically, the statistical method for studying the effects of these changes will be regression if the changes are quantitative and analysis of variance if the changes are qualitative. As always in statistical analysis, two questions need to be distinguished and addressed separately: Are the differences statistically significant (meaning not likely to have been caused by chance), and are the differences substantively significant (meaning large enough in magnitude to be important)?

Sharing the Results

After cycling through several iterations of constructing the model, programming the simulation, and doing the data analysis, the final step in the research is sharing the results with others. As in most fields of research, the primary method of sharing research results is through publication, most often in refereed journals or chapter-length reports in edited collections. In the case of social science simulation, there are several limitations with relying on this mode of sharing information. The basic problem is that it is hard to present a social science simulation briefly. There are at least three reasons:

1. Simulation results are typically quite sensitive to the details of the model. Therefore, unless the model is described in great detail, the reader is unable to replicate or even fully understand what was done. Articles and chapters are often just not long enough to present the full details of the model. (The issue of replication will be addressed next.)
2. The analysis of the results often includes some narrative description of histories of one or more runs, and such narrative often takes a good deal of space. While statistical analysis can usually be described quite briefly in numbers, tables, or figures, the presentation of how inferences were drawn from the study of particular histories usually cannot be brief. This is mainly due to the amount of detail required to explain how the model's mechanisms played out in a particular historical context. In addition, the paucity of well-known concepts and techniques for the presentation of historical data in context means that the writer cannot communicate this kind of information very efficiently. Compare this lack of shared concepts with the mature field of hypothesis testing in statistics. The simple phrase "$p < .05$" stands for the sentence: "The probability that this result (or a more extreme result) would have happened by chance is less than 5%." Perhaps over time, the community of social science modelers will develop a collection of standard concepts that can become common knowledge and then be communicated briefly, but this is not true yet.
3. Simulation results often address an interdisciplinary audience. When this is the case, the unspoken assumptions and shorthand terminology that provide shortcuts for every discipline may need to be explicated at length to explain the motivation and premises of the work to a wider audience.
4. Even if the audience is a single discipline, the computer simulations are still new enough in the social sciences that it may be necessary to explain very carefully both the power and the limitations of the

methodology each time a simulation report is published.

Since it is difficult to provide a complete description of a simulation model in an article-length report, other forms of sharing information about a simulation have to be developed. Complete documentation should include the source code for running the model, a full description of the model, how to run the program, and how to understand the output files. An example of such documentation is available for a study of ethnocentrism (see Axelrod, Hammond, & Grafen, 2004). The documentation is at umich.edu/~axe/AHG/main.htm

Replication of Simulations

Three important stages of the research process for doing simulation in the social sciences have been considered so far—namely the programming, analyzing, and sharing computer simulations. All three of these aspects are done for virtually all published simulation models. There is, however, another stage of the research process that is virtually never done, but which needs to be considered. This is replication. The sad fact is that new simulations are produced all the time, but rarely does anyone stop to replicate the results of anyone else's simulation model. Replication is one of the hallmarks of cumulative science. It is needed to confirm whether the claimed results of a given simulation are reliable, in the sense that they can be reproduced by someone starting from scratch. Without this confirmation, it is possible that some published results are simply mistaken due to programming errors, misrepresentation of what was actually simulated, or errors in analyzing or reporting the results. Replication can also be useful for testing the robustness of inferences from models. Finally, replication is needed to determine if one model can subsume another, in the sense that Einstein's treatment of gravity subsumes Newton's.

Rob Axtell, Michael Cohen, Rick Riolo, and I took up the replication challenge with eight published agent-based models (Axtell et al., 1996). With Murphy's Law operating at full strength, we identified replication problems with respect to ambiguity, gaps, and even errors in the published descriptions, as well as subtle differences between how different floating point systems calculated whether or not 9/3 equals 2+1. More important, perhaps, was that we were able to clarify three decreasing levels of replication: "numerical identity" in which the results are reproduced precisely, "distributional equivalence" in which the results cannot be distinguished statistically, and "relational equivalence" in which the qualitative relationships among the variables are reproduced.

CONCLUSION: BUILDING COMMUNITY

This chapter has discussed how to advance the state of the art of simulation in the social sciences. It described the unique value of simulation as a third way of doing science, in contrast to both induction and deduction. It then offered advice for doing simulation research, focusing on the programming of a simulation model, analyzing the results, sharing the results with others, and replicating agent-based simulations.

One final theme needs to be addressed, namely the building of a community of social scientists who do simulation. This chapter began with the observation that simulation studies are published in very widely dispersed outlets. This is an indication that social science simulators are only just beginning to build strong institutional links across traditional disciplinary boundaries, even though the work itself is often interdisciplinary in content and methodology.

The question now is what it would take to promote further the growth and success of social science simulation. My answer comes in four parts: methodology, standardization, education, and institution building.

This chapter has already discussed suggestions for progress in methodology. The next step is to begin to develop the internal structure and boundaries of the field. In particular, converging on commonly accepted terminology would be very helpful. A host of terms is now used to describe our field. Examples are artificial society, complex system, agent-based model, multi-agent model, individual-based model, bottom-up model, adaptive system, and the somewhat broader term, computational model. Having commonly accepted distinctions between these terms could certainly help specify and communicate what simulation is about.

Hand-in-hand with developing the terminology, a shared sense of the internal structure and boundaries of the field is needed. For example, simulation in the social sciences might continue to develop primarily within the separate disciplines of economics, political science, sociology, and so forth. There are powerful forces supporting disciplinary research, including the established patterns of professional education, hiring, publication, and promotion. Nevertheless, if simulation is to realize its full potential, there must be substantial interaction across the traditional disciplines.

Progress requires the development of an interdisciplinary community of social scientists who do simulation. Progress also requires the development of an even broader community of researchers from all fields who are interested in the simulation of any kind of system with many agents. Certainly, ecology and evolutionary biology have a great deal to offer for the study of decentralized adaptive systems. Likewise, computer science has recently started to pay a great deal of attention to how large systems of more or less independent artificial agents can work with each other in vast networks. In addition, mathematics has developed some very powerful tools for the analysis of dynamic systems (Flake, 1998). Even the playful field of artificial life offers many insights into the vast potential of complex adaptive systems. Conversely, social scientists have a great deal to offer evolutionary biologists, computer scientists, and others because of our experience in the analysis of social systems with large numbers of interacting agents.

The educating of modelers is typically done within the context of specific disciplines. To help build bridges across disciplines, Leigh Tesfatsion, and I have developed an on-guide for newcomers to agent-based modeling across the social sciences (Axelrod & Tesfatsion, 2006).[9]

As the field of agent-based modeling matures, the value of institutional arrangements increases. Such arrangements include journals devoted to simulation, professional organizations, conference series, funding programs, university courses, review articles, textbooks, and shared standards of research practice.[10] To realize the full potential of computer simulation will require the development of these institutional arrangements for community building. Who should be better able to build new institutions than the researchers who use simulation to study real and potential societies?

ACKNOWLEDGMENT

This is an updated version of an article published in *Journal of the Japanese Society for Management Information Systems, 12*(3). The article was originally published in: Conte, R., Hegselmann, R., & Terna, P. (Eds.). (1997). *Simulating social phenomena* (pp. 21-40). Berlin: Springer-Verlag. Reprinted with permission of the Japanese Society for Management Information Systems and Springer-Verlag.

I am also pleased to acknowledge the help of Ted Belding, Michael Cohen, Rick Riolo, and Hans Christian Siller. For financial assistance, I thank Intel Corporation, the Advanced Project Research Agency through a grant to the Santa Fe Institute, the National Science Foundation, and the University of Michigan LS&A College Enrichment Fund. Several paragraphs of this chapter have been adapted from Axelrod (1997b) and are reprinted with permission of Princeton University Press.

REFERENCES

Axelrod, R. (1987). The evolution of strategies in the iterated Prisoner's Dilemma. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 32-41). London: Pitman; Los Altos: Morgan Kaufman. Reprinted in Axelrod (1997b).

Axelrod, R. (1997a). The dissemination of culture: A model with local convergence and global polarization. *Journal of Conflict Resolution, 41,* 203-26. Reprinted in Axelrod (1997b).

Axelrod, R. (1997b). *The complexity of cooperation: Agent-based models of competition and collaboration.* Princeton, NJ: Princeton University Press.

Axelrod, R., & Tesfatsion, L. (2006). A guide for newcomers to agent-based modeling in the social sciences. In L. Tesfatsion & K. L. Judd (Eds.), *Handbook of computational economics, volume 2: Agent-based computational economics.* Amsterdam: North-Holland. Retrieved from http://www.econ.iastate.edu/tesfatsi/abmread.htm

Axelrod, R., Hammond, R. A., & Grafen, A. (2004). Altruism via kin-selection strategies that rely on arbitrary tags with which they coevolve. *Evolution, 58,* 1833-1838.

Axtell, R., Axelrod, R., Epstein, J., & Cohen, M. D. (1996). Aligning simulation models: A case study and results. *Computational and Mathematical Organization Theory, 1,* 123-141.

Bratley, P., Fox, B., & Schrage, L. (1987). *A guide to simulation* (2nd ed.). New York: Springer-Verlag.

Cohen, M. D., March, J. G., & Olsen, J. (1972). A garbage can theory of organizational choice. *Administrative Science Quarterly, 17,* 1-25.

Cyert, R., & March, J. G. (1963). *A behavioral theory of the firm.* Englewood Cliffs, NJ: Prentice-Hall.

Epstein, J., & Axtell, R. (1996). *Growing artificial societies: Social science from the bottom up.* Washington, DC: Brookings; Cambridge, MA: MIT Press.

Flake, G. W. (1998). *The computational beauty of nature* Cambridge, MA: MIT Press.

March, J. G. (1978). Bounded rationality, ambiguity and the engineering of choice. *Bell Journal of Economics, 9,* 587-608.

Poundstone, W. (1985). *The recursive universe.* Chicago: Contemporary Books.

Riolo, R. (1997). *The effects of tag-mediated selection of partners in evolving populations playing the iterated Prisoner's Dilemma.* Santa Fe Institute Working Paper, 97-02-016.

Schelling, T. (1974). On the ecology of micromotives. In R. Morris (Ed.), *The corporate society* (pp. 19-64; see especially pp. 43-54).

Schelling, T. (1978). *Micromotives and macrobehavior* (see especially pp. 137-155). New York: W. W. Norton.

Simon, H. A. (1955). A behavioral model of rational choice. *Quarterly Journal of Economics, 69,* 99-118.

ENDNOTES

1 While simulation in the social sciences began over four decades ago (e.g., Cyert & March, 1963), only in the last fifteen years has the field begun to grow at a fast pace.

2 This excludes articles on gaming and education, the psychological process of mental simulation, and the use of simulation with human subjects or as a strictly statistical technique.

3 The colloquium was published in the *Proceedings of the National Academy of Sciences, 99*(suppl. 3), 2002. It is available at http://www.pnas.org/content/vol99/issue90003/index.shtml

4 A potential exception is the *Journal of Artificial Societies and Social Simulation*. This is an online journal available at http://jasss.soc.surrey.ac.uk/JASSS.html Unfortunately it is not yet indexed by the *Social Science Citation Index*.

5 Induction as a search for patterns in data should not be confused with mathematical induction, which is a technique for proving theorems.

6 Some complexity theorists consider surprise to be part of the definition of emergence, but this raises the question of surprising to whom?

7 This section is adapted from Axelrod (1997b, pp. 210-211) and is used with permission of Princeton University Press.

8 A good example of such a package is Repast (see http://www.econ.iastate.edu/tesfatsi/repastsg.htm)

9 The online guide is available with live links at http://www.econ.iastate.edu/tesfatsi/abmread.htm

10 For details on all of these, see http://www.econ.iastate.edu/tesfatsi/ace.htm

Chapter VIII
Multi-Agent Systems Research and Social Science Theory Building

Harko Verhagen
Stockholm University, Sweden
Royal Institute of Technology, Sweden

ABSTRACT

This chapter describes the possible relationship between multi-agent systems research and social science research, more particularly sociology. It gives examples of the consequences and possibilities of these relationships, and describes some of the important issues and concepts in each of these areas. It finally points out some future directions for a bi-directional relationship between the social sciences and multi-agent systems research which hopefully will help researchers in both research areas, as well as researchers in management and organization theory.

INTRODUCTION

In the early 1980s, Newell (1982) defined the concept of Agent within computer science in his presidential speech as the first AAAI president, and agents and multi-agent systems have since been an area of interest for computer science students and researchers alike. The late seventies and early eighties of the 20th century are the early years of agent research, still searching for direction and a fitting name. From distributed artificial intelligence, coordinated distributed problem solving, and multi-agent systems (MAS), it is the latter name (and corresponding view on these systems) that survived, even if we in most MAS applications can see the distributed problem-solving paradigm (which had its focus on developing special-purpose systems to solve complex real-world problems).

Already in the early days of multi-agent systems research there have been attempts to and cries for the use of social science theories and concepts (Bond & Gasser, 1988; Conte & Gilbert, 1995; Verhagen & Smit, 1996). As for the social science, there the use of agent-based simulation models (or individual-based models) has been heralded as an instrument to conduct

social experiments using artificial societies not viable in human societies (Axelrod, 1997; Verhagen & Smit, 1997; Brent & Thompson, 1999; Lansing, 2002; Macy & Willer, 2002; Sawyer, 2003, 2004). Multi-agent systems can be seen as a development at least partly based on artificial intelligence, thus this is even a claim within the agent society in the same sense that AI was claimed to be "the" instrument for developing and testing theories of cognition (Simon, 1970). Time will tell if MAS will have more impact than AI has had and if it is indeed possible to build "artificial humans."

In the remainder of this chapter I will focus upon the relation between MAS research and the social sciences, present some recent work on integrating frameworks that might help to overcome some of the difficulties in combining these research areas, and finally present some remaining problems.

MULTI-AGENT SYSTEMS RESEARCH AND THE SOCIAL SCIENCES

The relationship between these two research areas can take three different forms:

1. **The social sciences inform or help MAS research:** The social sciences are used in two different ways in the development of theories and models of multi-agent systems. The first way of use is the "borrowing" of concepts developed in the social sciences, such as coordination, organization, convention, norm, trust, and so on in multi-agent systems research. A second way is to contrast agent theory with social theory, based on the distinctions between artificial systems and humans. For example, humans cannot be programmed such that they never violate a norm or always cooperate, but artificial systems can.
2. **MAS research informs or helps the social sciences:** According to Castelfranchi (1998, 2000), agent theory should also produce theories, models, and experimental, conceptual, and theoretical new instruments, which can be used to revise and develop the social sciences. He summarizes this point by stating that agent theory—and the related area of artificial intelligence—is not just an engineering discipline, but it is also a science and thus should develop theories, concepts, and methods of its own. I will give an overview of MAS research for social scientists, with a focus on MAS-based simulation studies, since I presume these to be the main product of interest in this category for social scientists.
3. **The social sciences and MAS research have a bi-directional relationship:** Bi-directional research can be obtained via multidisciplinary researchers or multidisciplinary research teams. The tight cooperation between MAS research questions and social science is a complicating factor, but may produce interesting and better grounded results for both areas. Few projects have a focus on both relations. A prime example of such work is the different projects within the German Socionics research effort (Malsch, 1998; Kron, 2002; Köhler, Rölke, Moldt, Valk, von Lüde, Langer, et al., 2003; Fischer, Florian, & Malsch, 2005).

The next three sections will describe each of these relationships in more detail.

SOCIAL SCIENCES THEORIES AND CONCEPTS AS AN INPUT TO MAS RESEARCH

Here I will focus mainly upon theories and frameworks from sociology. I will present a

Table 1. Sociological theories schematized (Adapted from Brante, 2001)

Sociological theories	relation	cultural scheme/causal mechanisms	acting unit
international level	relations between nations	dependency theory, globalization theory	nation
inter-institutional level/national	relations between institutions and organizations	social order, historical/evolutionary	institution
institutional level	status, roles, networks, hierarchies	contingency theory, neo-institutionalism	individuals within a social order (mediated interaction)
inter-individual level	face-to-face interaction	symbolic interactionism, conversation analysis	individuals in direct interaction
individual level	intra-individual components	socio-biology, rational choice	

meta-theoretical framework developed by Brante (2001) and the agent model developed by Carley and Newell (1994) to analyze the current use of these theories and concepts in MAS research.

Categorizing Social Theories

In a recent article, Brante (2001) developed a meta-theoretical framework to describe different societal levels and how theories may either explain events at one level or how theories tie two levels together. The basic model of Brante is the causal model (though not in a traditional sense)—that is, what is it the theory tried to explain, and what are its dependent and independent variables. This is combined with the unit of analysis—that is, what is the location of action, and its constraints, what is it the acting unit cannot change, what is its environment, and so on.

The base of scientific work (within any field) is characterized as a quest for causality, in that science seeks to provide causal explanations of effects. In sociology, these effects and causes are located at several levels of aggregation. In most sciences, there are strict descriptions of what counts as causal explanations. In Brante's article the following (relaxed) definition is used: "Sociology seeks to identify *social structures* harboring *causal mechanisms* that generate *empirically observable effects*" (p. 178, italics added).

What differs between different sociological research paradigms is what counts as these three elements. Brante proposes the following matrix (slightly adjusted for my purposes in Table 1 and "translated" to a matrix for psychological theories in Table 2).

Given the type of interaction, process, or acting unit one is interested in, these two tables give the researcher an idea of what the other two components are. It also states what other levels one has to consider as the dynamics that might be invisible for the current level of interest (i.e., the level just below it in the matrix) and the context (or product) of the interactions (the level just above the level of interest). Involving more than these three levels in research will prove difficult; one needs to have theories or concepts that run across several levels in these cases. For example, imagine a researcher interested in issues such as the emergence of war between national states, and who would like to use neural networks in a simulation model. According to the categorization presented, this research plan will prove impossible.

Table 2. Psychological theories categorized (Inspired by Brante, 2001)

Psychological theories	relation	causal mechanisms	acting unit
inter-individual level	other individuals	social psychology, social cognition	individual agent
individual level	memory types, utility, tools	information processing model, rationality, BDI, situated cognition	chunks of information
subconscious level	neurological signals, conscious vs. subconscious, emotions	neurological theories	electrical patterns or subconscious processing

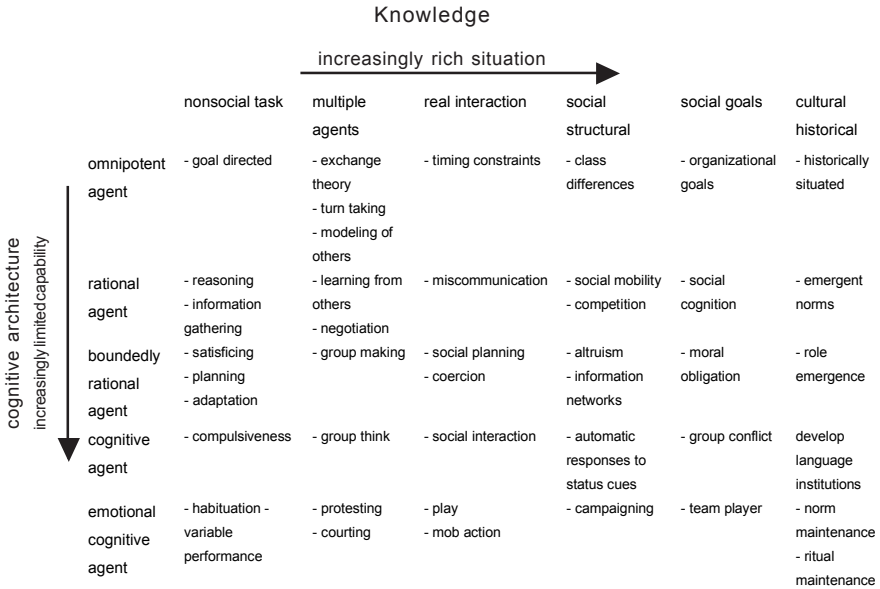
Models of Agency in the Social Sciences

Basically, Carley and Newell (1994) sketch what is necessary for developing an artificial human agent. Different types of agency (and corresponding concepts describing its behavior) are distinguished using two dimensions: the internal architecture of the agent and the amount of knowledge needed to handle the situation (see Figure 1).

The most simple and least human-like agent one can imagine using these dimensions is an agent who is omnipotent (i.e., without any limitations at all on its information processing capabilities) possessing only knowledge about the task it is working on. At the other extreme, we find an agent who is both emotional and cognitive, with knowledge of the task, as well as other agents, interacting in real time within a social structure with social goals and cultural and historical knowledge of that social structure. It is this agent that Carley and Newell label model social agent. Between the omnipotent agent and the model social agent all sorts of agents can be distinguished, which in turn can be linked to different types of social theories.

There is also a trade-off between information processing capabilities and behavior: the more severe the limitations on the information processing capabilities are, the more complex behavior may emerge. For example, the omnipotent agent does not have to gather information, since the agent already knows everything. As for the knowledge dimension, the more

Figure 1. Agent categorization matrix (Adapted from Carley & Newell, 1994)



knowledge an agent has, the more detailed its mental model of the world can be and the more realistic it will be, as compared to the richness of the mental models of humans. With a richer mental model, the agent has a richer repertoire of actions to choose from and its set of goals can be more complex. For instance, if an agent has no knowledge of social structure, the agent's place in the social environment does not have to (and even cannot) be determined. Since the architectural dimensions map nicely to Brante's matrix in the previous section, I will focus on a detailed description of the knowledge dimension.

The authors distinguish six situations with regard to the knowledge dimension, or more accurately the type of information on the situation the agent is in that is used in the decision-making behavior of the agent. The agent can make use of cultural-historical knowledge about the situation it is in. The situation is affected by and a result of a historical process leading to a specific culture, embodied in norms shared among the members of a group. If the agent does not take the cultural-historical knowledge into consideration, it can still be aware of social goals. Apart from the goals related to the task at hand and the goals related to self-maintenance and enhancement, the agent also has goals that have to do with social maintenance. These social goals can be irreconcilable with the goals of the individual or even of the task, thus influencing the problem-solving behavior of the agent. Lifting the constraints of the social goals leaves us with constraints due to the social structure the agent is in. The agent has knowledge of the structure of the group it is in and the position of the agents it interacts with. If the agent has no knowledge of the social structure, it may be engaged in face-to-face interaction. The agents do interact, but the interactions are not directly with individual agents but mediated via the environment.

The interactive situation limits the time an agent has. Absence of such a time limit results in an agent in a multiple agent situation. There are other agents in the environment, with their own goals and capabilities and displaying goal-directed behavior.

The last situation is the non-social task situation. The agent only has knowledge of the task and the task environment. Other agents may exist in that environment but are treated just like inanimate objects.

Models of Agency in Multi-Agent Systems Research

The basic agent definition as sketched out by Wooldridge (2002), for example, states that an agent is a computer system that is situated in some environment and that is capable of autonomous action in this environment in order to meet its design objectives where autonomy means control over behavior and internal state.

Wooldridge (2002) makes a distinction between weak and strong agency. An intelligent agent that is capable of flexible autonomous action displays weak agency. Flexibility builds upon three characteristics, namely reactivity (interaction with the environment), pro-activeness (take initiative regardless of external changes or commands), and social ability (able to interact or cooperate with other agents). A strong agency puts demands on the internals of the agent, with mentalistic properties such as beliefs, desires, intentions (together these three form the famous BDI), rationality, emotions, and so forth.

This last definition shows that the strong agency model combines subconscious properties with individual decision-making theories (without explicitly stating this). Including the social abilities means crossing yet another level. Therefore, within multi-agent research there is a call for integrating three levels that were

distinguished by Brante. If we use the Carley and Newell model to look at MAS research, we can note that omnipotent agents hardly exist; in fact if an agent can be omnipotent, we can do without the concept of agents. Agents are used in software development since knowledge is limited and local autonomy is needed. Also, the single agent paradigm is not really an implementation of agents, at least not of a multi-agent system.

MAS USE IN SOCIAL SCIENCE RESEARCH

Already in the early days of computer development, simulations were used in different research areas. Apart from the use in ballistics calculations, techniques such as individual-based modeling were to some extent used for simulation of biological systems (Barricelli, 1957) and to bridge the gap between the simple and the complex with respect to intelligence (Ashby, 1952). But in general, if computers were used for simulations of complex systems, these simulations would be based on differential equations and focus on results at the aggregate level. These models of, for instance, predator-prey populations could result in fairly accurate models, but were limited in the sense that the models excluded individual behavior and decision making (Brent & Thompson 1999), and were based on homogeneous agents. The development of multi-agent systems offers a possible solution to this problem with its (seemingly) natural mapping onto interacting individuals with incomplete information and capabilities, no global control, decentralized data, asynchronous computing, and allowing heterogeneous agents. MAS simulation models also offer the possibility to study the dynamics of the interaction processes instead of focusing on the (static) results of these processes (Lansing, 2002; Sawyer, 2003). MAS simulations can be seen as equivalent to experimental methods or as theory in themselves (Sawyer, 2004). In the former case, simulations are run to test the predictions of theories, whereas in the latter case simulations in themselves are formal models of theories. Formalizing the ambiguous, natural language-based theories of the social sciences helps to find inconsistencies and other problems, and thus contributes to theory building.

Using MAS for simulation studies as an experimental tool offers great possibilities. Most experiments with human societies are either unethical or even impossible to conduct. Experiments *in silicio* on the other hand are fully possible. These can also blow new life into the ever-present debate in sociology on the micro-macro link (e.g., Alexander, Giesen, Münch, & Smelser, 1987). MAS models mostly focus on the emergence of macro-level properties from the local interaction of adaptive agents that influence one another (Macy & Willer, 2002; Sawyer, 2003), but simulations in computational organization theory (e.g., Prietula, Carley, & Gasser, 1998; Carley & Prietula, 1994b), for example, try to analyze the influence of macro-level phenomena on individuals (Sawyer, 2003).Using MAS models to simulate the bi-directional relation between micro- and macro-level concepts would give one tools to analyze the theoretical consequences of the work done by theorists such as Habermas, Giddens, and Bourdieu to name a few (Sawyer, 2003).

Given the above presentation of Brante, one has to decide on what the micro and macro level are of course. In the case of humans, the macro level could be more than one level up, which means one needs to use and represent intermediary levels such as networks and institutions. These are well developed within the sociological literature. Less well-developed, important issues are the importance of the body, human dialogue and communication, and emotions (although see Camic & Joas (2004), Scheff (1990) and Flam (2000) for some recent work in these

areas), which are, according to Carley and Newell (1994), needed for the model of a social agent (the body category was not included in their analysis though).

SOCIAL SCIENCE AND MAS RESEARCH HAVING A BIDIRECTIONAL RELATIONSHIP

As stated at the beginning of this chapter, this is a quite recent development. The German Socionics project (Malsch, 1998; Kron, 2002; Lüde, et al., 2003; Fischer et al., 2005) is a prime example of this path. The Socionics community builds upon the principle that in each of its projects, social scientists and computer scientists participate and cooperate and started in 1998. Unfortunately, most results published focus on either the first or the second of the above mentioned relations between MAS and social science research. Moreover, most publications with a focus on the social sciences are in German, which limits their accessibility (the interested reader might want to check von Scheve & von Lüde, 2005; Fischer et al., 2005). In one example of research with bi-directional relations between MAS and sociology, I will shortly describe my own work in norm autonomous agents.

Bi-Directional Research Example: Norm Autonomous Agents

In the framework developed in Verhagen (2000), norm autonomous agents are described. In short, these agents are based upon cognitive (or goal autonomous agents as developed by Conte & Castelfranchi, 1995) and are extended with norms. The development of an agent typology up to and including the right for agents to decide on their own norm system is a contribution to MAS research. The agents are part of a normative framework, and at the same time reason about and are able to influence these norms. In this sense norm autonomous agents span the institutional (or even inter-institutional level) where norms get their meaning, the inter-individual level (groups of where norms are produced), and the individual level (where the individual decision making is taking place). These agents choose which goals are legitimate to pursue, based on a given system of norms. The agent has the autonomy of generating its own goals and to choose which it is going to pursue. Besides, the agent is equipped to judge the legitimacy of its own goals and other agents' goals. When a goal conflict arises (not to be confused with interest conflict), the agent may change its norm system, thereby changing priorities of goals, abandoning a goal, changing a goal, generating another goal, and so forth. The reasoning capability of these agents at the level of norms is called normative reasoning. Norm autonomous agents generate norms they can use to evaluate states of the world in terms of whether or not they could be legitimate interests. Legitimacy is a social notion and is in the end determined by the norms of the agent with respect to the agent society it is part of. Norm autonomous agents were used to conduct simulation studies to study the effect of different base models on the spreading and internalization of norms within a community of agents (Verhagen, 2001) contributing to the social sciences.

DISCUSSION

Recent developments in the social sciences (and even the natural sciences) such as the renewed interest in network theory (and the increased possibilities to use ever more powerful computer tools) and the increased focus in MAS research on likewise concepts (such as norms, roles, and institutions)—marked by workshops such as ANIREM2005, Roles2005,

NorMAS2005, and the ESAW and RASTA series of workshops—point to a convergence of MAS research and social science research on issues at the level Brante (2001) calls the institutional level. Even developments in formal logic (e.g., Broersen, Dastani, Hulstijn, & van der Torre, 2002; Boella & van der Torre, 2003) on the integration of the BDI framework with notions such as obligations are a driving force in the process. Remaining issues include that most work within the social sciences is of a descriptive nature, which makes direct application of them in, for example, solving agent coordination issues hard. On the other hand, the solution to agent coordination problems with a focus on an optimal solution points more to an engineering view on agent systems than to MAS research for theory development. The same goes the other way around; most social science research is based upon empirical observations and corresponding statistical treatment (or qualitative concept development) rather than on theory development. Thus a happy marriage between MAS research and the social sciences will be of interest to only a minority of the researchers in both fields, but interesting none the less. The biggest issue is of course the work on theories crossing Brante's borders. An example of this may be work such as Carley and Prietula (1994a) on a meso-theory that bridges the borders between characteristics of the physical, cognitive, and social world in some way reminiscent of Habermas' (1984) description of these three aspects of (human) reality with the connected theoretical concepts and paradigms.

ACKNOWLEDGMENTS

This chapter is based on discussions and cowork with the following authors (in alphabetical order): Guido Boella, Magnus Boman, Michael Masuch, Ruud Smit, and Leon van der Torre.

REFERENCES

Alexander, J. C., Giesen, B., Münch, R., & Smelser, N. J. (Eds.). (1987). *The micro-macro link.* Berkeley, CA: University of California Press.

Ashby, W. R. (1952). *Design for a brain.* New York: John Wiley & Sons.

Axelrod, R. (1997). Advancing the art of simulation in the social sciences. In R. Conte, R. Hegselmann, & P. Terna (Eds.), *Simulating social phenomena* (pp. 21-40). Berlin: Springer-Verlag.

Barbalet, J. (1998). *Emotion, social theory, and social structure: A macrosociological approach.* Cambridge, UK: Cambridge University Press.

Barbalet, J. (Ed.). (2002). *Emotions and sociology.* Oxford, UK: Blackwell Publishing.

Barricelli, N. A. (1957). Symbiogenetic evolution processes realized by artificial methods. *Methodos, 9*(35-36), 143-182.

Boella, G., & van der Torre, L. (2003, July 14-18). Attributing mental attitudes to normative systems. In J. Rosenschein, T. Sandholm, M. Wooldridge, & M. Yokoo (Eds.), *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS'03), Melbourne, Australia (pp. 942-943). New York: ACM Press.

Bond, A. H., & Gasser, L. (1988). An analysis of problems and research in DAI. In A. H. Bond & L. Gasser (Eds.), *Readings in distributed artificial intelligence* (pp. 3-35). San Francisco: Morgan Kaufmann.

Brante, T. (2001). Consequences of realism for sociological theory-building. *Journal for the Theory of Social Behavior, 31*(2), 167-195.

Brent, E., & Thompson, G. A. (1999). Sociology: Modeling social interaction with autonomous agents. *Social Science Computer Review, 17*(3), 313-322.

Broersen, J., Dastani, M., Hulstijn, J., & van der Torre, L. (2002). Goal generation in the BOID architecture. *Cognitive Science Quarterly* (special issue on "Desires, goals, intentions, and values: Computational architectures), *2*(3-4), 428-447.

Camic, C., & Joas, H. (Eds.). (2000). *The dialogical turn: New roles for sociology in the postdisciplinary age.* Latham, MD/Boulder, CO/New York/Toronto/Oxford: Rowman and Littlefield.

Carley, K., & Newell, A. (1994). The nature of the social agent. *Journal of Mathematical Sociology, 19*(4), 221-262.

Carley, K. M., & Prietula, M. (1994a). ACTS theory: Extending the model of bounded rationality. In K.M. Carley & M. Prietula (Eds.), *Computational organization theory*. Hillsdale, NJ: Lawrence Erlbaum.

Carley, K. M., & Prietula, M. (Eds.). (1994b). *Computational organization theory*. Hillsdale, NJ: Lawrence Erlbaum.

Castelfranchi, C. (1998). Modeling social action for AI agents. *Artificial Intelligence, 103,* 157-182.

Castelfranchi, C. (2000). Engineering social order. In A. Omicini, R. Tolksdorf, & F. Zambonelli (Eds.), *Engineering societies in the agents world* (pp. 1-18). Berling: Springer-Verlag.

Conte, R., & Gilbert, N. (1995). Computer simulation for social theory. In N. Gilbert & R. Conte (Eds.), *Artificial societies: The computer simulation of social life* (pp. 1-18). London: UCL Press.

Fischer, K., Florian, M., & Malsch, T. (Eds.). (2005). S*ocionics: Its contributions to the scalability of complex social systems.* Berlin: Springer-Verlag (LNCS/LNAI).

Flam, H. (2000). *The emotional man and the problem of collective action*. Peter Lang: Europäische Verlag der Wissenschaft.

Habermas, J. (1984). *The theory of communicative action. Volume one: Reason and the rationalization of society*. Boston: Beacon Press (trans. McCarthy, originally published as *Theorie des kommunikativen handels,* Frankf.a.M: Suhrkamp,1981).

Ilgen, D. R., & Hulin, C. L. (Eds.). (2000). *Computational modeling of behavior in organizations: The third scientific discipline.* Washington, DC: American Psychological Association.

Köhler, M., Rölke, H., Moldt, D., Valk, R., von Lüde, R., Langer, R., et al. (Eds.) (2003). Sozionik—modellierung soziologischer theorie. Müster: Lit Verlag.

Kron, T. (Ed.). (2002). *Luhmann modelliert. Ansätze zur simulation von kommunikationssystemen.* Opladen: Leske + Budrich.

Lansing, J. S. (2002). "Artificial societies" and the social sciences. *Artificial Life, 8,* 279-292.

Macy, M. W., & Willer, R. (2002). From factors to actors: Computational sociology and agent-based modeling. *Annual Review of Sociology, 28,* 143-166.

Malsch, T. (Ed.). (1998). *Sozionik: Soziologische ansichten über künstliche sozialität.* Berlin: Edition Sigma.

Newell, A. (1982). The knowledge level. *Artificial Intelligence, 18*(1), 87-12.

Prietula, M. J., Carley, K. M., & Gasser, L. (Eds.). (1998). *Simulating organizations: Computational models of institutions and groups.* Cambridge, MA: MIT Press.

Scheff, T. (1990). *Microsociology: Discourse, emotion and social structure.* Chicago: University of Chicago Press.

Sawyer, R. K. (2003). Artificial societies—Multi-agent systems and the micro-macro link in sociological theory. *Sociological Methods & Research, 31*(3), 325-363.

Sawyer, R. K. (2004). Social explanation and computational simulation. *Philosophical Explorations, 7*(3), 219-231.

Simon, H. A. (1970). *The sciences of the artificial.* Cambridge, MA: MIT Press.

von Scheve, C., & von Lüde, R. (2005). Emotion and social structures: Towards an interdisciplinary approach. *Journal for the Theory of Social Behavior, 3*(35), 303-328.

Verhagen, H. (2001). Simulation of the learning of norms. *Social Science Computer Review, 3*(19), 296-306.

Verhagen, H. J. E., & Smit, R. A. (1996). Modeling social agents in a multi-agent world. In W. van de Velde & J.W. Perram (Eds.), *Position papers of MAAMAW 1996,* Technical Report 96-1, Vrije Universiteit Brussel—Artificial Intelligence Laboratory, Belgium.

Verhagen, H. J. E., & Smit, R. A. (1997, September 22-25). Multi-agent systems as simulation tools for social theory testing. *Poster presented at ICCS and SS,* Cartona, Italy. Retrieved June 9, 2006, from http://www.dsv.su.se/~verhaven/research.html.

Wooldridge, M. (2002). *An introduction to multi-agent systems.* New York: John Wiley & Sons.

Chapter IX
A Dynamic Agent-Based Model of Corruption

Rajesh Chakrabarti
Georgia Institute of Technology, USA

ABSTRACT

The author builds an agent-based model wherein the societal corruption level is derived from individual corruption levels optimally chosen by heterogeneous agents with different risk aversion and human capital. The societal corruption level, in turn, affects the risk-return profile of corruption for the individual agents. Simulating a multi-generational economy with heterogeneous agents, the author shows that there are locally stable equilibrium corruption levels with certain socio-economic determinants. However, there are situations when corruption can rise until it stifles all economic activity.

"You live in a society where everybody steals. Do you choose to steal? The probability that you will be caught is low ... and, even if you are caught, the chances of your being punished severely for a crime so common are low. Therefore you too steal. By contrast, if you live in a society where theft is rare, the chances of your being caught and punished are high, so you choose not to steal." (Mauro, 1998, p. 12)

INTRODUCTION

Understanding the dynamics of corruption level in a country is crucial for policy formulation. Can corruption keep rising indefinitely? The Indian political philosopher Kautilya talks about corruption as far back as in the 4th century B.C. It is unlikely that corruption has been rising continuously for over the last two millennia. Are there economy-wide forces that determine the "equilibrium" level of corruption in a country? Is this equilibrium a steady state or does it have cycles? What parameters determine its levels? How does the level of development affect it? Why is corruption more widespread in the developing countries than in the industrial nations? The aim of this chapter is to develop a dynamic equilibrium model of corruption that will help us answer these questions.

Tanzi (1998) and Bardhan (1997) provide fairly comprehensive surveys of the vast literature in the area of corruption. While most theoretical studies of corruption tend to focus on the micro models of the phenomenon studying individual acts of corruption, the empirical papers typically study corruption at the country level. There have been a few macro models of corruption with micro-foundations, like Acemoglu and Verdier (1998) and Ehrlich and Lui (1999), but their implications in a dynamic setting are not obvious.

The central insight of this chapter is that corruption at the societal level is the outcome of individual choice of corruption levels faced with a risk-return trade-off faced by an individual, similar to a portfolio selection problem. The trade-off, in turn, depends on the overall corruption level. The higher the level of corruption, the lower is its risk and rewards. We develop a multi-generation equilibrium model using this notion. Each generation consists of agents varying in their risk-aversion and human capital endowment. Each agent chooses to have an individual level of dishonesty (or individual corruption) based upon his individual risk-aversion, human capital endowment, and his generation's perception of the societal corruption level. These choices give rise to a national level of corruption that determines the perceptions of the next generation. Given the difficulty in arriving at a closed-form solution of the model, simulations are used to arrive at the results.

The next section discusses the issues in the definition and measurement of corruption. The third section lays down the formal model. The fourth section describes the simulations and presents the simulation results. The fifth section concludes with pointers towards future research.

DEFINING AND MEASURING CORRUPTION

Corruption is a slippery concept (see Bardhan, 1997; Tanzi, 1998). At a broad level perhaps it may be defined as Transparency International defines it: "abuse of entrusted power for private profit." As recent corporate scandals like Enron and WorldCom suggest, corruption exists in the private sector, though the public sector gets the maximum blame and scrutiny. For our purposes, a corrupt activity must satisfy three criteria. It must have a positive expected economic value to its perpetrators, since otherwise the perpetrators would have no incentive to be corrupt. It must carry some risk of socio-legal censure (else, irrespective of its ethicality, it must be the *custom* in the society in question and cannot be called *corruption* by the relevant social standards). Finally, it must adversely affect the economy. Corruption could be welfare-enhancing in the presence of sub-optimal laws (see Lui, 1985). However, the empirical evidence convincingly establishes the ill-effects of corruption (see Mauro, 1995; Campos, Lien, & Pradhan, 1999; Tanzi & Davoodi, 1997).[1] In other words, there is a dead-weight loss to society because of corruption (see Schleifer & Vishny, 1993). There are thus both income-reducing and income-redistributing effects of corruption.

A key distinction between a corrupt activity and other illegal activities is that the opportunities for corruption are not equally available to every member of society, but rather are highly correlated with the economic power enjoyed by an individual. This is not the case with other forms of illegal activity which are either equally possible for anyone to perpetrate or depend on factors not so closely related to economic power. The model we build in the following section takes into account this feature.

The risk of corruption itself is a function of the level of corruption in society (see Mauro, 1998). It is hard to catch and convict a corrupt person in a corrupt society. Social tolerance of corruption also grows after it reaches a certain threshold level.

Finally, there is no objective measure of corruption. People—scholars and the laity alike—have to rely on their *perception* of corruption, as do international organizations like Transparency International. Thus the measures of corruption available are, at best, noisy. (This is true for almost all forms of illegal activities (see Sah, 1991).

Given these features, we can measure the level of corruption in a society by using an index between 0 and 1, where 0 corresponds to a complete lack of corruption and 1 implies a debilitating level of corruption—that is, the society produces nothing if corruption level is 1.

In a similar vein, an individual's level of corruption (which we shall refer to as the level of *dishonesty* simply to distinguish from societal *corruption* level) can also be measured by a *dishonesty index* ranging between 0 and 1, where 0 indicates that the individual is completely honest and steals no part of the economic rents available to him or her and 1 means that the entire available rent is appropriated. It is important to note that this index is only the measure of the appropriated rents *relative to* the available rents, given the agent's economic power. Thus a president with a dishonesty index of 0.1 is likely to be appropriating much more economic rent in absolute terms than a clerk with a dishonesty index of 0.9.

A society's corruption level is an average of the dishonesty levels of the individuals. However, an honest pauper cannot undo the sins of a corrupt president. The national corruption index, therefore, should be a weighted average of the dishonesty indices of the individuals, with the weights being the economic power (or the size of available economic rents) enjoyed by the individuals.

The model described in the next section is built with these features in mind. It attempts to link individual choice of corruption levels to societal levels of corruption. All these features give our model a much more realistic setting than in most previous work studying corruption. These gains in generality, however, come with the cost that the model ceases to be analytically tractable. We study the results of our model using the simulation approach in the spirit of the increasingly popular *bottom up* technique of artificial *agent-based modeling* (see Epstein & Axtell, 1996, for an early and well-known example).[2]

THE MODEL

The Static Model

Assume an economy comprising n agents. Each individual i, i=1,…,n , has a certain endowment of human capital, k_i, and an aversion towards risk summarized by the parameter, b_i, in his utility function, $u_i = \hat{y}_i - b_i\sigma_y^2$, where $\hat{y}_i$ is the expected income level and σ_y^2 is the variance of $\hat{y}_i$. An agent is active for only one period, and thus the human capital has to be used in one go. 'Human capital' is defined in its broadest sense here to refer to the entire human contribution to the production process. In this sense it is similar to the quality-adjusted labor input of the neoclassical growth model.

At the aggregate level we think of the total output being the result of two basic inputs—social input S and individual input K. We may think of K as simply the sum total of the individual human capital put to use and S as the institutional set-up—a public good that is essential for creation of value. S may be thought of as the way production and indeed society

itself is organized and governed—subsuming within itself the entire legal and socio-political and economic framework that channels all individual efforts into an orchestrated productive enterprise. In other words, S captures the network externalities of the social structure which makes the economy *greater than the sum of its parts*. So we have the aggregate production function, $Y = SK$.

The parameter S itself is likely to depend on the level of K. As the average level of human capital, or 'knowledge and skills', rise in society, not only do individuals become more productive at the micro level, but through the design of more efficient social systems, raise the level of social input as well. Assuming a linear dependence, then $Y = \alpha K^2$ where $\alpha = S/K$.

Every individual has a *dishonesty index*, p_i, $i=1,...,n$, in the closed interval [0,1]. This index may be thought of as the proportion of available rent that the individual appropriates through corrupt activity. Thus 0 denotes a completely honest person while 1 signifies a person who has stolen all that was possible from his position of economic power. The societal *corruption index*, q is the obtained as

$$q = \frac{1}{K}\sum_{i\leq n} p_i k_i .$$

In the presence of corruption, the efficacy of the social input is reduced as efforts of individuals result in lesser output. In other words, $Y = \{(1-q)S\}K$. This is the output-reducing effect of corruption. Therefore q may be interpreted as the proportional dead-weight loss of output owing to corruption. Corruption also has distributive effects on output. Thus a proportion q of the total output is now distributed as the spoils of corruption, while the remainder $(1-q)$ part of the output goes to the agents as compensation.[3] Thus the corrupt people gain at the expense of the honest. The size of the *corruption pie* in the society then is qY. Looked at from a macro perspective, corrupt individuals are in a contest to grab a part of this corruption pie. This characterization of corruption undoubtedly treats it just like another form of tax (e.g., Schleifer & Vishny, 1993).

The spoils of corruption are not risk-free. We define y_c^i to be the income from corruption of an agent i who is completely dishonest (i.e. $p_i = 1$). This y_c^i, then, is a random variable following a normal distribution.[4] Given that there is likely to be a positive relationship between the human capital of an agent and his expected share of the corruption pie, the mean of the distribution of y_c^i is proportional to the human capital endowment of the agent. It also depends positively on the size of the pie (qY) and negatively on the human capital weighted cumulative efforts of other contenders ($\sum_{j\neq i} p_i k_i$ or, for a large enough population, approximately qK). The variance of the distribution, a measure of the risk of corruption, is positively related to the level of *effective social capital,(1-q)S* and the proportion of national income devoted to anti-corruption vigilance (γ). Also the risk is assumed to increase with the human capital of the individual (k_i), simply because of greater visibility and impact. Anti-corruption programs are more likely to go for the *big fishes* in order to maximize the resulting *catch* in money terms for a given investment in such programs (e.g., the recent convictions of corporate bigwigs like Martha Stewart and Dennis Kozlowski in the United States). Thus a *fully* corrupt agent's income from corruption may be modeled as

$$y_c^i \sim N\left(\frac{k_i}{qK} qY, \gamma(1-q)Sk_i\right) \quad \text{or}$$

$$y_c^i \sim N(Sk_i, (1-q)\gamma Sk_i) .$$

Notice that *ceteris paribus,* both the risk and return of corrupt activity increase with the quality of the institutional set-up, an assumption that matches our intuition when comparing developed nations with developing countries.

An individual can choose his level of corruption or dishonesty level anywhere between 0 and 1. In choosing this dishonesty level p_i, the agent is, in effect, making a classical portfolio decision. The entitlement to his income flow from honest activity $\{(1-q)S\}k_i$ is a risk-less asset, while the income from his corrupt activity y_c is risky. The weight of this risky asset depends not only on the distribution of y_c^i, but also on his level of risk-aversion b_i.

Therefore, the agent chooses his level of dishonesty index p_i so as to maximize his utility function u_i introduced before. Thus we can write down his problem as:

$$\underset{p_i}{Max}\ (1-q)\{(1-q)S\}k_i + p_i k_i\{(1-q)S\} - b_i [\gamma S(1-q)p_i k_i]^2 \tag{1}$$

If we assume that individual agents ignore the impact they have on the societal corruption level, then this problem has a simple solution:

$$p_i = \frac{1}{2b_i\gamma^2 k_i(1-q)S}. \tag{1'}$$

Thus, the individual's dishonesty level is decreasing in his risk-aversion, the proportional expenditure on vigilance, γ, his own human capital level, and the quality of social institutions, and increasing in the societal level of corruption. Also, with a very simple model of risk-averse agents, we have been able to produce the inverse relation between the level of development as reflected in human capital endowment (since S is an increasing linear function of *K*) and corruption—a well-known stylized fact (see Treisman, 2000).

We must note here that the fact that the spoils from corruption are stochastic to the individual agent makes the total claim on output also stochastic, albeit with a much lower variance. There is no guarantee then that the total claims will match up with the output produced. This problem, however, is not central to the discussion here and may be assumed away with the assumption of an external insurer who absorbs the aggregate shock. This assumption is quite realistic in today's world of international capital mobility.

The agents, therefore, make their choice of dishonesty levels based on, inter alia, the societal corruption level. The societal corruption level, in turn, is the result of these choices. Clearly this is a fixed-point problem and it may not be difficult to prove the existence of a rational expectation equilibrium in this set-up. However, it may be closer to reality to assume an adaptive expectation process where agents make their choices on their perception of pre-existing societal corruption. Besides, the relative levels of human capital should be endogenous. This brings us to the multi-period version of the model.

The Multi-Period Model: The Evolution of Corruption

Let us assume an overlapping generation model for the economy with two generations alive at any one point in time, each generation living for two periods. The population is constant and every agent has one offspring. During the first half of his life, an agent accumulates human capital through a process that we shall soon describe and learns about the state of the society by observing the seniors. That is where he makes an assessment of the societal corruption level which he assumes will stay the same

Table 1. Glossary of terms

$\bar{b}$	Mean of the distribution of b, the risk-aversion parameter in the utility function in a generation
b_{range}	Range of b
S	Social capital
α	The ratio of social capital to total human capital
$\bar{k}$	Mean human capital endowment in the first generation
γ	Proportion of national income spent on anti-corruption vigilance
p_i	Individual i's level of dishonesty, $p_i \in [0,1]$
q_t	Level of corruption in society in generation t, $q_t \in [0,1]$
q_{start}	Initial value of q based on which agents in the first generation make their choices
τ	A measure of equality in access to human capital
q_{eqm}	The "equilibrium" (average of last 50 generations) value of q
y_{eqm}	The "equilibrium" (average of last 50 generations) value of y
q_{conv}	The coefficient of variation in q in the last 50 generations
y_{conv}	The coefficient of variation in y in the last 50 generations
$prob_q$	Probability that the slope of the trend line through the last 50 observations of q is zero
$prob_y$	Probability that the slope of the trend line through the last 50 observations of y is zero

when in the second half of his life. This adaptive expectation assumption is more appropriate here for two reasons. Firstly, the exact level of corruption depends on the joint distribution of k and b—too complex to solve for q without assuming a correlation structure. Secondly, people's perception of the social level of corruption is likely to be influenced by early impressions. Besides, the bounded rationality of adaptive expectations has been shown to attain *near optimal* utility levels in several situations (see Akerlof & Yellen, 1985; Jones & Stock, 1987; Naish, 1993).

Since both honest and corrupt incomes are increasing in an agent's level of human capital, agents would like to maximize their human capital. We also assume that the agents care for their children, leading to a positive correlation between a parent's realized income and her offspring's human capital level. However, public schooling and individual differences muddy this relationship. Consequently, the human capital endowment of an agent consists of a mixture of two exogenously bounded distributions: (1) where the relative position of an agent in the distribution of human capital is given by that of his parent's realized income (y_i') in the income distribution of the previous generation, and (2) in the uniform distribution between the bounds. The importance τ is a measure of the effectiveness of public schooling or political institutions to break the monopoly of power. Thus the human capital level is given by:

$$k_i = [k_{\min} + \frac{y_i' - y'_{\min}}{y'_{\max} - y'_{\min}} (k_{\max} - k_{\min})](1-\tau) + \tau Z \quad (2)$$

where $Z \sim U(k_{min}, k_{max})$.

These assumptions complete a fully dynamic model of the society. Note that risk aversion is randomly distributed among agents of every generation. In the next section we study the nature, existence, and stability of equilibria in this model. Table 1 provides a glossary of the terms and symbols used in the model.

Table 2.

Panel A. Parameters varied

Average level of risk aversion, $\bar{b}$	3	3.5	4
Range of risk-aversion, b_{range}	1	1.5	2
Proportion of income spent on vigilance, γ	0.1	0.12	0.14
Mean human capital endowment in the first generation, $\bar{k}$	0.5	0.6	0.7
Equality in access to human capital, τ	0.1	0.5	0.9
Initial value of social corruption, q_{start}	0.1	0.3	0.5

Panel B. Parameters held constant

Population	1,000
Number of generations simulated	100
Number of replications	30
Range of human capital endowment	1
The ratio of social capital to total human capital, α	½

THE SIMULATIONS

The Method and the Parameters

Given the difficulty in obtaining a closed-form solution and proof of the dynamic paths of corruption in the above model, we create an artificial society and simulate the evolution of corruption in the society over several generations.[5] The key questions are as follows:

- **Evolution of Corruption Level:** Runaway corruption leading to total collapse of the system, vanishing corruption, or a more or less steady state level? This dynamic behavior of corruption and also income can be captured by the *extent of convergence* in levels of corruption and income respectively over time.
- **Importance of History:** The role of initial corruption level is particularly important for countries with high current levels of corruption like Kenya or Bangladesh. We study the impact of initial values using the parameter, q_{start}.
- **Effect of Socio-Economic Variables on Corruption:** Essential for combating corruption. We study this through the comparative statics of our model.

Our simulations create an artificial dynamic multi-generation society where the agents behave in accordance with the decision rule laid down in the model. Each generation has the risk-aversion parameter (b_i) distributed randomly over the agents according to a uniform distribution with mean $\bar{b}$ and range b_{range}. The first generation has human capital distributed according to a uniform distribution with mean $\bar{k}$ and range 1. The subsequent generations acquire human capital by the process described above. Each generation has 1,000 agents, and 100 generations are simulated to study the time path of societal corruption (q) and national income (y). The "long-run" values—that is, the average values of the *last* 50 generations of the corruption level q and income y, as well as other measures of their convergence—are noted. This process is then repeated 30 times and average values reported.

This gives us a set of observations for one particular combination of parameter values. It is then repeated to scan a section of the parameter space. The parameter values considered are provided in Table 2.

There are six parameters: average ($\bar{b}$) and range (b_{range}) of risk-aversion, mean human capital ($\bar{k}$), all of the first generation, anti-corruption expense (γ), degree of access to human capital (τ), and initial corruption level (q_{start}). Three values of each parameter are examined. This leads to 729 (=3^6) distinct parameter combinations. The parameter intervals were decided with both realism and span in

mind, though some arbitrariness is unavoidable. For parameters with real-life measurable counterparts, access to human capital, γ is unlikely to ever exceed 10-15%. The values of τ are chosen so that they span most of the unit interval. Ideally, q_{start} should also have done the same, but for our choice of the values of the other parameters, a level of q_{start} above 0.6 tends to lead to hyper-corruption.[6]

Our observation consists of six variables: *steady state* corruption and income levels *(q_{eqm}, y_{eqm}),* degree of convergence (q_{conv}, y_{conv}), and steady state indicators ($prob_q$, $prob_y$). It is, however, worthwhile to note here that this technique can only detect *stable* equilibria. Since total convergence is not achieved in finite time, we study the trend and variation in the last few periods to examine convergence.

In our simulations, we look at the *long-run averages*—average over the last 50 generations—of corruption level q (q_{eqm}) and national income y (y_{eqm}) to ascertain the equilibrium levels. To check convergence we study the coefficient of variation of values in the last 50 generations and the p-value of the hypothesis of zero slope of the regression line through the last 50 values ($prob_i$ where $i = q$ or y). The results are discussed in the following section.

Hyper-Corruption

We refer to the situation where the societal corruption level is 1 and no economic value is created has *hyper-corruption*. Some countries (like Zaire or Kenya) have at times come close to completely dysfunctional economies, owing to crippling levels of corruption. At or near this state of hyper-corruption or kleptocracy, social institutions themselves change and new power structures emerge through events like political disintegration of a nation or a revolution. While our model does not cover such changes, we allow for the possibility of hyper-corruption and identify parameter combinations that might lead to it.

There are, indeed, certain regions of the selected parameter space that lead to hyper-corruption with near certainty. Here we discuss a few of them. For instance, consider keeping initial average risk aversion $\bar{b}$ at 4, its range b_{range} at 1, initial average human capital endowment $\bar{k}$ at 0.5, access to human capital τ at 0.1, and anti-corruption expense γ at 0.1. Any initial corruption level q_{start} above 0.66 is now almost sure to lead to hyper-corruption. Alternatively, combinations like a q_{start} of 0.1, γ below 0.06 *or* γ of 0.1, and $\bar{b}$ below 2 almost surely cause hyper-corruption. The behavior of societal corruption is not smooth near these critical points. For instance, in the first example, a q_{start} of 0.65 leads to equilibrium corruption of only 0.12. In the second case, a γ of 0.07 leads to a long-run value of only 0.25, and in the third case, a $\bar{b}$ value of 2.3 results in a long-run value of 0.25.

The topography of corruption in a segment of our parameter space appears to be a relatively smooth valley (as will be revealed by later results) surrounded by sudden steep peaks. Most real-life societies reside in these valleys which we analyze subsequently. However, the state of hyper-corruption is the social counterpart of a 'black hole', with its deadly circumference of attraction extending in all dimensions. Once countries stray into its threshold, they are sucked in inescapably.

Simulation Results: The Time Path of Corruption and Income

We cover 729 distinct parameter combinations in our simulations. The overall average 'long-run' (last 50 generations' average) value is 0.078. The value itself is of little importance. It is the pattern of the time paths and degree of convergence that are of real interest. The

Table 3. Descriptive statistics

Variable	Mean	Std. Dev.	Min	Max
q_{eqm}	0.07869	0.04211	0.02768	0.23683
y_{eqm}	169251	48904.3	96093.1	238475
q_{conv}	0.02974	0.02216	0.01119	0.13675
y_{conv}	0.02917	0.01925	0.01153	0.10894
$prob_q$	0.81187	0.15651	0.18954	0.99941
$prob_y$	0.82106	0.15078	0.19024	0.99972

range of *long-run* values is fairly large, from a low of 0.028 to a high of about 0.237, certifying the spanning abilities of the chosen section of the parameter space.

To check convergence, we look for the absence of a trend in the last 50 values of q (or y) and the coefficient of variation in these values. The p-values of the hypotheses of flat regression lines through the last 50 values of q and y, $prob_q$, and $prob_y$, measure the probability of absence of a trend. The coefficients of variation q_{conv} and y_{conv} record the degree of convergence.

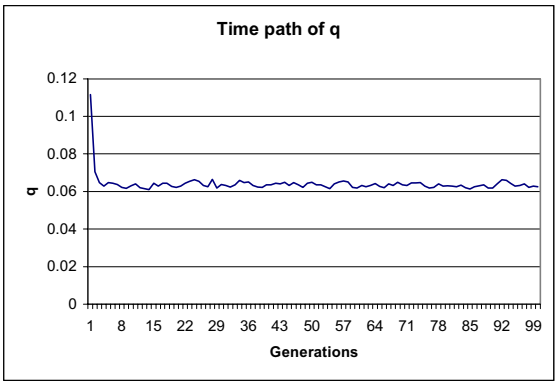
The mean, minimum, and maximum of both $prob_q$ and $prob_y$ are about 0.82, 0.19, and 1 respectively.[7] Only 43 of the 729 values of $prob_q$ lie below 0.5.[8] The coefficient of variation in the last 50 generations has an average of about 0.03 for both q and y, and a maximum of 0.14 for q and 0.11 for y. It appears that the process stabilizes in 100 generations in most cases. The descriptive statistics of the different variables are presented in Table 3. Time paths of corruption and income for a particular parameter combination are shown in Figure 1 for illustration.

Comparative Statics: Effects of Socio-Economic Parameters

Panel A of Table 4 provides average values of the variables of interest for all parameter values considered. Panel B presents the results of regressions of each variable on the different parameters. These regressions do not assume linear causality. They simply constitute a presentation format of the comparative statics results.

The negative effect of average risk aversion on corruption is intuitive as more risk-averse people must be honest according to our model. The negative impact of initial mean human capital level $\bar{k}$ is also expected, since the individual choice of dishonesty level is

Figure 1. An illustrative example of time paths of corruption level and income (all parameters held at their mid-values: one replication only)



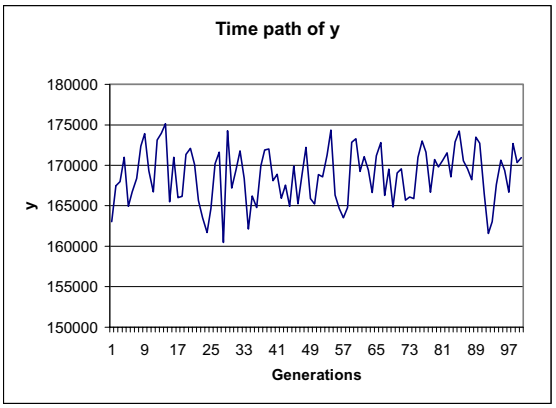


Table 4.

Panel A. Average values of variables by parameter levels

		q_{eqm}	y_{eqm}	q_{conv}	y_{conv}	$prob_q$	$prob_y$
	3	0.095	165957	0.031	0.030	0.806	0.818
$\bar{b}$	**3.5**	0.076	169676	0.030	0.029	0.837	0.844
	4	0.064	172119	0.029	0.028	0.792	0.801
	1	0.076	171018	0.030	0.029	0.812	0.821
b_{range}	**1.5**	0.078	169363	0.030	0.029	0.810	0.819
	2	0.082	167371	0.029	0.029	0.814	0.823
	0.5	0.113	112262	0.042	0.041	0.807	0.821
$\bar{k}$	**0.6**	0.073	164846	0.026	0.025	0.821	0.828
	0.7	0.050	230644	0.022	0.022	0.808	0.814
	0.1	0.079	167006	0.055	0.051	0.718	0.726
τ	**0.5**	0.079	170357	0.020	0.021	0.843	0.855
	0.9	0.078	170389	0.014	0.015	0.874	0.882
	0.1	0.115	163640	0.033	0.032	0.801	0.817
γ	**0.12**	0.072	170238	0.029	0.029	0.831	0.838
	0.14	0.050	173874	0.027	0.027	0.803	0.809
	0.1	0.079	169259	0.030	0.029	0.808	0.819
q_{start}	**0.3**	0.079	169274	0.030	0.029	0.813	0.821
	0.5	0.079	169219	0.030	0.029	0.814	0.823

Panel B. t-statistics of coefficients in regressions of variables on parameters

	$\bar{b}$	b_{range}	$\bar{k}$	τ	γ	q_{start}
q_{eqm}	**-21.79**	**3.70**	**-43.90**	-0.61	**-45.10**	0.03
y_{eqm}	**13.60**	**-8.05**	**261.29**	**7.47**	**22.59**	-0.09
q_{conv}	-1.71	-0.59	**-18.37**	**-36.53**	**-4.94**	0.05
y_{conv}	-1.67	-0.66	**-21.71**	**-40.77**	**-5.04**	0.04
$prob_q$	-1.11	0.11	0.05	**12.01**	0.18	0.48
$prob_y$	-1.35	0.13	-0.55	**12.61**	-0.67	0.36

inversely affected by $\bar{k}$ (through social capital *S*) in equation 1. In similarly higher anti-corruption efforts, γ reduces corruption by making it more risky.

There are, however, a few non-obvious results—for instance, the effect of initial dispersion of risk-aversion b_{range} or τ on the equilibrium values of corruption and income. The simulations reveal that a wider dispersion in risk aversion raises corruption and cuts income in the long-run. The degree of equality to human capital τ, on the other hand, leaves the equilibrium corruption level unaffected, but raises income in the long run and helps immensely in attaining convergence in both corruption and income.

Regarding better vigilance, while *q* expectedly decreases with anti-corruption efforts γ, the net economic gains from greater vigilance is not obvious. The average values in Panel A of Table 4 provide some clues. Raising γ by 2 percentage points first raises the the equilibrium income by a full 4%. The second such raise leads only to a 2.1% increase. Apparently, raising vigilance does pay in terms of higher equilibrium income, but has diminishing marginal returns.[9]

The initial level of corruption does not seem to affect the long-run levels of corruption or income. Long-run levels of corruption in societies are evidently the results of their socio-economic *fundamentals* and not the initial values. In other words, the equilibria are locally stable over the section of the parameter space explored.

CONCLUSION AND FUTURE RESEARCH

Building a multi-generational agent-based model with heterogeneous, risk-averse agents and using simulations, this chapter shows that societies have locally stable equilibrium levels of corruption that depend on a small number of socio-economic parameters. However under certain combinations of these parameter values, it is possible for corruption to go on an ever-increasing trajectory until it stifles all economic activity. The equilibrium levels of corruption depend primarily on the degree of risk aversion, the proportion of national income spent on anti-corruption vigilance, and the level of human capital in the society.

By providing a structure to the phenomenon of corruption and other socio-economic variables, the present study helps us better understand the myriad empirical findings in the recent literature.

Religious traditions and culture may affect morality and risk-taking behavior of agents in a society and the dispersion therein. Legal origin may well affect the industrial organization—α in our model. A higher level of development may be reducing corruption through a higher level of human capital endowment as in our equation 1. Democratic institutions may be working through more egalitarian access to human capital (τ) and structure of the state—federal vs. unitary—may be affecting the overal anti-corruption expenditures (γ) or affecting the industrial organization (α).

Many questions persist. What determines a society's risk aversion? How exactly do religious or cultural factors affect it? What is the role of institutions and how do they relate with the model presented here? What factors, other than human capital, affect *S*?

Societies in transition appear to have more corruption than others—whether a transition from a pastoral/feudal structure to an industrial structure (as in Africa) or that from a socialist system to a market-oriented one (the former republics of the USSR). Is it possible that at a transition, the society's old familiar ways of organizing production give way to a new industrial organization opening up a window of low social capital *S* that unleashes a wave of corruption? Could this lead to hyper-corruption?

Calibrating the model with empirically observed variable value may provide important policy suggestions, allowing governments to actually affect the socio-economic factors driving corruption. Agent-based modeling can help answer several of these questions and poses a promising research agenda in an area important to policymakers, academics, and citizens alike.

ACKNOWLDGMENT

The author gratefully acknowledges helpful remarks and suggestions from Antonio Bernardo, Daniel Treisman, Joe Kairys, Richard Roll, and the participants of the 2000 Computing in Economics and Finance (CEF) Conference at Barcelona. All errors and flaws are the sole responsibility of the author.

REFERENCES

Acemoglu, D., & T. Verdier. (1998). Property rights, corruption and the allocation of talent: A general equilibrium approach. *Economic Journal, 108,* 1381-1402.

Akerlof, G. A., & Yellen, J. L.. (1985). Can small deviations from rationality make significant differences to economic equilibria? *American Economic Review, 75*(4), 708-720.

Bardhan, P. (1987). Corruption and development: A review of issues. *Journal of Economic Literature, XXXV*(September), 1320-1346.

Campos, J. E., Lien D., & Pradhan, S. (1999). The impact of corruption on investment: Predictability matters. *World Development, 27*(6), 1059-1067.

Ehrlich, I., & Lui, F. T. (1999). Bureaucratic corruption and endogenous economic growth. *Journal of Political Economy, 107,* S270-S292.

Epstein, J. M., & Axtell, R. L. (1996). *Growing artificial societies: Social science from the bottom up.* Cambridge, MA: MIT Press and Brookings Press.

Jones, S. R. G., & Stock, J. H. (1987). Demand disturbances and aggregate fluctuations: The implications of near rationality. *Economic Journal, 97*(385), 49-64.

Lui, F. T. (1985). An equilibrium queuing model of bribery. *Journal of Political Economy, 93*(4), 760-781.

Mauro, P. (1995). Corruption and growth. *Quarterly Journal of Economics, 90,* 681-712.

Mauro, P. (1998, March). Corruption: Causes, consequences, and agenda for further research. *Finance and Development,* 11-14.

Naish, H. F. (1993). The near optimality of adaptive expectations. *Journal of Economic Behavior and Organization, 20*(1), 3-22.

Sah, R. K. (1991). Social osmosis and patterns of crime. *The Journal of Political Economy, 99*(6), 1272-1295.

Shleifer, A., & Vishny, R. W. (1993). Corruption. *Quarterly Journal of Economics, 108*(3), 599-617.

Tanzi, V., & Davoodi, H. (1997). *Corruption, public investment and growth.* IMF Working Paper, WP/97/139.

Tanzi, V. (1998). Corruption around the world: Causes, consequences, scope, and cures. *IMF Staff Papers, 45*(4), 559-594.

Treisman, D. (2000). The causes of corruption: A cross-national study. *Journal of Public Economics, 76,* 399-457.

ENDNOTES

1 As the Mission Statement of Transparency International puts it: "Corruption is one of the greatest challenges of the contemporary world. It undermines good government, fundamentally distorts public policy, leads to the misallocation of resources, harms the private sector and private sector development and particularly hurts the poor…" (also see Schleifer & Vishny, 1993).

2 See Leigh Tesfatsion's instructive and useful Web site http://www.econ.iastate.edu/tesfatsi/ace.htm for an up-to-date guide to the growing literature in the area.

3 This is done for simplicity. The two proportions do not need to be the same, but presumably both would be proportional to the level of corruption.

4 It might appear, at first glance, that a person can only have two possibilities—get away with corruption or get caught. However, we must remember that we are not talking about a single act by the agent, but the level of honesty he practices throughout his career. Normal distribution then becomes the obvious choice for characterizing his returns from his whole career or a part of it.

5 The GAUSS code for running the simulations is available on request.

6 Discussed in detail in the next sub-section.

7 The similarities in the distributions of $prob_q$ and $prob_y$ are intuitive, as a trend in q is likely to cause a trend in y.

8 The corresponding number for $prob_y$ is 37.

9 This is without considering other social benefits of reduced corruption.

Chapter X
Human Nature in the Adaptation of Trust

Bart Nooteboom
Tilburg University, The Netherlands

ABSTRACT

This chapter pleads for more inspiration from human nature in agent-based modeling. As an illustration of an effort in that direction, it summarizes and discusses an agent-based model of the build-up and adaptation of trust between multiple producers and suppliers. The central question is whether, and under what conditions, trust and loyalty are viable in markets. While the model incorporates some well-known behavioral phenomena from the trust literature, more extended modeling of human nature is called for. The chapter explores a line of further research on the basis of notions of mental framing and frame switching on the basis of relational signaling, derived from social psychology.

INTRODUCTION

For the object of study, I choose trust for several reasons. First, if anything is human, it is (dis)trust. Second, if anything is subject to adaptation, it is trust in its build-up and breakdown, and as both the basis for a relationship and its outcome. Third, trust forms an important issue in economics, and in behavioral science more widely. Trust is needed to limit transaction costs and costs of contracting and control. In the literature on transaction costs and interfirm relations, there has been a debate whether trust can exist in markets, under pressures of competition. Agent-based simulation seems an appropriate tool for experimentation, to investigate under what conditions trust is viable in markets.

Many attempts have been made at agent-based modeling of trust and related issues. The purpose of trust models varies widely. Generally, they study emergent properties of complex interaction that would be hard or impossible to tackle analytically. Some study the effectiveness of sanctions and/or reputation mechanisms and agencies to support them, for example, in information systems or supply chains (Zacharia et al., 1999; Meijer & Verwaart,

2005; Diekmann & Przepiorka, 2005), or in artificial societies (Younger, 2005). Some study self-organization, for example, in the internalization of externalities in a common pool resource (Pahl-Wostl & Ebenhöh, 2004), the emergence of leadership in open-source communities (Muller, 2003), or the emergence of cooperative social action (Brichoux & Johnson, 2002). Others investigate the working of decision heuristics (Pahl-Wostl & Ebenhöh, 2004; Marsella, Pynadath, & Read, 2004).

The general set-up is that of multiple agents who can profit from each other, but who are uncertain about the quality or competence that is offered, sometimes allowing for multiple dimensions of quality, and dependencies between them (Maximilien & Singh, 2005). Other studies focus on the benevolence or intentions of agents—that is, absence of cheating in free-ridership, defection, or expropriation of knowledge or other resources—and many look at both competence and intentions (Castelfranchi & Falcone, 1999; Pahl-Wostl & Ebenhöh, 2004; Breban, 2002; Muller, 2003; Gans et al., 2001). This is in line with the distinction made in the trust literature between competence trust and intentional trust (e.g., Nooteboom, 2002).

Mostly, agents are oriented only towards their self-interest, such as maximum profit, but some studies also allow for fairness and equity as objectives or dimensions of value (Pahl-Wostl & Ebenhöh, 2004). Mostly, trust is measured as a number between 0 and 1, and, following Gambetta (1988), is often interpreted as a subjective probability that goals will be achieved or no harm will be done. Mostly, conduct is individual, but sometimes allowance is made for coalitions (Breban, 2002).

Few studies of defection explicitly model both sides of the coin: the expectation of defection by others (trust) and one's own inclination to defect (trustworthiness). Also, most studies treat trust as of purely extrinsic value in the achievement of profit, and do not include the possible intrinsic value of trust. Notable exceptions are Pahl-Wostl and Ebenhöh (2004) and Marsella et al. (2004).

Trust is generally updated on the basis of experience: sometimes only one's own experience in interaction, sometimes (also) on the basis of reputation mechanisms, sometimes with the services of some "tracing agency" (Zacharia et al., 1999; Meijer & Verwaart, 2005; Diekman & Przepiorka, 2005). Few studies are based on an explicit inference of competence or intentions, and even fewer studies explicitly model the decision heuristics used. Exceptions here also are Pahl-Wostl and Ebenhöh (2004) and, with great psychological sophistication, Marsella et al. (2004). Those studies will be considered in more detail later. A key question is whether agents have 'a theory of mind' on the basis of which they attribute competencies and intentions to others.

While most studies model trust as adaptive, in the sense that it develops as a function of private or public experience, there is very little study, as far as I know, of adaptiveness of the importance attached to trust relative to profit, and of the adaptiveness of one's own trustworthiness or inclination to defect.

In this chapter, by way of illustration, a model is discussed with some of these features. It focuses on intentional trust, in terms of loyalty or defection, based on private experience (no reputation effects). Trust is adapted on the basis of observed defection, but only with simple reinforcement, without theory of mind and explicit decision heuristics. Next to trust, it includes trustworthiness—that is, inclination to defect. Trustworthiness and the importance attached to trust are both adaptive as a function of experience.

The central purpose of the study is theoretical: to investigate whether the claim of transaction cost economics that trust cannot survive

under competition (Williamson, 1993) is correct. Under what conditions, if at all, are trust and trustworthiness viable in markets where the performance criterion is purely profit? The analysis is conducted in the context of transaction relations between multiple buyers and suppliers, which is the classical setting for the analysis of transaction costs. Thus, the present chapter is related to other sections of the present volume on *industrial structures and innovation* and *supply chain management.*

This chapter proceeds as follows. First, it summarizes this example of an agent-based computational model of trust. Second, it explores possibilities to proceed further in this direction, in an attempt to bring more human nature into the modeling of trust, in the employment of decision making heuristics offered by social psychology.

A MODEL OF ADAPTIVE TRUST

Trust

Trustworthiness may be based on self-interest, but also on benevolence, based on solidarity or loyalty. This is related to two different definitions of trust. According to one definition, trust entails vulnerability of the trustor to possibly harmful actions of the trustee, with the expectation that, for whatever reason, no great harm will be done. The reasons for this expectation may include control or deterrence, in which the trustee refrains from opportunism either because he has no opportunity for it, due to contractual or hierarchical constraints, or no incentives for it, since he is dependent on the trustor or wishes to protect his reputation. For this general notion, which includes safeguards on the basis of control, Nooteboom (2002) proposed not to use the term *trust* but the more general term of *reliance*. Reasons for trustworthiness may also include motives that go beyond (narrow) self-interest, such as the wish to behave appropriately, according to social or moral norms or values, or empathy or identification with the trustor, in combination with feelings of sympathy, friendship, or solidarity (MacAllister, 1995; Lewicki & Bunker, 1996). This is what people mostly mean by the term *trust*.

Is Trust Viable in Markets?

I will summarize and discuss a model of the emergence and adaptation of trust published by Klos and Nooteboom (2001). The purpose of the model was to develop a tool for assessing the viability of trust, in the sense of benevolence, between firms in markets. That is a much-debated issue (for a survey, see Nooteboom, 2002). Economics, in particular transaction cost economics (TCE), doubts the viability of benevolence on the argument that under competition, in markets, firms are under pressure to utilize any opportunity for profit (Williamson, 1993). However, especially under the uncertainty and volatility of innovation, reliance on the basis of control—such as complete contracts, but also reputation mechanisms—is infeasible or unreliable, so that benevolence is especially needed as a basis for governance, as a substitute or complement for necessarily incomplete contracts (Nooteboom, 1999, 2004) and reputation mechanisms. Thus, it is of some theoretical and practical importance to investigate whether, or when, benevolence may be viable. I propose that benevolence, going beyond calculative self-interest, can exist in markets but is nevertheless subject to circumstances, such as pressures of survival, depending on intensity of competition and the achievement of profit (Pettit, 1995), and experience. The purpose of the model is to explore these circumstances.

To serve its purpose, the model should incorporate essential elements of TCE logic. TCE proposes that people organize to reduce transaction costs, depending on conditions of uncertainty and specific investments, which yield switching costs and a resulting risk of *hold-up*. The model employs TCE logic, but also deviates from TCE in two fundamental respects. First, while TCE assumes that optimal forms of organization will arise, yielding maximum efficiency, that is problematic. The making and breaking of relations between multiple agents with adaptive knowledge and preferences may yield complexities and path-dependencies that preclude the achievement of maximum efficiency. Even if all agents can in principle access all relevant partners and have relevant knowledge about them, actual access depends on competition for access, and on unpredictable patterns of making and breaking relations among multiple agents. Second, while TCE assumes that reliable knowledge about loyalty or trustworthiness is impossible (Williamson, 1975), so that opportunism must be assumed, it is postulated here that to some extent trust is feasible, by inference from observed behavior.

The methodology of agent-based computational economics (ACE) is well suited to model complexities of multiple interactions, and to see to what extent theoretical benchmarks of maximum efficiency can in reality be achieved. It enables us to take a process approach to trust (Zand, 1972; Zucker, 1986; Smith Ring & van de Ven, 1994; Gulati, 1995), by modeling the adaptation of trust and trustworthiness in light of experience in interaction.

The Model

In the model, buyers and suppliers are matched on the basis of preferences that are based on both trust and potential profitability, where trust can also have intrinsic value. This matching, depending on the preferences agents make, continues or breaks transaction relations. Trust is based on observed loyalty of partners—that is, absence of switching to a different partner. In line with industrial economics, profit is a function of product differentiation (which increases profit margin), economy of scale from specialization, and learning by cooperation in ongoing relations. Use is made of the notion (from TCE) of specific investments in relationships. Those have value only within the relationship, and thus would have to be made anew when switching to a different partner. Specific investments yield more differentiated products with a higher profit margin. Economy of scale yields an incentive for buyers to switch to a supplier who supplies to multiple buyers, which yields a bias towards opportunism, in breaking relations with smaller suppliers. However, this can only be done for activities that are based on general-purpose assets, not relation-specific investments for specialty products.

The percentage of specialty products is assumed to be equal to the percentage of specific investments as a parameter of the model that can be set. The specialty part, which is relation specific, yields higher profit and is also subject to learning by cooperation, as a function of an ongoing relation. Thereby, it yields switching costs and thus yields a bias towards loyalty.

In sum, the model combines the essential features of TCE: opportunism by defection, specific investments, economy of scale for non-specific investments, and switching costs. However, the model adds the possibility of trust as a determinant of preference, next to potential profit.

In the model, agents are adaptive in three ways. In the preference function, specified in an appendix, the relative weights of potential profit and trust are adaptive as a function of realized profit. In this way, agents can learn to

attach more or less weight to trust, relative to potential profit. Agents adapt their trust in a partner as a function of his loyalty, exhibited by his continuation of the relationship. As a relation lasts, trust increases incrementally, but with decreasing returns, and it drops discontinuously when defection occurs. Agents also adapt their own trustworthiness, modeled as a threshold of exit from a relation, on the basis of realized profit. Agents only defect, in switching, when incremental preference exceeds the threshold. This models the idea that while agents maybe loyal, that has its limits. Thus, agents can learn to become more or less trustworthy in the sense of being loyal.

Note that adaptation of both the weight attached to trust and the threshold of defection occurs on the basis of realized profit. This biases the model in favor of Williamson's (1993) claim that trust cannot survive in markets. In the model, trust and trustworthiness can only emerge when they enhance realized profit. The model allows us to explore under what conditions, in terms of parameter settings, trust and loyalty increase or are stable—that is, when they are conducive to profit and hence viable in markets.

Starting values of agent-related parameters, such as initial trust, threshold of defection, and weight attached to trust, can be set for each agent separately. This allows us to model initially high or low trust societies in setting parameters accordingly for all or most agents, or to model high trust agents in low trust societies, and vice versa, to study whether and when trust is viable or is pushed out by opportunism. Other, non agent-related parameters, such as the percentage of product differentiation and specific assets, strength of economy of scale, strength of learning by cooperation, speed with which trust increases with duration of a relation, number of buyers, number of suppliers, and number of time steps in a run, are fixed per experiment.

In sum, the model is set up to experiment with conditions for trust to grow or decline, as a function of realized profit, depending on trade-offs between advantages of defection (for economy of scale) and advantages of loyalty (in learning by doing in an ongoing relationship). Further technical details of the model are specified in Appendix A.

Simulation Results

Initial expectations were as follows:

- In interactions between multiple, adaptive agents, maximum efficiency is seldom achieved, due to unforeseeable complexities of interaction.
- In conformance with TCE, in the absence of trust, outsourcing occurs only at low levels of asset specificity.
- High trust levels yield higher levels of outsourcing at all levels of asset specificity.
- Under a wide range of parameter settings, high trust levels are sustainable in markets.
- The choice between an opportunistic switching strategy and loyalty depends on the relative strength of scale effects and learning by cooperation.

All these expectations are borne out by recent simulation experiments (Gorobets & Nooteboom, 2005). Of course, simulation is not equivalent to empirical testing. The test is virtual rather than real. It has only been shown that under certain parameter settings, emergent properties of interaction satisfy theoretical expectations. The significance of this depends on how reasonable the assumptions in the model and the parameter settings are considered to be.

The overall outcome is that both trust and opportunism can be profitable, but they go for

different strategies. This suggests that there may be different individual agents or communities, going for different strategies, of switching or of loyalty, which settle down in their own self-sustaining systems. If we compare across the different settings of high, medium, and low initial trust, under different conditions concerning the strength of scale effect relative to learning by cooperation, and concerning initial weight attached to trust and initial thresholds of defection, profit declines more often than it increases, as we go from high to low trust. Further details are given in Appendix B.

The following paradox emerges from the analysis. Potential profit from learning by cooperation is highest for the highest level of product differentiation, but precisely then, when trust is low, buyers prefer to make rather than buy, and thereby forego the opportunities for learning by cooperation. When buyers focus on profitability rather than trust, profit from economy of scale is instantaneous while learning by cooperation is slow, and the potential for economy of scale is low at high levels of differentiation. Thus, under low trust and low weight attached to it, buyers lock themselves out from the advantages of collaboration. When they outsource, it is mostly at low levels of differentiation, when learning by cooperation yields only modest returns, but then they learn to appreciate its accumulation in lasting relationships. They wind up in outsourcing at high differentiation only *by mistake*, then learn to appreciate it, and once learning by doing gets under way, a focus on profit keeps them in the relationship. In time, as profit turns out to be consistent with loyalty and trust, they learn to attach more weight to them.

This illustrates a principle noted before in the trust literature. As a default—that is, a stance taken until reasons for an alternative stance appear (Minsky, 1975)—trust is to be preferred to distrust. Excess trust can be corrected on the basis of experience with untrustworthy partners, while distrust prevents one from engaging in collaboration to learn that partners are in fact trustworthy, if that is the case.

MORE HUMAN NATURE

Discussion of the Model

In the model, human nature is modeled to some extent. Trust is reinforced, incrementally, by observed loyalty, and drops discontinuously in case of observed disloyalty. In evaluating an actual or potential relationship, agents consider both potential profitability and trust they have, and the weight attached to the one relative to the other is adapted on the basis of experience, in the form of past profit. Similar adaptation applies to their own trustworthiness (absence of defection).

However, modeling of cognition and decision making is still primitive in that:

- The rationality of agents is bounded in that they do not take into account opportunities that they have no own experience with, but that are observable. In particular, nontrusting agents who rob themselves of the opportunity to learn that collaboration and loyalty may be profitable do not learn from observing such profit of more trusting competitors.
- In assessing trustworthiness from observed behavior, agents are myopic, looking only at their own experience with the agent. In other words, the model does not contain a reputation mechanism and gossip.
- Adaptation is highly automatic, in combination with random shifts. There is no modeling of processes of inference and decision making, and of the emotions in-

volved. The model does not incorporate a theory of mind.

The first two shortcomings can be repaired without a fundamental shift of model design, by including spillovers of experience in a reputation mechanism, where profits and experiences of loyalty of (some or all) other agents contribute to one's adaptation. The third shortcoming is much more fundamental, since it requires the modeling of social-cognitive processes. Options for doing this are explored in the remainder of this chapter.

More Human Nature from Other Studies

Two studies I found stand out in their dealing with human nature in processes of inference and decision making. Pahl-Wostl and Ebenhöh (2004) emphasize a human approach in terms of decision heuristics and mental frames that agents select from, such as a frame oriented towards cooperation or towards maximizing, as well as switching between frames as a function of experience. They recognize a range of relevant mental categories in cooperative behavior: cooperativeness, fairness (concerning others and concerning me), conformity, reciprocity (positive and negative, in retribution), risk aversion, commitment, and trustworthiness (not being opportunistic). A large and necessary step in the modeling of agents is to equip them with a theory of mind—that is, a basis for inferring competencies and intentions of other agents, as a basis for their decision making. This route of taking decision heuristics known from social psychology is also taken, with impressive sophistication, by Marsella et al. (2004) in their development of virtual agents. This modeling of beliefs, influence, and belief change is intended as a training device, for example, for teachers to learn how to deal with bullies in the classroom.

I am confident that this is the way to go, for some applications at least. Reich (2004) pleads for the use of formal logic in the analysis of reactions to actions, and anticipated reactions to that, on the basis of decision rules. That is no doubt valid, but a socio-cognitive theory is needed to specify those rules. Below, I elaborate some further ideas to proceed along this line of bringing in more human nature, also using insights from social psychology.

Deliberative and Automatic Response

The trust literature recognizes a duality of rational and automatic response. In social psychology, Esser (2005) also recognized rational deliberation and automatic response as two modes of *information processing*. However, the non-deliberative or automatic mode seems to split into two different forms: unemotional routine and emotion-laden impulse, out of faith, friendship, suspicion, in a leap of faith or a plunge of fear.

Emotions, which determine *availability* to the mind, as social psychologists call it, may generate impulsive behavior and may trigger a break of routinized behavior. A question then is whether the latter automatically triggers an automatic response, or whether an emotionally triggered break with routine can lead on to a rational deliberation of response. For that, the emotion would have to be somehow neutralized, controlled, supplemented, or transformed for the sake of deliberation. In the build-up and breakdown of trust, this is of particular importance in view of the indeterminacy of causation. Expectations may be disappointed due to mishaps, lack of competence, or opportunism, and it is often not clear which is the case.

If a relationship has been going well for some time, trust and trustworthiness may be

taken for granted, in routinized behavior. A jolt of fear from exceptional events may be needed to break out of the routine, but in view of the causal ambiguity of what went wrong, one may need to give the trustee the benefit of the doubt, allowing for mishaps or lack of competence, rather than jumping to the conclusion of opportunism. When does this happen and when not?

In the trust literature, it has been proposed that as a relationship develops, at some point reliance (whether it is based on control or trust) is based on cognition—that is, on knowledge concerning the intentions and capabilities of a trustee. Subsequently, actors may develop *empathy*, understanding of how a partner feels and thinks, and next partners may develop *identification*, where they see their fortunes as connected and they start to feel and think alike (McAllister, 1995; Lewicki & Bunker, 1996). As noted by Luhmann (1980), when people start to cooperate, they get the chance to adopt each other's perspectives. In empathy trust may be associated with feelings of solidarity and in identification with feelings of friendship. In going from knowledge-based trust to empathy and identification-based trust, behavior appears to become less deliberative and more automatic, due to both emotions and routinization.

Mental Framing

The question now is how we can further clarify the trust process, in terms of how people think and judge, making and adapting interpretations and choices of action, in a fashion that is amenable, at least in principle, to inclusion in an agent-based model.

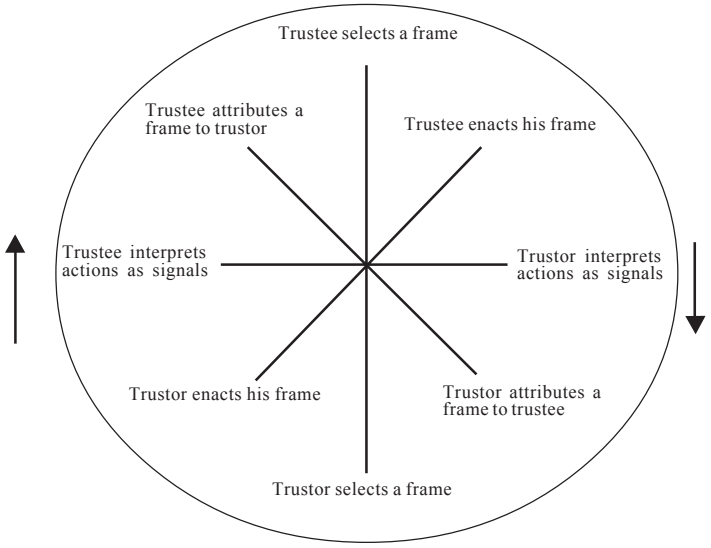
For this, I employ the notion of mental "framing", adopted from sociology and social psychology (Lindenberg, 2000, 2003; Esser, 2005). According to Esser, a mental frame is a "situation-defining orientation" that consists of "two simultaneously occurring selections: the selection of a mental model of the situation on the one hand and that of the mode of information processing in the further selection of action"(Esser, 2005, p. 95, present author's translation from the German). Thus, a mental frame is also associated with action scripts of response appropriate for enacting the frame. For mental frames, Lindenberg (2003) recognized three: "acting appropriately" (AA), also called the "solidarity frame" (Wittek, 1999); "guarding one's resources" (GR), to ensure survival; and a "hedonic frame" (H), where one gives in to temptations for gratifying the senses.

These three frames are adopted here because they align closely with the distinction, in the trust literature, between *benevolence* and *opportunism*, with the latter including both pressures of survival, which seems close to *guarding one's resources*, and vulnerability to temptation when it presents itself, which seems close to the *hedonic frame*. The frames may support or oppose each other, and while at any moment one frame is *salient*, in determining behavior, conditions may trigger a switch to an alternative frame.

If frames serve to both *define a situation* (Esser, 2005) and to guide actions (Lindenberg, 2000, 2003), how are these two combined? As noted by Luhmann (1984, p. 157), in interaction people start building expectations of each others' expectations, on the basis of observed actions. According to the notion of relational signaling (Lindenberg, 2000, 2003; Wittek 1999; Six, 2004), the actions that a trustee undertakes, triggered by a mental frame, in deliberation or automatic response, constitute relational signals that are observed and interpreted by the trustor.

For frame selection I propose the following. The trustee selects a frame, which generates actions that function as signals to the trustor, who on the basis of these signals attributes a salient frame to the trustee and selects a frame

Figure 1. Cycle of frame selection and attribution



for his own response, in the selection of a script, which generates actions taken as signals by the trustee, who attributes a frame to the trustor and selects his own frame. This yields a cycle of selection and attribution, in ongoing interaction, as illustrated in Figure 1. Note that while a trustor (trustee) may select the same frame as the one attributed to the trustee (trustor), in what amounts to a *tit-for-tat response*, this is not necessarily the case. One may persevere in acting benevolently in the face of opportunism, and one may opportunistically exploit the benevolent. Along this cycle, in deliberative response people may try to anticipate effects of actions, their signaling, and the response in attribution, selection, and action. This models Luhmann's notion of the formation of expectations of expectations.

The following questions remain:

1. How, more precisely, do frame selection and attribution take place?
2. How does frame selection lead to action?
3. What determines automatic or deliberative response (in selection and attribution)?

Here, these questions cannot all be answered. For answers, use can be made of decision heuristics recognized in social psychology (for a survey, see Bazerman, 1998; for further elaboration, see Kahneman, Slovic, & Tversky, 1982). Here, I reflect a little further on how frame selection and attribution might be modeled.

Selection and Attribution

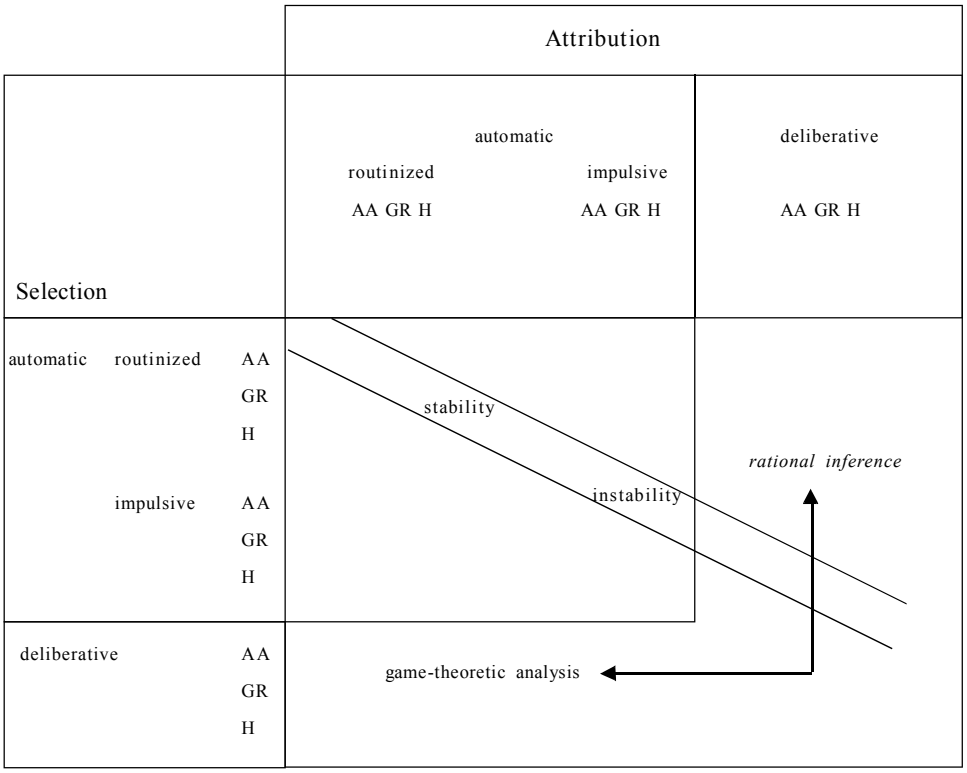
The salience, and hence stability, of a frame and the likelihood of switching to a subsidiary frame depends on whether it is supported by those other frames. For example, acting appropriately in a trustworthy fashion is most stable when it also builds resources and satisfies hedonic drives. One will switch to a frame of

self-interest when temptation or pressure exceeds one's ability to resist. Conversely, one will switch from a self-interested to an other-directed frame when threat or temptation subsides and loyalty assumes more prominence. Decision heuristics from social psychology may be used to understand how this happens (Nooteboom, 2002).

Attribution of a self-interested frame (H, GR) to the trustee seems likely to trigger the defensive selection of a similar frame by the trustor, particularly when the attribution is based on strong triggers (*availability*) of fear of loss, in what amounts to a tit-for-tat strategy. However, that is not necessarily the case, even when the attribution is automatic rather than deliberative. People may control a shock of fear of loss and stick to an other-directed frame (AA), in several ways. Firstly, such a response may be deliberative, in the realization that a misinterpretation may be at play, with a misattribution of opportunism where in fact a mishap or lack of competence may be the cause of failure. However, this may be a psychologically difficult feat to achieve, and one may need the sobering caution from a third party or go-between. See Nooteboom (2002) for an analysis of roles that go-betweens can play in the building and maintenance of trust.

The trustor may respond with a different frame from the one he attributed to the trustee, and both attribution and selection may be automatic, in the two ways of routinized or impulsive response, or deliberative. Three frames for attribution and selection (AA, GR, H), in three modes (routinized, impulsive, deliberative) yield 81 logically possible action-response combinations, as illustrated in Table 1.

Table 1. Attribution and selection



Deliberative attribution entails rational inference of scripts and corresponding frames, and deliberative selection typically entails game-theoretic-type analysis of projected response in attribution to chosen actions. Here, the connection between action scripts and mental frames may be confounded in *interest seeking with guile*: one may choose actions that belong to scripts that enact an AA frame, while in fact one's salient frame is GR.

Impulsive attribution combined with impulsive frame selection will tend to yield instable relations, while routinized attribution in combination with routinized selection, if attributed and selected frames are the same (lie on the diagonal of the table), is likely to result more in stable relations.

The analysis demonstrates the importance of empathy, for correct attribution, on the basis of knowledge of the trustee's idiosyncrasies of conduct and thought, and his strengths and weaknesses, in competence, loyalty, and resistance to temptation and pressures of survival.

For example, one may try to interpret an action as enacting the frame of acting appropriately. For example, the trustee's openness about a mistake is seen as fitting into the set of actions that belong to acting appropriately. In deliberate attribution one carefully tests assumptions concerning the attribution of a frame, considering whether other actions confirm that frame, and whether the action may also fit alternative frames. In routine attribution one attributes without much consideration, according to past anchors, and in impulsive attribution one tries to fit actions into frames that surge to attention as *available* on the basis of fear or other emotion.

From interaction, including the disappointment of expectations, one may learn and innovate in several ways. One may discover new variations upon existing repertoires of actions associated with a frame, a new allocation of actions across mental frames, novel actions, or even novel mental frames. This learning may serve for a better attribution of frames to trustees, and for an extension of one's own repertoires of action and mental frames. Here, even the breach of trust may be positive, as a learning experience, and may be experienced as such.

FURTHER RESEARCH

I have only been able to give a rough sketch of how human nature, as explained in social psychology, may provide a basis for modeling social-cognitive processes in agent-based models in general, and in the build-up and adaptation of trust in particular. Much work remains to be done in translating this into model design.

In particular, we need to fill in the details of how frame attribution and selection take place, in Figure 1 and Table 1. This may be based, in more detail, on decision heuristics identified in social psychology.

However, as recognized also by Marsella et al. (2004), there is the usual trade-off to be considered between detail and management of complexity. While, as Marsella et al. say, complexity may lie in the detail with which agents are modeled, this is feasible and desirable only with very few interacting agents, while in other studies, complexity is emergent from the system of interaction between many agents, more simply modeled.

REFERENCES

Arthur, W. B. (1991). Designing economic agents that act like human agents: A behavioral approach to bounded rationality. *American Economic Review, 8*(1/2), 353-359.

Arthur, W. B. (1992). Designing economic agents that behave like human agents. *Journal of Evolutionary Economics, 3*(1), 1-22.

Bazerman, M. (1998). *Judgment in managerial decision making.* New York: John Wiley & Sons.

Breban, S. (2002, July 15-19). A coalition formation mechanism based on inter-agent trust relationships. *Proceedings of the AAMAS Conference,* Bologna, Italy. Retrieved from Portal.acm.org

Brichoux, D., & Johnson, P. E. (2002). The power of commitment in cooperative social action. *Journal of Artificial Societies and Social Simulation, 5*(3). Retrieved from http://jass.soc.surrey.ac.uk/5/3/1.html/

Castelfranchi, C., & Falcone, R. (1999). Social trust: A cognitive approach. *Proceedings of the CAiSE Conference,* Heidelberg, Germany. Retrieved from Portal.ac.org

Diekmann A., & Przepiorka, W. (2005). *The evolution of trust and reputation: Results from simulation experiments.* Unpublished paper, Department of Humanities, Social and Political Sciences, Swiss Federal Institute of Technology, Switzerland.

Esser, H. (2005). Rationalität und bindung—das modell der frame-selektion und die erklärung des normativen handlens. In M. Held, G. Kubon-Gilke, & R. Sturm (Eds.), *Normative und institutionelle grundfragen der okonomik, jahrbuch 4, reputation und vertrauen* (pp. 85-112). Marburg: Metropolis.

Gambetta, D. (1988). Can we trust trust? In D. Gambetta (Ed.), *Trust; making and breaking of cooperative relations* (pp. 213-237). Oxford, UK: Blackwell.

Gans, G., Jarke, M., Kethers, S., Lakemeyer, G., Ellrich, L., Funken, C., & Meister, M. (2001). Towards (dis)trust-based simulations of agent networks. *Proceedings of the 4th Workshop on Deception, Fraud and Trust in Agent Societies* (pp. 13-25), Montreal. Retrieved from www-kbsg.informatik.rwth-aachen.de/literature/gans2001towards.pdf

Gorobets, A., & Nooteboom, B. (2005). *Adaptive build-up and break-down of trust: An agent-based computational approach.* Paper in review. Retrieved from www.bartnooteboom.nl

Gulati, R. (1995). Does familiarity breed trust? The implications of repeated ties for contractual choice in alliances. *Academy of Management Journal, 30*(1), 85-112.

Khaneman, D., Slovic, P., & Tversky, A. (Eds.). (1982). *Judgment under uncertainty: Heuristics and biases.* Cambridge UK: Cambridge University Press.

Kirman, A. P., & Vriend, N. J. (2001). Evolving market structure: An ACE model of price dispersion and loyalty. *Journal of Economic Dynamics and Control, 25*(3&4), 459-502.

Klos, T. B., & Nooteboom, B. (2001). Agent-based computational transaction cost economics. *Journal of Economic Dynamics and Control, 25,* 503-526.

Lane, D. A. (1993). Artificial worlds and economics, part II. *Journal of Evolutionary Economics, 3*(3), 177-197.

Lewicki, R. J., & Bunker, B. B. (1996). Developing and maintaining trust in work relationships. In R. M. Kramer & T. R. Tyler (Eds.), *Trust in organizations: Frontiers of theory and research* (pp. 114-139). Thousand Oaks, CA: Sage.

Lindenberg, S. (2000). It takes both trust and lack of mistrust: The workings of cooperation and relational signaling in contractual relationships. *Journal of Management and Governance, 4,* 11-33.

Lindenberg, S. (2003). Governance seen from a framing point of view: The employment relationship and relational signaling. In B. Nooteboom & F. E. Six (Eds.), *The trust process: Empirical studies of the determinants and the process of trust development* (pp. 37-57). Cheltenham, UK: Edward Elgar.

Luhmann, N. (1980). *Rechtssociologie 2* (extended ed.). Hamburg: Reinbeck.

Marsella, S. C., Pynadath, D. V., & Read, S. J. (2004). PsychSim: Agent-based modeling of social interactions and influence. *Proceedings of the International Conference on Cognitive Modeling* (pp. 243-248). Mahwah, NJ: Lawrence Erlbaum.

Maximilien, E. M., & Singh, M. P. (2005, July 25-29). Agent-based trust model involving multiple qualities. *Proceedings of the AMAS Conference,* Utrecht, The Netherlands. Retrieved from Portal.acm.org

McAllister, D. J. (1995). Affect- and cognition-based trust as foundations for interpersonal cooperation in organizations. *Academy of Management Journal, 38*(1), 24-59.

Meijer, S., & Vervaart, T. (2005). Feasibility of multi-agent simulation for the trust and tracing game. In M. Ali & F. Esposito (Eds.), *Proceedings of IEA/AIE* (pp. 145-154). Heidelberg: Springer-Verlag.

Minsky, M. (1975). A framework for representing knowledge. In P. H. Winston (Ed.), *The psychology of computer vision.* New York: McGraw-Hill.

Muller, P (2003, November 7-9). On reputation, leadership and communities of practice. *Proceedings of the EAEPE Conference,* Maastricht.

Nooteboom, B. (1999). *Inter-firm alliances: Analysis and design.* London: Routledge.

Nooteboom, B. (2000). *Learning and innovation in organizations and economies.* Oxford, UK: Oxford University Press.

Nooteboom, B. (2002). *Trust: Forms, functions, foundations, failures and figures.* Cheltenham, UK: Edward Elgar.

Nooteboom, B. (2004). *Inter-firm collaboration, learning and networks: An integrated approach.* London: Routledge.

Pahl-Wostl, C., & Ebenhöh, E. (2004, June 14-17). Heuristics to characterize human behavior in agent-based models. In C. Pahl-Wostl, S. Schmidt, A. E. Rizzoli, & A. J. Jakeman (Eds.), *Complexity and Integrated Resources Management, Proceedings of the 2nd Biennial Conference of IEMSS* (pp. 177-184), Osnabrück.

Pettit, P. (1995). The virtual reality of homo economicus. *The Monist, 78*(3), 308-329.

Reich, W. (2004). Reasoning about other agents: A plea for logic-based methods. *Journal of Artificial Societies and Social Simulation, 7*(4). Retrieved from http://jass.soc.surrey.ac.uk/7/4/4.html/

Six, F. (2004). *Trust and trouble: Building interpersonal trust within organizations.* PhD dissertation, Erasmus University, Rotterdam.

Smith Ring, P., & van de Ven, A. (1994). Developmental processes of cooperative interorganizational relationships. *Academy of Management Review, 19*(1), 90-118.

Tversky, A., & Kahneman, D. (1983). Probability, representativeness, and the conjunction fallacy. *Psychological Review, 90*(4), 293-315.

Williamson, O. E. (1975). *Markets and hierarchies.* New York: The Free Press.

Williamson, O. E. (1993). Calculativeness, trust, and economic organization. *Journal of Law & Economics, 36,* 453-486.

Wittek, R. P. M. (1999). *Interdependence and informal control in organizations.* PhD dissertation, University of Groningen, The Netherlands.

Younger, S. (2005). Reciprocity, sanctions, and the development of mutual obligation in egalitarian societies. *Journal of Artificial Societies and Social Simulation.* Retrieved from http://jass.soc.surrey.ac.uk/8/2/9.html/

Zand, D.E. (1972). Trust and managerial problem solving. *Administrative Science Quarterly, 17*(2), 229-239.

Zucker, L. G. (1986). Production of trust: Institutional sources of economic structure. In B., Staw, & L. L. Cummings, *Research in Organizational Behavior, 8,* 53-111.

APPENDIX A: DETAILS OF THE MODEL

Preference and Matching

Preference is specified as follows:

$$\text{score}_{ij} = \text{profitability}_{ij}^{\alpha_i} \cdot \text{trust}_{ij}^{1-\alpha_i} \qquad (1)$$

where: score_{ij} is the score i assigns to j, $\text{profitability}_{ij}$ is the profit i can potentially make 'through' j, trust_{ij} is i's trust in j, and $\alpha_i \in [0, 1]$ is the weight i attaches to profitability relative to trust—that is, the *profit-elasticity* of the scores that i assigns; i may adapt the value of α_i from each timestep to the next.

At each time step, all buyers and suppliers establish a strict preference ranking over all their alternatives. Random draws are used to settle the ranking of alternatives with equal scores. The matching of partners is modeled as follows. On the basis of preferences, buyers are assigned to suppliers or to themselves, respectively. When a buyer is assigned to himself, this means that he makes rather than buys. In addition to a preference ranking, each agent has a *minimum tolerance level* that determines which partners are acceptable. Each agent also has a quota for a maximum number of matches it can be involved in at any one time. A buyer's minimum acceptance level of suppliers is the score that the buyer would attach to himself. Since it is reasonable that he completely trusts himself, trust is set at its maximum of 1, and the role of trust in the score is ignored: $\alpha = 1$. The algorithm used for matching is a modification of Tesfatsion's (1997) deferred choice and refusal (DCR) algorithm, and it proceeds in a finite number of steps, as follows:

1. Each buyer sends a maximum of o_i requests to its most preferred, acceptable suppliers.
2. Each supplier 'provisionally accepts' a maximum of a_j requests from its most preferred buyers and rejects the rest (if any).
3. Each buyer that was rejected in any step fills its quota o_i in the next step by sending requests to the next most preferred, acceptable suppliers that it has not yet sent a request to.
4. Each supplier again provisionally accepts the requests from up to a maximum of a_j most preferred buyers from among newly received and previously provisionally accepted requests, and rejects the rest. As long as one or more buyers have been rejected, the algorithm goes back to step 3.

The algorithm stops if no buyer sends a request that is rejected. All provisionally accepted requests are then definitely accepted.

Trust and Trustworthiness

An agent *i*'s trust in another agent *j* depends on what that trust was at the start of their current relation and on the past duration of their current relation:

$$t_i^j = t_{\mathrm{ini}t,i}^j + (1 - t_{\mathrm{ini}t,i}^j)\left(1 - \frac{1}{fx + 1 - f}\right), \qquad (2)$$

where t_i^j = agent *i*'s trust in agent *j,*

- $t_{\mathrm{ini}t,i}^j$ = agent *i*'s initial trust in agent *j,*
- x = the past duration of the current relation between agents *i* and *j, and*
- f = trustFactor.

This function is taken simply because it yields a curve that increases with decreasing returns as a function of duration x, with 100% trust as the limit and the speed of increase determined by the parameter f .

In addition, there is a base level of trust, which reflects an institutional feature of a society. If an agent *j*, involved in a relation with an agent *i,* breaks their relation, then this is interpreted as opportunistic behavior and *i*'s trust in *j* decreases; in effect, *i*'s trust drops by a percentage of the distance between the current level and the base level of trust; it stays there as *i*'s new initial trust in *j*, $t_{\mathrm{ini}t,i}^j$ until the next time *i* and *j* are matched, after which is starts to increase again for as long as the relation lasts without interruption.

The other side of the coin is, of course, one's own trustworthiness. This is modeled as a threshold τ for defection. One defects only if the advantage over one's current partner exceeds that threshold. It reflects that trustworthiness has its limits, and that trust should recognize this and not become blind (Pettit, 1995; Nooteboom, 2002). The threshold is adaptive as a function of realized profit.

Costs and Profits

Buyers may increase gross profits by selling more differentiated products, and suppliers may reduce costs by generating production efficiencies. There are two sources of production efficiency: economy of scale from a supplier producing for multiple buyers, and learning by cooperation in ongoing production relations. Economy of scale can be reaped only in production with general-purpose assets, and learning by cooperation only in production that is specific for a given buyer, with buyer-specific assets.

We assume a connection between the differentiation of a buyer's product and the specificity of the assets required to produce it. In fact, we assume that the percentage of specific products is equal to the percentage of dedicated assets. This is expressed in a variable $d_i \in [0, 1]$. It determines both the profit the buyer will make when selling his products and the degree to which assets are specific, which determines opportunities for economy of scale and learning by cooperation.

Economy of scale is achieved when a supplier produces for multiple buyers. To the extent that assets are specific, for differentiated products, they cannot be used for production for other buyers. To the extent that products are general purpose—that is, production is not differentiated—assets can be switched to produce for other buyers. In sum, economy of scale, in production for multiple buyers, can only be achieved for the non-differentiated, non-specific part of production, and economy by learning by cooperation can only be achieved for the other, specific part.

Both the scale and learning effects are modeled as follows:

$$y = \max\left(0, 1 - \frac{1}{fx + 1 - f}\right), \qquad (3)$$

where:

- for the scale effect, *f*=scaleFactor, *x* is general-purpose assets of supplier *j* summed over all his buyers and scale efficiency $y = e_{s,j}$; and
- for the learning effect, *f*=learnFactor; *x* is the number of consecutive matches between supplier *j* and buyer *i* and learning efficiency $y = e^{i}_{l,j}$.

Formula (3) expresses decreasing returns for both scale and experience effects. For the scale effect, it shows positive values along the vertical axis only for more than 1 general-purpose asset. This specifies that a supplier can be more scale-efficient than a buyer producing for himself only if the scale at which he produces is larger than the maximum scale at which a buyer might produce for himself. For the learning effect, a supplier's buyer-specific efficiency is 0 in their first transaction, and only starts to increase if the number of transactions is larger than 1. If a relation breaks, the supplier's efficiency due to his experience with the buyer drops to zero. The resulting specification of profit is specified as follows in the next section.

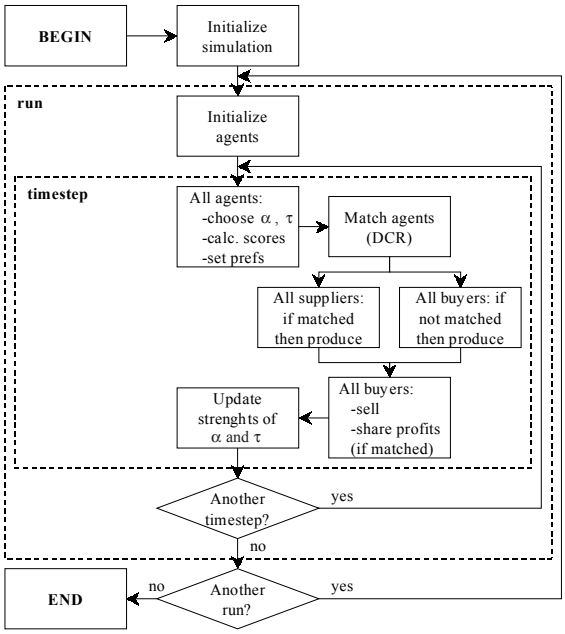
Adaptation

Agents adapt the values for $\alpha \in [0, 1]$ (weight attached to profit relative to trust) and $\tau [0, 0.5]$ (threshold of defection) from one time step to the next, which may lead to changes in the scores they assign to different agents. Here, adaptation takes place on the basis of past, realized profit. While τ could conceivably rise up to 1, a maximum of 0.5 was set because initial simulations showed that otherwise relations would get locked into initial situations with little switching. Note that this biases the model in favor of opportunism. At each step, each agent assigns a strength to each possible value of α and τ. This expresses the agent's confidence in the success of using that particular value. The various strengths always add up to constants C_α and C_τ respectively. At the start of each timestep, the selection of values for a and t is stochastic, with selection probabilities equal to relative strengths—that is, strengths divided by C_α and C_τ, respectively. The strengths of the values that were chosen for α and τ at the start of a particular timestep are updated at the end of that timestep, on the basis of the agent's performance during that timestep, in terms of realized profit: the agent adds the profit obtained during the timestep to the strengths of the values that were used for a or t. After this, all strengths are renormalized to sum to C_α and C_τ again (Arthur, 1993). The idea is that the strength of values that have led to high performance (profit) increases, yielding a higher probability that those values will be selected again. This is a simple model of 'reinforcement learning' (Arthur, 1991, 1993; Kirman & Vriend, 2000; Lane, 1993).

The Algorithm

The algorithm of the simulation is presented by the flowchart in Figure A1. This figure shows how the main loop is executed in a sequence of discrete time steps, called a *run*. Each simulation may be repeated several times as multiple runs, to even out the influence of random draws in the adaptation process. At the beginning of a simulation, starting values are set for certain model parameters. The user is prompted to supply the number of buyers and suppliers, as well as the number of runs, and the number of timesteps in each run. At the start of each run, all agents are initialized, for example with starting values for trust, and selection probabilities for α and τ. In each timestep, before the matching, each agent chooses values for α and τ, calculates scores, and sets preferences. Then the matching algorithm is applied. In the matching, agents may start a relation, continue a

Figure A1. Flowchart of the simulation



relation, and break a relation. A relation is broken if, during the matching, a buyer does not send any more requests to the supplier, or he does, but the supplier rejects them.

After matching, suppliers that are matched to buyers produce and deliver for their buyers, while suppliers that are not matched do nothing. Buyers that are not matched to suppliers produce for themselves (*self-matched*, in *make* rather than *buy*). Afterward, all buyers sell their products on the final-goods market. Profit is shared equally with their supplier, if they have one. Finally, all agents use that profit to update their preference rankings (via α and τ), used as input for the matching algorithm in the next timestep. Across timesteps, realized profits are accumulated for all buyers and suppliers, and all the relevant parameters are tracked.

Note that, by implication, suppliers may fail to produce and then have zero profit. Thus, there is no explicit mechanism of death. However, the procedure may be interpreted as exit of all suppliers with zero profit, accompanied by potential entry on new suppliers, announcing their readiness to give quotes to buyers, up to the maximum number of suppliers specified for the run. Note also that it is conceivable, given the logic of matching, that a supplier breaks with a buyer in his aim to go for a more attractive one, then lose the bidding for that buyer and be left empty-handed. Then, it would be more reasonable for the supplier to first verify his goal attainment before breaking his existing relationship. However, in a large set of simulations, across a wide area of parameter space, this happened only once, at a very high level of opportunism, and it may not be unrealistic that sometimes such error is made, in an over-eagerness to switch to a more attractive partner.

APPENDIX B: DETAILS OF SIMULATION OUTCOMES

High initial trust dictates *buy* relative to *make* for all levels of specific investments. For high specific investments, buyers' maximum profit is almost the same as in the cases of average or low initial trust. Low initial trust imposes *make* relative to *buy,* but buyers' maximum profits for low specific investments are smaller than in the case of high initial trust. Overall, across all parameter settings, profit tends to be higher under high rather than under low trust.

Under medium or low trust, high product differentiation favors *make* relative to *buy* because the switching cost is larger and there is less potential for economy of scale. But if learning by cooperation becomes stronger, relative to scale effects, buyers employ that advantage in a strategy of ongoing relations with suppliers, and achieve a higher profit than when they *make* themselves. If agents put their emphasis on trust (α=0) and loyalty (τ=0.5), buy-

ers get a big advantage in the terms of profit for high specific investments by following the strategy of learning by cooperation. If agents focus on profitability rather than on trust (α=1) and neglect loyalty (opportunistic, τ=0), buyers get some advantage for low specific investments by following the scale strategy (50%) and producing themselves (50%). But if agents are loyalists (τ=0.5), buyers get an advantage for both low and average d by following the scale strategy (60% and 40% respectively) and producing themselves (40% and 60% respectively). Generally, under low trust and low weight attached to trust, buyers forego opportunities for collaboration that may yield learning by cooperation. In sum, high initial trust favors outsourcing (*buy*) and it gives an advantage for all agents in comparison with low initial trust, where buyers get a smaller profit by insourcing (*make*).

In addition to the expected results, the model yields a few unanticipated results. One is that buyers organize closer to maximum possible efficiency for high levels of specific investments/specialization. The reason is that for low levels of specific investments, there is more scope for scale effects, but this is difficult to attain by having suppliers supply to the maximum number of buyers. A strong effect of *learning by cooperation,* a high weight attached to trust, and high loyalty favor the learning by cooperation strategy for high levels of specific investments, while a high weight attached to profit and high loyalty favor the scale strategy for low and average levels of specific investments.

Finally, it is not always the case that a high weight attached to profitability relative to trust favors opportunism. Once a buyer begins to profit from learning by cooperation, an emphasis on profit may also lead to loyalty in an ongoing relationship.

Chapter XI
Cognitively Based Modeling of Scientific Productivity

Isaac Naveh
University of Missouri, USA

Ron Sun
Rensselaer Polytechnic Institute, USA

ABSTRACT

This chapter advocates a cognitively realistic approach to social simulation. based on a model for capturing the growth of academic science. Gilbert's (1997) model, which was equation based, is replaced in this work by an agent-based model, with the cognitive architecture CLARION providing greater cognitive realism. Using this agent model, results comparable to human data are obtained. It is found that while different cognitive settings may affect aggregate productivity of scientific articles, generally they do not lead to different distributions of productivity. It is argued that using more cognitively realistic models in social simulations may lead to novel insights.

SOCIAL SIMULATION AND COGNITIVE MODELING

A significant new trend in social sciences has been that of agent-based social simulation (ABSS). This approach consists of constructing models of societies of artificial agents. Agents are autonomous entities with well-defined rules of behavior. Running such a model entails instantiating a population of agents, allowing the agents to run, and observing the interactions between them. It thus differs from traditional (equation-based) approaches to simulation, where relationships among conceptual entities (e.g., social groups and hierarchies, or markets and taxation systems) are expressed through mathematical equations. Agent-based modeling has a number of advantages over equation-based modeling (Axtell, 2000; Sun 2006).

Interestingly, the evolution of simulation as a means for computational study of societies

has been paralleled by developments in computational modeling at the individual level. Whereas earlier models of cognition tended to emphasize one of the aspects of cognition (for instance, memory or learning), some recent approaches have been more integrative, with a focus on putting the pieces together. The result of this integrative approach is *cognitive architectures,* which are essentially models that capture different aspects of cognition and their interaction. Such models tend to be generic and task independent. Cognitive architectures have greatly grown in expressive power in recent years, and now capture a variety of cognitive phenomena, including various types of memory/representation, modes of learning, and sensory-motor capabilities (e.g., Anderson & Lebiere 1998; Sun 2002).

So far, however, the two fields of social simulation and cognitive architectures have developed in near-isolation from each other (with some exceptions; e.g., Carley & Newell, 1994; Sun 2006; Naveh & Sun, in press). Thus, most of the work in social simulation assumes very rudimentary cognition on the part of the agents. At the same time, while the mechanisms of individual cognition have been the subject of intensive investigation in cognitive science and cognitive architectures (e.g., Anderson, 1983; Rumelhart & McClelland 1986; Sun, 2002), the relationships between sociocultural forces and individual cognition remain largely unexplored (again with some exceptions).

We believe, however, that the two fields of social simulation and cognitive architectures can be profitability integrated. As has been argued before (Sun & Naveh, 2004; Moss, 1999; Castelfranchi, 2001), social processes ultimately rest on the choices and decisions of individuals, and thus understanding the mechanisms of individual cognition can lead to better theories describing the behavior of aggregates of individuals. So far, most agent models in social simulation have been extremely simple (in the form of very simple automata with a few ad-hoc assumptions) or entirely absent (in the case of equation-based modeling). However, we believe that a more realistic cognitive agent model, incorporating realistic tendencies, inclinations, and capabilities of individual cognitive agents can serve as a more realistic basis for understanding the interaction of individuals (Moss, 1999). Although some cognitive details may ultimately prove to be irrelevant, this cannot be determined *a priori,* and thus simulations are useful in determining which aspects of cognition can be safely abstracted away.

At the same time, by integrating social simulation and cognitive modeling, we can arrive at a better understanding of individual cognition. By studying cognitive agents in a social context, we can learn more about the sociocultural processes that influence individual cognition.

In this chapter, we first describe the model proposed by Gilbert (1997) for capturing the growth of academic science. Gilbert's model lacks agents capable of meaningful autonomous action. We then describe a cognitive architecture, CLARION, that captures the distinction between explicit and implicit learning. This architecture has been used to model a variety of cognitive data (see Sun, Merrill, & Peterson, 2001; Sun, 2002; Sun, Slusarz, & Terry, 2005). We demonstrate how Gilbert's simulation can be redone in an enhanced way with CLARION-based agents (Sun & Naveh, in press). We argue that the latter approach provides a more cognitively realistic basis for social simulation.

PREVIOUS MODELS OF ACADEMIC SCIENCE

Science develops in certain ways. In particular, it has been observed that the number of authors

contributing a certain number of articles to a scientific journal follows a highly skewed distribution, corresponding to an inverse power law. This distribution, which is known as a Zipf distribution, is common to a number of other phenomena in information science. In the case of scientific publication, the tendency of authorship to follow a Zipf distribution was observed by Lotka (1926) and is known as Lotka's Law.

Simon (1957) developed a simple stochastic process for approximating Lotka's Law. One of the assumptions underlying this process is that the probability that a paper will be published by an author who has published *i* articles is equal to a/i^k, where *a* is a constant of proportionality.

Using Simon's work as a starting point, Gilbert (1997) attempts to model the growth of science, including Lotka's Law. He obtains his simulation data based on the assumption that the system selects a focal paper randomly first, which can be represented as a point in a two-dimensional space of ideas, and then it randomly selects a number of other papers, each of which occupies a different point in the nearby region and pulls the original point in its direction slightly. The resulting paper can be located on that two-dimensional space based on the above factors. Papers are randomly assigned authors, based on a stochastic process that takes the ratio of papers to published authors into account. To capture the constraint that academic papers must be original, a newly published paper must lie at least *m* coordinate units away from any other existing paper, where *m* is a constant.

Another assumption is that the number of papers produced in a given time period is determined by the number of papers in existence during the previous time period, by specifying a small probability of each existing paper acting as the seed for a new paper (and then by selecting an author for that paper).Thus, it is papers that spawn more papers, with authors playing only an ancillary role in the process.

Using this model, Gilbert obtained an idea space divided into clusters, which he identified as corresponding to different scientific specialties. Each cluster originated in a few seminal papers and accumulated additional papers at an increasing rate over time. This model yielded a set of publication trends that accorded with human data, including the power curve described above (as can be seen in Tables 1 and 2). A highly uneven distribution of number of publications per author was observed, with the majority of authors publishing but one paper. A similarly skewed outcome was obtained for the number of citations received per author, with most authors receiving a modest number of

Table 1. Number of authors contributing to chemical abstracts

# of Papers	Actual	Simon's Estimate	Gilbert's Simulation	CLARION Simulation
1	3,991	4,050	4,066	3,803
2	1,059	1,160	1,175	1,228
3	493	522	526	637
4	287	288	302	436
5	184	179	176	245
6	131	120	122	200
7	113	86	93	154
8	85	64	63	163
9	64	49	50	55
10	65	38	45	18
11	419	335	273	145

Table 2. Number of authors contributing to Econometrica

# of Papers	Actual	Simon's Estimate	Gilbert's Simulation	CLARION Simulation
1	436	453	458	418
2	107	119	120	135
3	61	51	51	70
4	40	27	27	48
5	14	16	17	27
6	23	11	9	22
7	6	7	7	17
8	11	5	6	18
9	1	4	4	6
10	0	3	2	2
11	22	25	18	16

citations, and a minority of authors receiving a large number of citations.

However, to a significant extent, Gilbert's model is not cognitively realistic. The model assumes that authors are non-cognitive and interchangeable; it therefore neglects a host of cognitive phenomena that characterize scientific inquiry (e.g., learning, creativity, evolution of field expertise, etc.). Using a more cognitively realistic model, we can envision addressing some of these omissions, as well as exploring other emergent properties of the model and their correspondence to real-world phenomena. The challenge, then, is to develop a model that explains macro-level phenomena in terms of micro-level cognitive processes.

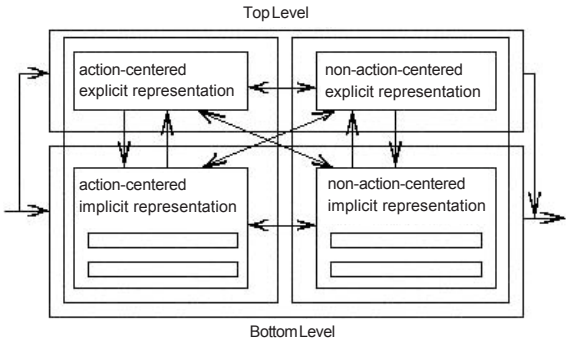
OUR COGNITIVE ARCHITECTURE FOR SOCIAL SIMULATION

The Focus on Explicit vs. Implicit Learning

In an attempt to understand the processes underlying human learning, various categories of knowledge have been proposed. Among them, one enduring distinction is that between explicit and implicit—or conscious and unconscious—learning (e.g., Reber, 1989). While both implicit and explicit learning have been actively investigated, the complex interaction between these two modes of learning has largely been downplayed or discounted (with a few exceptions; e.g., Mathews et al., 1989; Sun et al., 2001).

However, despite the lack of study of such interaction, recent evidence suggests that it is difficult to find a situation in which only one type of learning is employed (Reber, 1989; Seger, 1994). Our review of empirical data suggests that while one can manipulate conditions so that one type of learning is favored over the other, in nearly every case, both types are involved, with varying degrees of contributions from each (e.g., Sun et al., 2001; Mathews et al., 1989).

Figure 1. The CLARION architecture



In the next subsection, we describe a cognitive architecture, CLARION, which seeks to capture the interaction between explicit and implicit learning (Sun & Peterson, 1998; Sun et al., 2001).CLARION learns in bottom-up fashion, by extracting explicit rules from implicit knowledge, in accordance with what has been observed in humans (e.g., Karmiloff-Smith, 1986).

A Sketch of CLARION

CLARION is a general cognitive architecture with a dual representational structure (Sun, 1997; Sun, 2002). It consists of two levels: the top level encodes explicit knowledge, and the bottom level encodes implicit knowledge. See Figure 1 for a sketch of the model.

At the bottom level, the inaccessible nature of implicit knowledge is captured by a subsymbolic distributed representation provided by a backpropagation neural network. This is because representational units in a distributed environment are capable of performing tasks but are subsymbolic and generally not individually meaningful (see Rumelhart et al., 1986; Sun, 2002). Thus, they are relatively inacces-

sible. Learning at the bottom level proceeds in trial-and-error fashion, with the neural networks being guided by reinforcement learning (i.e., Q-learning; see Watkins, 1989).

At the top level, explicit knowledge is captured by a symbolic or localist representation, in which each unit is easily interpretable and has a clearer meaning. This characteristic captures the property of explicit knowledge being more accessible and manipulable (Sun, 2002). Learning at the top level involves, first, constructing a rule that corresponds to a "good" decision made by the bottom level. This rule is subsequently refined, either by generalizing or specializing it. Here, the learning process is guided by an "information gain" measure that compares the success ratio of various modifications of the current rule.

The overall algorithm of CLARION's action decision making is the following:

1. Observe the current state x.
2. Compute in the bottom level the value of each of the possible actions (a_i's) associated with the state x: $Q(x, a_1)$, $Q(x, a_2)$, …, $Q(x, a_n)$.
3. Find out all the possible actions ($b_1, b_2, \ldots, b_m$) at the top level, based on the state x and the rules in place at the top level.
4. Compare the values of a_i's with those of b_j's (which are sent down from the top level), and choose an appropriate action a.
5. Perform the action a, and observe the next state y and (possibly) the reinforcement r.
6. Update the bottom level in accordance with the *Q-Learning-Backpropagation* algorithm, based on the feedback information (as will be explained later).
7. Update the top level using the *Rule-Extraction-Refinement* algorithm (explained as follows).
8. Go back to Step 1.

At the bottom level, a Q-value measures the "quality" of an action in a given state; that is, $Q(x, a)$ indicates how desirable action a is in state x. Actions are selected based on Q-values. To acquire the Q-values, Q-learning, a reinforcement learning algorithm (Watkins, 1989), is used. Q-learning is implemented in backpropagation networks. See Sun (2002) for full details. Q-values are then used to decide probabilistically on an action to be performed (using a distribution of Q-values).

At the top level, explicit knowledge is captured by simple prepositional rules. An algorithm for extracting rules using information culled from the bottom level is the *Rule-Extraction-Refinement*, or RER, algorithm. The basic idea is as follows: whenever an action decided by the bottom level is successful, a rule (with conditions corresponding to the current input state and an action corresponding to the one selected by the bottom level) is created and added to the top level. Then, in subsequent interactions with the world, an agent may refine a rule by considering its outcome: if successful, an agent may try to generalize a rule by relaxing its conditions to make it more universal. If a rule is unsuccessful, an agent may try to specialize a rule by imposing further constraints on the rule and making them exclusive of the current state. This is an online version of hypothesis testing processes studied in other contexts.

To integrate the two levels, a number of methods are possible. Here, levels are selected stochastically, with a base probability of selecting each level. Other selection methods are possible as well (see Sun et al., 2001).

SIMULATION OF ACADEMIC SCIENCE WITH CLARION

In our simulation of academic science, we move to an agent-based model (Sun and Naveh, in press. Different from Gilbert's assumptions

in his simulation, we treat authors as cognitive agents. Thus, authors are not merely passive placeholders, but cognitively capable individuals whose success or failure depends on their ability to learn in the scientific world. Successful authors (that is, agents who manage to identify promising research leads early on) will go on to publish numerous papers in their area, whereas unsuccessful authors will be removed from the system and replaced.

Similar to Gilbert's approach, our model characterizes the scientific world as consisting of "papers," each of which proposes a new piece of knowledge and of authors who generate new papers. This coincides with the constructivist view of scientific inquiry, which sees scientific knowledge first and foremost as the product of a *constructive* process. On this view, the fruits of scientific inquiry are produced in a predetermined technological, linguistic, and social context. This is reflected in our model, in which papers are constructed from previous papers and themselves serve as a basis for future constructions.

To publish a paper, an agent adopts a focal idea (as represented by an existing paper), non-randomly, in accordance with some cognitive processes. The cognitive processes may be implicit or explicit. The agent then uses other ideas (published papers) that pull the original idea in different directions, also non-randomly, based on similar cognitive processes.

In addition, apart from utilizing existing ideas, an agent also performs local search to "optimize" the resulting idea. This reflects the fact that authors do not merely cobble together ideas from existing sources, but also try to integrate the different ideas and refine the final product.

Because our simulation involves learning agents, there is the possibility of failure; this is important, because humans can produce papers that prove to be unpublishable. This is in contrast to Gilbert's approach, in which ideas are undifferentiated in their quality. Instead, in our simulation, each agent has a set of evaluation functions that determine the quality of ideas in the multi-dimensional idea space. These functions specify the most important considerations in terms of evaluating a scientific idea (e.g., clarity, insightfulness, empirical evidence, theoretical results, and application potential). Agents are aware of these functions. However, just as researchers in the real world cannot predict precisely when the result of their research will meet with approval and interest, so the agents' individual valuations of these functions may differ from the community-wide valuation. This is reflected in a set of *individual* evaluation functions for each agent, consisting of a varied version of the global, or *communal* evaluation functions.

The author population consists of CLARION-based agents. The feedback to agents is based on paper acceptance or failure. In addition, agents are awarded partial feedback at each step of the paper generation process, amounting to a fraction of the unfinished paper's evaluation (as determined by the agent's own evaluation functions). This reflects the fact that agents do not stumble blindly through the publication process, but rather are guided to a certain extent by their experience and intuition.

An agent uses the bottom level of CLARION to select a focal idea and a number of pull ideas. These two tasks are carried out by one network. The network "learns" using the *Q-learning-Backpropagation* algorithm, which corresponds to a simple form of reinforcement learning and naturally captures sequences of actions (i.e., selecting the focal idea, then the first and second pull ideas).

On the other hand, an agent uses RER rule learning in CLARION to extract rules that determine: (1) how to choose focal ideas, and (2) how to choose pull ideas. These rules are used in conjunction with other rules already existing at the top level concerning local search,

which represent *a priori* knowledge.

Agents are pre-trained for a certain number of cycles before entering the system. A cycle corresponds to a single attempt by an agent to publish a paper, whether successfully or not. Reflecting a *publish or perish* academic environment, agents are evaluated every few cycles based on their publication record (success rate). If the latter falls below a minimum expected standard, the agent is removed from the academic world. It the agent passes all the evaluations, it retires upon reaching the maximum allowable age. Whenever an agent retires (or is removed), a new agent takes its place.

The results of our simulation are shown in Tables 1 and 2, along with results (reported by Simon) for *Chemical Abstracts* and *Econometrica,* and estimates obtained from previous simulations by Simon (1957) and Gilbert (1997). The figures in the tables indicate number of authors contributing to each journal, by number of papers each has published.

The CLARION simulation data for the two journals can be fit to the power curve $f(i) = a / i^k$, resulting in an excellent match. The results of the curve fit are shown in Table 3, along with correlation and error measures.

In our simulation, the number of papers per author reflected the cognitive ability and the cognitive suitability of an author, as opposed to being based on auxiliary assumptions such as those made by Gilbert (1997). This explains, in part, the greater divergence of our results from the human data: whereas Gilbert's simulation consists of equations selected to match the human data, our approach relies on much more detailed and lower-level mechanisms—namely, a cognitive agent model that is generic rather than task specific. The result of the CLARION-based simulation is therefore emergent, and not a result of specific and direct attempts to match the human data. We put more distance between mechanisms and outcomes, which makes it harder to obtain a match with the human data. Thus, the fact that we were able to match the human data shows the power of our cognitive agent-based approach compared to traditional methods of simulation.

Table 3. Results of fitting CLARION data to power curves; CA stands for Chemical Abstracts and E stands for Econometrica

Journal	*a*	*k*	Pearson R	R-Square	RMSE
CA	3806	1.63	0.999	0.998	37.62
E	418	1.64	0.999	0.999	4.15

VARYIATIONS OF COGNITIVE PARAMETER SETTINGS

Because CLARION captures a wide variety of cognitive phenomena, we can vary parameters that correspond to specific cognitive factors, and observe the effect on performance as a function of *cognition.* This confers an important advantage over other, more task-specific models, where differences in performance tend to be artifacts of the particular model used and may be of little independent interest. With CLARION, the parameters being altered are the fundamental building blocks of cognition, and thus observed differences in performance are far more likely to stem from testable differences in individual cognition.

The effect of learning rate on performance is shown in Figure 2. An agent's learning rate essentially determines its responsiveness to success or failure. As can be seen, the best performance is obtained under a moderately high learning rate (0.1-0.3). If the learning rate is too high, an agent's recent experiences tend to disproportionately impact the learning process. This leads the agent to submit articles that are too similar to recently successful articles, which leads to more non-unique articles that

Figure 2. The effect of learning rate on collective number of articles published

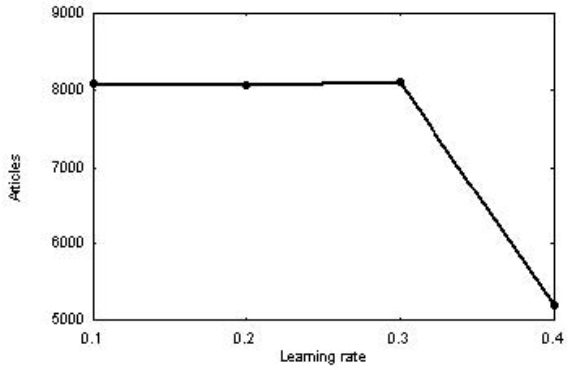


Figure 3. The effect of explicit vs. implicit learning on collective number of articles published

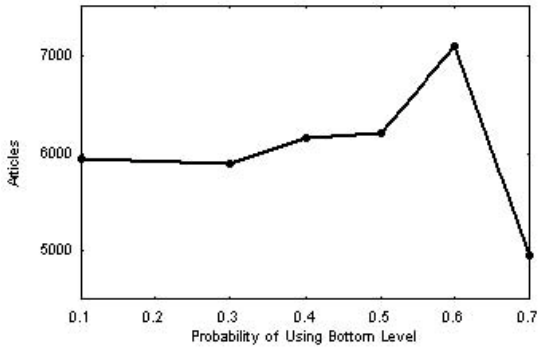
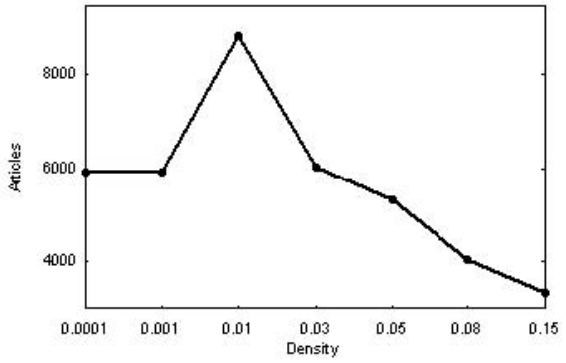


Figure 4. The effect of density on collective number of articles published



are in turn rejected; equally, an agent is too swift to abandon a previously promising line of research as the result of a single rejection. If, on the other hand, the learning rate is too low, an agent will be slow to capitalize on recent successes and failures.

The fact that a balance between implicit learning and explicit rule learning is desirable can be seen in Figure 3. While increasing the reliance on implicit learning can lead to modest gains, over-emphasizing it at the expense of explicit rule learning slows down the learning process dramatically. This is especially true during the initial steps of learning, when neural networks are still imperfectly trained, and rules, as crisp guidelines that are based on past success, are useful for speeding up learning.

Crucial in this connection is the existence of a high-quality rule base. This can be ensured, among other things, by the proper selection of a *density* parameter, as shown in Figure 4. If the parameter is too low, then rules persist even when they are no longer needed (for instance, when an agent has already exhausted a particular line of research encoded as a rule, and has moved on to other fields, represented by different combinations of ideas).On the other hand, when the parameter is too high, even successful rules are often deleted before they can be fully utilized. In both cases, performance suffers.

An agent's exploration of the idea space is modulated in considerable part by the built-in randomness (i.e., *temperature*) of its search process. As can be seen in Figure 5, agents are at their most prolific under a moderately high temperature setting—that is, when they show a willingness to experiment (i.e., to pursue new leads, which would not occur under a low temperature setting), while still being guided by their experience in the majority of cases. This observation accords with

Figure 5. The effect of temperature on collective number of articles published

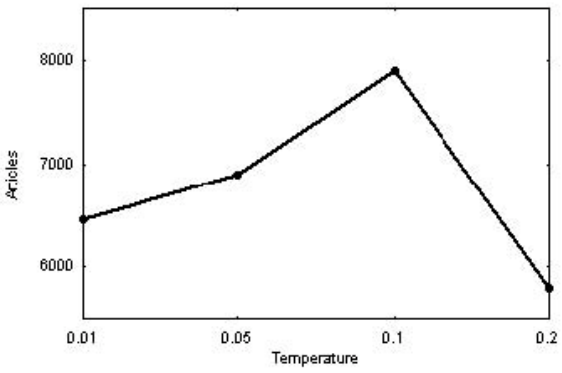


Figure 6. The effect of generalization threshold on collective number of articles published

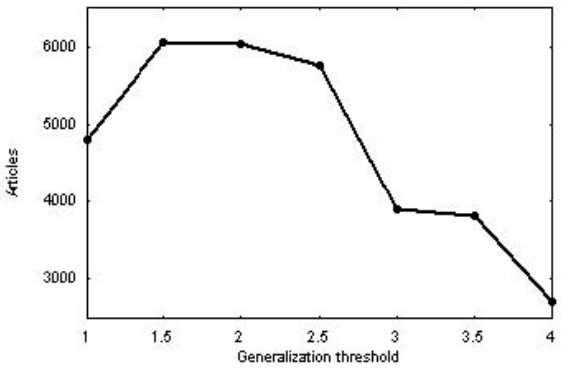
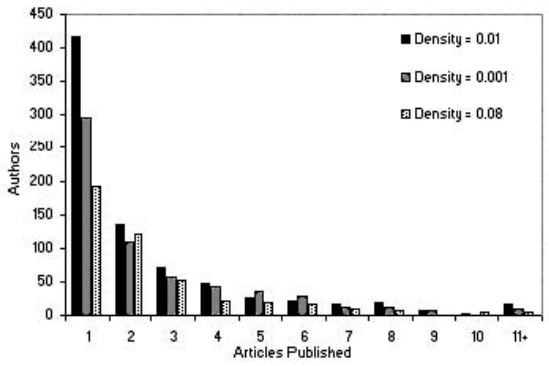


Figure 7. Authors contributing to final paper count, by number of articles that each has published; CLARION simulation results for different settings of the density parameter

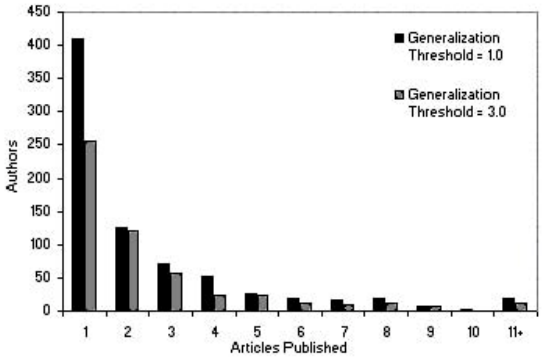


what we know of the role of serendipity in scientific discovery. In many areas of science—for instance, medicine—most of the major discoveries have been serendipitous, the result of seemingly irrelevant investigations. While such discoveries were formerly attributed to good fortune, it has since been argued (e.g., in Oliver, 1991) that serendipity is, in part, a cognitive faculty that can be nurtured and developed. Our model captures this characteristic by adopting a modest degree of randomness in the decision-making process.

As with other parameters considered so far, an agent's generalization threshold must be carefully selected (see Figure 6). If it is set too low, even less successful rules will be generalized, leading to a lower quality rule base. Too high, and it will prevent the generalization even of successful rules. In the latter situation, an agent will rigidly apply successful ideas only in the precise context in which they initially appeared (for instance, as the second *pull* idea in generating a paper) without recognizing their more general applicability.

As the preceding discussion shows, the cognitive parameters of individual agents are crucial in determining the rate of scientific progress in a society. By varying these parameters, we can arrive at communities that produce lesser or greater numbers of papers. Apart from this aggregate measure of scientific productivity, however, it is also interesting to see if the patterns of individual contribution observed in earlier simulations will be preserved. In particular, we want to see if the power curve obtained earlier will be obtained under different cognitive settings. Our results show that it is. As can be seen in Figures 7 and 8, different

Figure 8. Authors contributing to final paper count, by number of articles that each has published; CLARION results for different settings of the generalization threshold parameter



settings of the density and generalization threshold parameters lead to larger or smaller numbers of papers in aggregate, but they do not fundamentally change the authorship curve, which follows an inverse power distribution in all cases. Similar results were obtained for some other, though not all, ranges of cognitive parameters.

This result, which may be termed cognitive-social invariance, is an important one, since it shows that some regularities that characterize societies are to some extent invariant with respect to agent cognition (within a reasonable parameter range).While some societies may prove more successful than others in terms of total scientific productivity, the same large-scale patterns (distributions) may be observed regardless of cognitive differences. This reduces the likelihood that the patterns observed are a byproduct of a particular set of cognitive parameters. In contrast, in Sun and Naveh (2004), we have shown that some other patterns are indeed directly related to the settings of cognitive parameters.

GENERAL DISCUSSIONS

One important aim of this study has been to determine whether the results of a previous model of academic science (Gilbert, 1997) can be reproduced without resorting to the broad simplifications of an equation-based model. The results of our simulation suggest that the observed growth of academic science can indeed be captured even if one migrates to an agent-based model. Such a migration offers several important benefits. First, it allows us to leave behind certain artificial assumptions (for instance, that papers automatically spawn more papers). Second, it affords us the opportunity of studying the macro-level repercussions of behavior at the micro level. Third, it allows us to study patterns of interaction between individual agents. Although the latter interactions occur only indirectly in our model (either through the collision of too-similar papers generated by different authors, or through the exploitation of others' ideas in generating new papers), they nonetheless result in a model that is more socially realistic than Gilbert's equation-based model.

What makes our approach unusual, however, is not that it represents actors as agents, but that it takes agent cognition seriously. So far, most agent models in simulations have been rather simple, with little attention being paid to the mechanisms of individual cognition. This study shows that a more cognitively realistic simulation, with CLARION, can replicate the results of earlier simulations. It thus provides a dual corroboration of these models, by showing them to be independent both of whether or not an agent-based model is used, and of whether or not cognitive representations are involved. Therefore, while some cognitive details clearly cannot be abstracted away, others can, and along the way, we discover important cognitive-social invariances.

Apart from validation, however, cognitive realism in simulations may lead us to better representations of the target phenomenon. For instance, we identified earlier a possible way of representing the role of serendipity in science: namely, as a researcher's willingness to explore apparently suboptimal combinations of ideas, rather than adhering to *tried-and-true* sequences. The ability to represent such aspects of observed phenomena in terms organic to the agent model, rather than through auxiliary assumptions (for instance, by adding a *randomizing* function to the idea selection process in Gilbert's simulation) is an advantage of cognitively realistic simulations.

Another advantage of cognitive realism is that it allows us to theorize about the relative roles that individual cognitive factors play in the emergence of large-scale social phenomena. Thus, we were able to vary parameters of CLARION that corresponded to aspects of cognition and tested their effects on outcomes. Our investigation showed, for instance, that the tendency to engage in inductive reasoning (that is, an agent's generalization threshold) could dramatically influence the number of papers generated by the community. It moreover suggested that this phenomenon could be described by a u-shaped curve. Such results suggest how patterns of communal thinking may change as a consequence of shifts at the individual cognitive level.

CONCLUDING REMARKS

Whereas the use of simulation as a way of describing social phenomena continues to grow, the issue of adopting a realistic cognitive process description has largely been ignored. Against this background we propose using more complex cognitive models, known as cognitive architectures, to capture human behavior. Our model of scientific publication assumed that, in constructing new ideas from previous ones, authors were guided chiefly by cognitive processes. The simulations were done under different cognitive settings and suggest that some of the patterns obtained (for instance, Lotka's Law) are to some extent independent of the cognitive parameters selected.

By paying more attention to the details of individual cognition, we can arrive at more accurate representations of target social phenomena. We can also learn which cognitive mechanisms are significant in shaping social interactions, and which are not. Finally, we can study the emergence of large-scale social behavior from micro-level cognitive processes.

Note that CLARION has been implemented as a set of Java packages. For more information, see the CLARION Web page at http://www.cogsci.rpi.edu/~rsun/clarion.html

REFERENCES

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought.* Mahwah, NJ: Lawrence Erlbaum.

Axtell, R. L. (2000). Why agents? On the varied motivations for agent computing in the social sciences. *Proceedings of the Workshop on Agent Simulation: Applications, Models and Tools*. Argonne, IL: Argonne National Laboratory.

Carley, K. M., & Newell, A. (1994). The nature of the social agent. *Journal of Mathematical Sociology, 19*(4), 221-262.

Castelfranchi, C. (2001). The theory of social functions: Challenges for computational social science and multi-agent learning. *Cognitive*

Systems Research, Special Issue on the Multi-Disciplinary Studies of Multi-Agent Learning, *2*(1), 5-38.

Gilbert, N. (1997), A simulation of the structure of academic science. *Sociological Research Online, 2*(2).

Karmiloff-Smith, A. (1986). From meta-processes to conscious access: Evidence from children's metalinguistic and repair data. *Cognition, 23,* 95-147.

Lotka, A. J. (1926). The frequency distribution of scientific productivity. *Journal of the Washington Academy of Sciences, 16,* 317-323.

Mathews, R., Buss, R., Stanley, W., Blanchard-Fields, F., Cho, J., & Druhan, B. (1989). Role of implicit and explicit processes in learning from examples: A synergistic effect. *Journal of Experimental Psychology: Learning, Memory and Cognition, 15,* 1083-1100.

Moss, S. (1999). *Relevance, realism and rigor: A third way for social and economic research.* CPM Report No. 99-56, Center for Policy Analysis, Manchester Metropolitan University, UK.

Naveh, I., & Sun, R. (in press). A cognitively based simulation of academic science. Computational and Mathematical Organization Theory.

Oliver, J.E. (1991). *The incomplete guide to the art of discovery.* New York: Columbia University Press.

Reber, A. (1989). Implicit learning and tacit knowledge. *Journal of Experimental Psychology: General, 118*(3), 219-235.

Rumelhart, D., & McClelland, J. (Eds.). (1986). *Parallel distributed processing I.* Cambridge, MA: MIT Press.

Seger, C. (1994). Implicit learning. *Psychological Bulletin, 115*(2), 163-196.

Simon, H. A. (1957). *Models of man, social and rational.* New York: John Wiley & Sons.

Sun, R. (2002). *Duality of the mind.* Mahwah, NJ: Lawrence Erlbaum.

Sun, R. (2006). Cognition and multi-agent interaction: From cognitive modeling to social simulation. New York: Cambridge University Press.

Sun, R., Merrill, E., & Peterson, T. (2001). From implicit skills to explicit knowledge: A bottom-up model of skill learning. *Cognitive Science, 25*(2), 203-244.

Sun, R., & Naveh, I. (2004). Simulating organizational decision making using a cognitively realistic agent model. *Journal of Artificial Societies and Social Simulation, 7*(3).

Sun, R., Slusarz, P., & Terry, C. (2005). The interaction of the explicit and the implicit in skill learning: A dual-process approach. *Psychological Review, 112*(1), 159-192.

Watkins, C. (1989). *Learning with delayed rewards.* PhD Thesis, Cambridge University, UK.

Chapter XII
Nature-Inspired Knowledge Mining Algorithms for Emergent Behaviour Discovery in Economic Models

David Al-Dabass
Nottingham Trent University, UK

ABSTRACT

Economic models exhibit a multiplicity of behaviour characteristics that are nonlinear and time-varying. Emergent behaviour appears when reduced order models of differing characteristics are combined to give rise to new behaviour dynamics. In this chapter we apply the algorithms and methodologies developed for nature-inspired intelligent systems to develop models for economic systems. Hybrid recurrent nets are proposed to deal with knowledge discovery from given trajectories of behaviour patterns. Each trajectory is subjected to a knowledge mining process to determine its behaviour parameters. The knowledge mining architecture consists of an extensible recurrent hybrid net hierarchy of multi-agents where the composite behaviour of agents at any one level is determined by those of the level immediately below. Results are obtained using simulation to demonstrate the quality of the algorithms in dealing with the range of difficulties inherent in the problem.

INTRODUCTION

Recurrent inference networks are introduced to represent knowledge bases that model dynamic intelligent systems. Through a differential abduction process, the causal parameters of the system behaviour are determined from measurements of its output to represent the knowledge embedded within (Al-Dabass, Zreiba, Evans, & Sivayoganathan, 2001). The

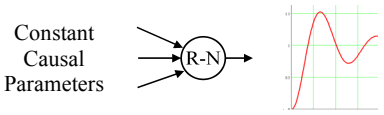
use of dynamical knowledge mining processes ensures that knowledge evolution is tracked continuously (Al-Dabass et al., 2002a). Meta-knowledge, defined in terms of the causal parameters of the evolution pattern of this first-level knowledge, is further determined by the deployment of second-level dynamical processes (Bailey, Grossman, Gu, & Hanley, 1996). In data mining applications, for example, there is a need to determine the causes of particular behaviour patterns. Other applications include cyclic tendencies in stock values and sales figures in business and commerce, changes in patient recovery characteristics, and predicting motion instabilities in complex engineering structures (Al-Dabass et al., 2002c). Full mathematical derivation is given together with simulations and examples to illustrate the techniques involved.

- **Knowledge Models:** To understand and control the behaviour of economic systems, models that represent the knowledge embedded within these systems are formulated and used to acquire this knowledge from measurements. In data mining applications, for example, there is a need to determine the causes of particular behaviour patterns. Applications include cyclic tendencies in stock sales figures in business and commerce, sudden movements in share indices changes, and in no economic areas, patient recovery characteristics and predicting motion instabilities in complex engineering structures.
- **Hybrid Inference Networks:** To represent the knowledge embedded within intelligent systems, a multilevel structure is put forward. By its very nature this knowledge is continually changing and needs dynamic paradigms to represent and acquire its parameters from observed data. In a normal inference network, the cause-and-effect relationship is static, and the effect can be easily worked out through a deduction process by considering all the causes through a step-by-step procedure which works through all the levels of the network to arrive at the final effect. However, reasoning in the reverse direction, such as that used in diagnosis, starts with observing the effect and working back through the nodes of the network to determine the causes.
- **Knowledge Mining for Stock Market Models:** Work in this chapter extends these ideas of recurrent or dynamical systems networks to economic models where some or all the data within the knowledge base is time varying. The effect is now a time-dependent behaviour pattern, which is used as an input to a differential process to determine knowledge about the system in terms of time-varying causal parameters. These causal parameters will themselves embody knowledge (meta knowledge) which is obtained through a second-level process to yield second-level causal parameters. These processes consist of a differential part to estimate the higher time derivative knowledge, followed by a non-linear algebraic part to compute the causal parameters.

ECONOMIC SYSTEM MODELLING AND SIMULATION USING HYBRID RECURRENT NETWORKS

Numerous economic systems in practice exhibit complex behaviour that cannot be easily modelled using simple nets (Berndt & Clifford, 1996). In this part we re-cast this problem in terms of hybrid recurrent nets, which consist of combinations of static nodes, either logical or arithmetic, and recurrent nodes. The behaviour of a typical recurrent node is modelled as a

Figure 1. A Recurrent Node (R-N) exhibits a temporal behaviour at the output despite having constant causal parameters



second-order dynamical system. The causal parameters of such a recurrent node may themselves exhibit temporal tendencies that can be modelled in terms of further recurrent nodes. Layers of recurrent nodes are added until a complete account of the behaviour of the system has been achieved (Al-Dabass, Evans, & Sivayoganathan, 2002b). Algorithms are given to estimate the values of the parameters of these models from behaviour trajectories of intelligent systems. One novel aspect of the work lies in having a simple hierarchical sixth-order linear model to represent a fairly complicated behaviour encountered in numerous real examples in finance, biology, and engineering.

Hybrid Recurrent Network Models

Many physical, economical, and biological phenomena exhibit temporal behaviour even when the input *causal* parameters are constant (see Figure 1).

To model this oscillatory behaviour, a second-order integral hybrid model is proposed, shown in Figure 2. This model is based on the well-known second-order dynamical system which has the following form:

$$\omega^{-2}\, x'' + 2.\, \zeta.w^{-1}.x' + x = u \qquad (1)$$

where x is the output of the node and ω, ζ, and u are the natural frequency, damping ratio, and input respectively, which represent the three causal parameters that form the input. To con-

Figure 2. Hybrid integral-recurrent net to model the temporal behaviour of the node in Figure 1

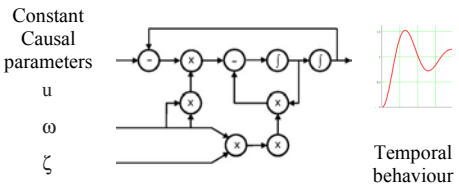


figure this differential model as a recurrent network, twin integral elements are used to form a hybrid integral-recurrent net as shown in Figure 2.

Structure of the Hybrid Integral-Recurrent Net

The net shown in Figure 2 is a direct representation of equation 1 and can be derived as follows:

1. By multiplying both sides by ω^2 we get:

   $$x'' + 2.\, \zeta.\omega.x' = \omega^2.(u - x) \qquad (1\text{-A})$$

   OR

   $$x'' = \omega^2.(u - x) - 2.\, \zeta.\omega.x' \qquad (1\text{-B})$$

2. The output of the net x is fed back to the first subtraction node on the left, as the input from the left of this node is u the output is (u – x).
3. The middle input (to the whole net) from the left is ω. It is fed as two separate inputs to the multiplication node x to form ω^2 at its output, shown with an up arrow feeding as the lower input of the multiplier node above it, which is the second node from the left in the top chain of nodes.
4. The output of this multiplier node is therefore $\omega^2.(u - x)$—that is, the RHS of equation 1-A.

5. The bottom input from the left (to the whole net) is z which is fed as the lower input to the first of the two multipliers in the bottom chain of two nodes—as the top input to this node is w, the output is z.w. which is multiplied by 2 in the second node in the chain to produce 2. ζ.ω.
6. The last node on the right in the top long node chain is an integrator node that generates x as stated in (2) above. As it is an integrator node, the input to it must therefore be the derivative of x (i.e., x'). This is multiplied by the output of the right node in the bottom two-node chain (which is 2. ζ.ω) to produce 2. ζ.ω.x' , which is the second term in the LHS of equation 2-1-A or the second term on the RHS of equation 2-1-B.
7. By subtracting this output from the output of the middle node in the top row, we get the full RHS of equation 1-B (i.e., $\omega^2.(u - x) - 2. \zeta.\omega.x'$).
8. As the output of the second integrator from the right (in the top chain) is the first derivative of x, x', the input to this integrator node must be x'' (i.e., the LHS of equation 1-B).
9. Simply connecting the output of the middle node of the top chain (which is $\omega^2.(u - x) - 2. \zeta.\omega.x'$) into the input of the second integrator from the right (x'') will just complete the equation.

Models of Hierarchical Recurrent Nodes

The output trajectory of the system may be more complex than can be represented by a simple second-order differential model. In this case each causal parameter may itself be modelled as having a dynamical behaviour, which may or may not be oscillatory. One such case is where two of the three causal parameters have second-order dynamical characteristics, as shown in Figure 3.

The second-order model of a node in a given layer in the hierarchy is given by equation 1 above. Starting with the final output node, let both u and omega have their own second-order dynamics. The input **u** is the output of the following second-order system:

$$\omega_u^{-2}\,\mathbf{u''} + 2.\,\zeta_u.\,\omega_u^{-1}.\mathbf{u'} + \mathbf{u} = u_u \qquad (2)$$

The natural frequency **ω** is the output of the following second-order system:

$$\omega_\omega^{-2}\,\boldsymbol{\omega}'' + 2.\,\zeta_\omega.\,\omega_\omega^{-1}.\boldsymbol{\omega}' + \boldsymbol{\omega} = u_\omega \qquad (3)$$

Thus the behaviour trajectory is generated by the following sixth-order vector differential equation (using Runge Kutta in Mathcad for this example).

Figure 3. Two of the causal parameters of the final node have temporal behaviour modelled as 2nd order hybrid integral recurrent nets.

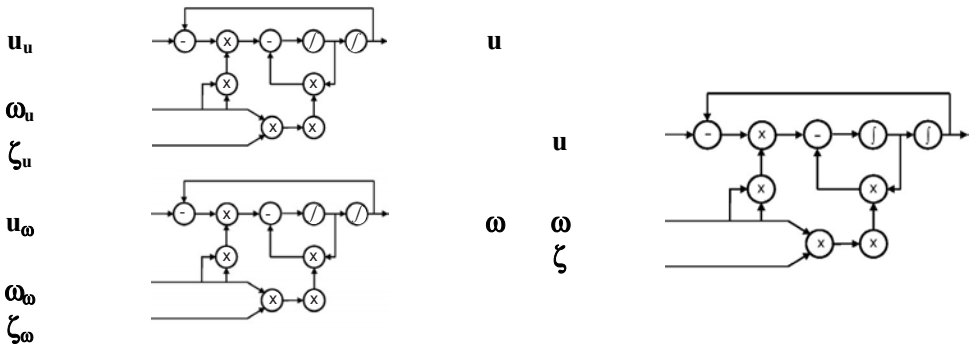


Figure 4. The derivative vector for generating the economic system output using subsystems for u and omega.

$$D(t\ x) := \begin{bmatrix} x_2 \\ x_5\ x_5\ x_3 - 2\ z\ x_5\ x_2 - x_5\ x_5\ x_1 \\ x_4 \\ (wu\ wu\ uu) - 2\ zu\ wu\ x_4 - wu\ wu\ x_3 \\ x_6 \\ (ww\ ww\ uw) - 2\ zw\ ww\ x_6 - ww\ ww\ x_5 \end{bmatrix}$$

First Order Vector Form: To provide a simulation output of the node trajectory, the second-order equation is converted to a second-order vector differential equation that can be easily computed.

x1 = x, and x2 = x'

In Figure 4, x1 and x2 represent x and x', x3 and x4 represent u and u', and x5 and x6 represent ω and ω' respectively. To generate the trajectory shown in Figure 5, the following values were used: for the u subsystem, u started from 0 aiming at $u_u = 1$ at a rate of $\omega_u = 5$ rad/s with $\zeta_u = 0.3$. For the ω subsystem, ω started from 4 rad/s aiming at $u_\omega = 32$ rad/s at a rate of $\omega_\omega = 4$ rad/s with $\zeta_w = 0.1$. Figure 5 shows the resulting compound trajectory of x (oscillatory trace), together with the trajectories for u (upper trace) and ω (lower trace).

DERIVATIVE ESTIMATION USING RECURRENT NETWORKS

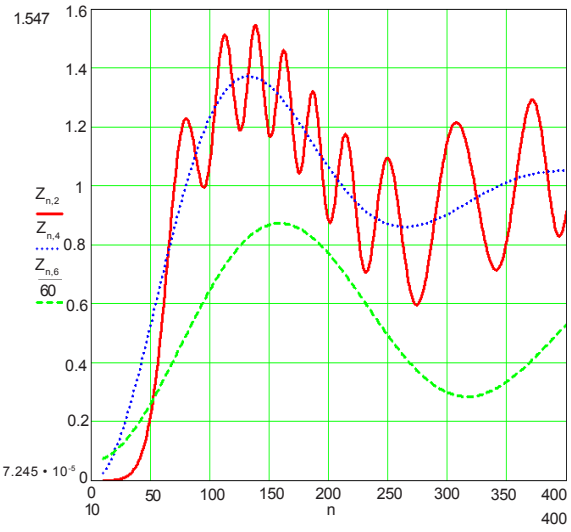
Based on models that describe the behaviour of complex natural and physical systems, a number of explicit static algorithms are developed to estimate the parameters of recurrent second-order models that approximate the behaviour of these complex higher-order systems (Ren, Al-Dabass, & Su, 1996; Bovet & Crescenzi, 1994). These algorithms rely on the availability of the time derivatives of the trajectory. In this section a cascaded recurrent network architecture is proposed to "abduct" these derivatives in successive stages (Cant, Churchill, & Al-Dabass, 2001). The technique is tested successfully on parameter tracking algorithms ranging from the constant parameter algorithm that only requires derivatives up to order 4 to an algorithm that tracks two variable parameters and requires up to the eighth time derivatives.

Algorithm for Constant Parameters from Single Point Data

Consider using the first to fourth time derivatives at a single point. Given the second-order system:

$$\omega^{-2}\ x'' + 2.\ \zeta.\omega^{-1}.x' + x = u \qquad (4)$$

Figure 5. Simulated trajectory of a hierarchical recurrent node (oscillatory trace), with 2 variable inputs: u (upper trace) and omega (lower trace)



Differentiate with respect to t:

$$\omega^{-2}\, x''' + 2.\, \zeta.\omega^{-1}.x'' + x' = 0 \quad (5)$$

divide by x'':

$$\omega^{-2}\, x'''/\, x'' + 2.\, \zeta.\omega^{-1}.\, + x'/\, x'' = 0 \quad (6)$$

and differentiate with respect to t again to give:

$$\omega^{-2}.[(x''.\, x'''' - x'''^2)\, /\, x''^2] + 0 + [(x''^2 - x'.\, x''')\, /\, x''^2] = 0 \quad (7)$$

Expressions for estimated ω, estimated ζ using (5), and estimated u result as follows:

$$E\omega^{-2} = [x''.\, x'''' - x'''^2]\, /\, [x'.\, x''' - x''^2] \quad (8)$$

$$E\zeta = -[E\omega^{-2}\, x''' + x']\, /\, [2.\, \omega^{-1}.x''] \quad (9)$$

$$Eu = E\omega^{-2}.\, x'' + 2.E\zeta.\, E\omega^{-1}\, .\, x' + x \quad (10)$$

High-Order Algorithms

Assume that the first and higher time derivative of u to be non zero. For simplicity assume that both a and b (the coefficients of x'' and x' to make symbol manipulation easier) are constant and hence disappear on first differentiation. The extra information needed for u', u'', u''', and u'''' to be non zero is extracted from the fifth, sixth, seventh, and eighth time derivatives of the trajectory. Only the case for the u' is shown here, the others for u'' and so on are simple extensions of the idea and are left as an exercise for the reader.

$$a.x'' + b.x' + x = u \quad (11)$$

Differentiate wrt to t and assume u' is non zero to give:

$$a.x''' + b.x'' + x' = u' \quad (12)$$

Differentiate again and set u'' = 0 gives:

$$a.x'''' + b.x''' + x'' = 0 \quad (13)$$

Divide equation 11 by x''' to isolate b:

$$a.x''''/x''' + b + x''/x''' = 0 \quad (14)$$

Differentiate again to eliminate b:

$$a.(x'''''.x'' - x''''^2)/x'''^2 + (x'''^2 - x''\, .\, x'''')/x'''^2 = 0 \quad (15)$$

Re-arranging for a gives:

$$E(a) = (x''\, .\, x'''' - x'''^2)/(x''''' .\, x''' - x''''^2) \quad (16)$$

Solve for *b* by substituting a from equation 16 into equation 14:

$$E(b) = -x''/x''' - a.x''''/x'''$$

which after substituting for a and manipulating gives:

$$E(b) = (x''.x''''' - x'''.x'''')/(x'''''^2 - x'''\, .\, x''''') \quad (17)$$

We can now substitute these values for *a* and *b* into equation 1 to solve for u,

$$u = a.x'' + b.x' + x$$

A Recurrent Architecture to Estimate Time Derivatives

The structure of each cell of the recurrent network is shown in Figure 6. The output of each cell feeds the input to the next one to generate the next higher-order time derivative (see Figure 7). The output of the system and the cascade of first-order recurrent network filters are simulated using the fourth-order Runge-

Figure 6. A single stage recurrent sub-net using an integrator in the feedback path to estimate the derivative x' = w(x-E(x)); the net is a low pass filter with a cut off frequency w.

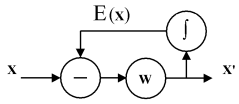


Figure 7. A 2nd order recurrent network to estimate 1st and 2nd time derivatives.

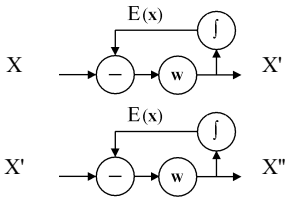
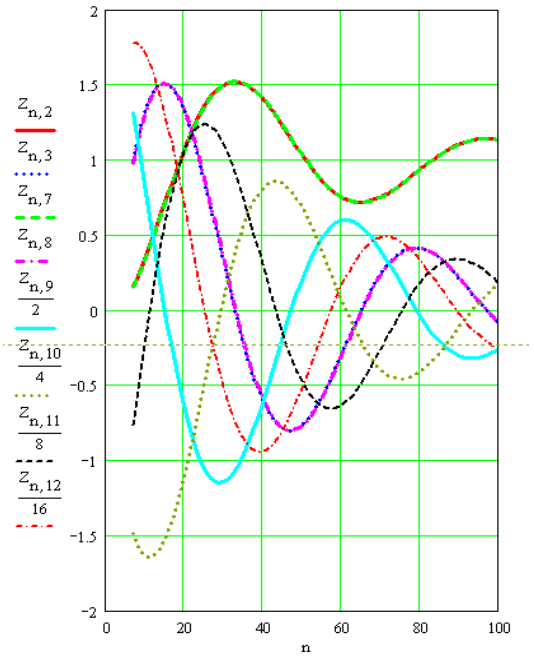


Figure 8. A cascade of five recurrent cells plus the second-order trajectory model

$$x := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad D(t\ x) := \begin{bmatrix} x_2 \\ -\ ^2 x_1 - 2\ \ x_2 + \ ^2 u \\ G\left(x_1 - x_3\right) \\ G\left[G\left(x_1 - x_3\right) - x_4\right] \\ G\left[G\left[G\left(x_1 - x_3\right) - x_4\right] - x_5\right] \\ G\left[G\left[G\left[G\left(x_1 - x_3\right) - x_4\right] - x_5\right] - x_6\right] \\ G\left[G\left[G\left[G\left[G\left(x_1 - x_3\right) - x_4\right] - x_5\right] - x_6\right] - x_7\right] \end{bmatrix}$$

$$Z := \text{Rkadapt}(x\ t0\ t1\ N\ D)$$

Figure 9. A typical set of time derivatives estimated from the trajectory of an oscillatory second-order dynamical system.



Kutta method in Mathcad. The derivatives vector is shown in Figure 8. Figure 9 shows a typical set of derivatives estimated from a damped oscillatory trajectory.

RESULTS AND DISCUSSION

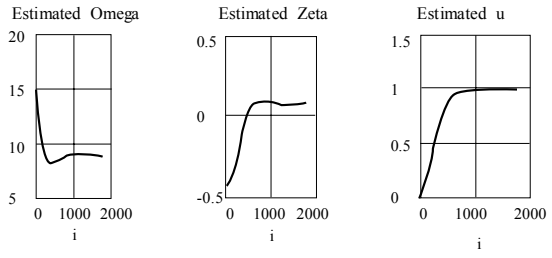
First Algorithm Using Constant Parameters

This algorithm uses a single time point and four higher-order time derivatives. The filter cascade provide a continuous estimate of the first to fourth time derivative x', x'', x''', and x''''. This provides a continuous estimate of all parameters at each point on the trajectory. The results of the estimation are given in Figure 10, which shows fast and accurate convergence.

Discussion

Estimated values for constants parameters are very close to the desired set values. The derived algorithms estimate ω, ζ, and u for a good range of values: ω from 1 to10, ζ between +/- (0.01 to 1), and u between +/- (0.5-40), and give accurate estimates. Estimation errors decreased as ω increased, particularly for small ζ (less than 0.5), where oscillation provided wide variation in the variables to decrease errors. The differences between the (simulated) system

Figure 10. Estimated constant omega, zeta, and u



time derivatives (x, x', and x'') and their estimates from the filter cascade depended on G (the cut-off frequency): high G provided more accurate estimation of derivatives but made the algorithms prone to noise and vice versa. Another disadvantage of high G from the simulation point of view is that simulation time increases considerably due to the integration routine adapting to ever smaller steps. The algorithm provides fast convergence.

Results for the Higher-Order Algorithm

Mathcad routines are set up to generate the input u as a second-order system with its own parameters of natural frequency, damping ratio, and input. The input subsystem damping ratio was set to 0.05 to generate an oscillatory behaviour for long enough to test the parameter tracking algorithm thoroughly. The frequency of the input is set to 16 radians per second, one quarter of the frequency of the data natural frequency. The derivative generation cascade is increased by one to produce the fifth time derivative. The results are shown in Figure 11.

The actual input is the smooth trace, which gives approximately one-and-one-quarter cycles over a period of half a second as expected—that is, 16 radians/s = 2.546 Hertz. The large jagged trace shows the results from the previous constant u derivation algorithm, which is failing completely to track the input parameter. The third trace shows the result of the new algorithm, which is managing to track the input much more closely; however, it starts to diverge slightly near the peak of the cycle, but then returns to track it well right down and round the lower trough of the input trajectory.

To check the quality of tracking as time progresses, a second set of results (Figure 12) is obtained, with integration time extended to one second to give two-and-a-half cycles. It is clear that tracking remains stable. It is interesting to note that the old algorithm, while completely failing to track the upper half of the input trajectory, seems to track it well during the lower half, but not as well as the new algorithm.

KNOWLEDGE MINING IN HYBRID INFERENCE NETS

Deduction and Abduction in Inference Networks

To engineer a knowledge base to represent economic systems, a multilevel structure is needed. By its very nature, the knowledge embedded within these systems is continually changing and needs dynamic paradigms to represent and acquire their parameters from observed data. In a normal inference network, the cause-and-effect relationship is static and the effect can be easily worked out through a deduction process by considering all the causes through a step-by-step procedure which works through all the levels of the network to arrive at the final effect. On the other hand, reasoning in the reverse direction, such as that used in diagnosis, starts with observing the effect and working back through the nodes of the network to determine the causes; this is termed knowledge mining.

Dynamical Knowledge Mining Processes

These ideas are applied here to recurrent or dynamical systems networks where some or all

Figure 11. Results of the high-order algorithm

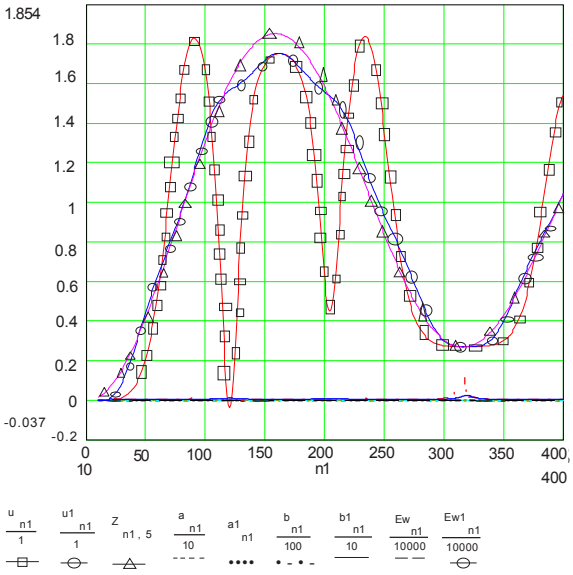
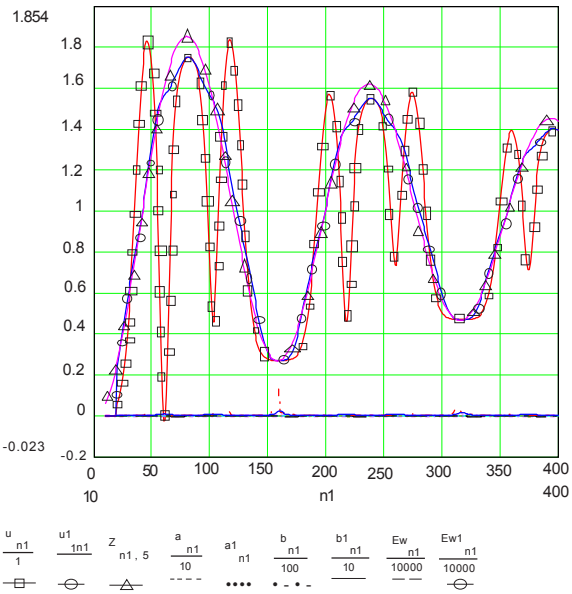


Figure 12. Results of the high-order algorithm for one second integration time



of the data within the knowledge base is time varying. The effect is now a time-dependent behaviour pattern, which is used as an input to a differential abduction process to determine the knowledge about the system in terms of time varying causal parameters. These causal parameters will themselves embody knowledge (meta knowledge), which is obtained through a second-level mining process to yield second-level causal parameters. These mining processes consist of a differential part to estimate the higher time derivative knowledge, followed by a non-linear algebraic part to compute the causal parameters.

Hierarchical Causal Parameters with Temporal Behaviour

The output trajectory of the system may be more complex than can be represented by a simple second-order differential model. In this case each causal parameter is itself modelled as having a dynamical behaviour, which may or may not be oscillatory. One such case is where two of the three causal parameters have second-order dynamical characteristics, as was shown in Figure 3.

Knowledge Mining Algorithms

Several explicit algorithms for the three usual parameters characterising the behaviour of second-order models have been derived (Al-Dabass et al., 1999a) based on information available from the systems time trajectory. Leaving the second-order model in its second time derivative form and using three points on the trajectory, each providing position, velocity, and acceleration, a set of three simultaneous algebraic equations were solved to yield estimates of input, natural frequency, and damping ratio. An online dynamical algorithm was then configured to combine estimates of the trajectory time derivatives with these explicit static non-linear functions to provide continuous parameter estimation in real time.

Multipoint Algorithms

Several algorithms are easily derived to estimate values of causal parameters using as many points from the trajectory as necessary to form a set of simultaneous algebraic equations. The parameters to be estimated form the unknown variables, and the trajectory values and their time derivatives form the constant parameters of these equations (Al-Dabass et al., 1999b, 2002a, 2002b).

Algorithm 1: Three-points in x, x' and x''. Consider estimating ω, ζ, and u using three sets of x, x', and x'':

$$\omega^{-2}\, x_1'' + 2.\, \zeta.\omega^{-1}.x_1' + x_1 = u \qquad (18)$$

$$\omega^{-2}\, x_2'' + 2.\, \zeta.\omega^{-1}.x_2' + x_2 = u \qquad (19)$$

$$\omega^{-2}\, x_3'' + 2.\, \zeta.\omega^{-1}.x_3' + x_3 = u \qquad (20)$$

Subtracting (19) from (18) and (20) from (18) gives:

$$\omega^{-2}.(x_1'' - x_2'') + 2.\, \zeta.\omega^{-1}.(x_1' - x_2') + (x_1 - x_2) = 0 \qquad (21)$$

$$\omega^{-2}.(x_1'' - x_3'') + 2.\, \zeta.\omega^{-1}.(x_1' - x_3') + (x_1 - x_3) = 0 \qquad (22)$$

Divide (21) by $(x_1' - x_2')$ and (22) by $(x_1' - x_3')$ gives:

$$\omega^{-2}.(x_1'' - x_2'') / (x_1' - x_2') + 2.\, \zeta\omega^{-1} + (x_1 - x_2) / (x_1' - x_2') = 0 \qquad (23)$$

$$\omega^{-2}.(x_1'' - x_3'') / (x_1' - x_3') + 2.\, \zeta\omega^{-1} + (x_1 - x_3) / (x_1' - x_3') = 0 \qquad (24)$$

and subtracting gives:

$$\omega^{-2}.[(x_1'' - x_2'') / (x_1' - x_2') - (x_1'' - x_3'') / (x_1' - x_3')] + [(x_1 - x_2) / (x_1' - x_2') - (x_1 - x_3) / (x_1' - x_3')]=0 \qquad (24\text{-A})$$

Using the following notations:

$\Delta 12 = (x_1 - x_2)$, $\Delta' 12 = (x_1' - x_2')$, $\Delta'' 12 = (x_1'' - x_2'')$

$\Delta 13 = (x_1 - x_3)$, $\Delta' 13 = (x_1' - x_3')$, $\Delta'' 12 = (x_1'' - x_3'')$

we get expressions for estimated ω, ζ using (21), and estimated u:

$$E\omega^2 = [\Delta''13.\Delta'12 - \Delta'13] / [\Delta'12.\Delta'13 - \Delta 13.\Delta'12]$$

$$E\zeta = [-E\omega^{-2}.\Delta 12 - \Delta 12] / [2.\, E\omega^{1}.\Delta'12]$$

$$Eu = E\omega^{-2}.x_1'' + 2.\, E\zeta..E\omega^{-1}\, .x_1' + x_1$$

Algorithm 2: Two-points and one extra derivative. Consider using two sets of x, x', x'', and x'''.

$$\omega^{-2}\, x_1'' + 2.\, \zeta.\omega^{-1}.x_1' + x_1 = u \qquad (25)$$

$$\omega^{-2}\, x_2'' + 2.\, \zeta.\omega^{-1}.x_2' + x_2 = u \qquad (26)$$

Subtracting (26) from (25) and dividing by $(x_1' - x_2')$:

$$\omega^{-2}.(x_1'' - x_2'') / (x_1' - x_2') + 2.\, \zeta.\omega^{-1} + (x_1 - x_2) / (x_1' - x_2') = 0 \qquad (27)$$

Differentiating (27) with respect to t gives:

$$[\omega^{-2}[(x_1' - x_2').(x_1''' - x_2''')] - (x_1'' - x_2'')^2] / (x_1' - x_2')^2 + 0 + [(x_1' - x_2')^2 - (x_1 - x_2).(x_1'' - x_2'')] /(x_1' - x_2') = 0 \qquad (28)$$

Using the following notations:

$\Delta 12 = (x_1 - x_2)$, $\Delta' 12 = (x_1' - x_2')$,

$\Delta''12 = (x_1'' - x_2'')$ and $\Delta'''12(x_1''' - x_2''')$

We get expressions for estimated ω, estimated ζ using (27), and estimated u:

$E\omega^2 = [\Delta'12.(\Delta'''12) - (\Delta''12)^2] / [\Delta'12)^2 - \Delta12.\Delta''12]$

$E\zeta = [-E\omega^{-2}.\Delta12/\Delta'12 - \Delta12/\Delta'12] / [2.E\omega^{-1}]$

$Eu = E\omega^{-2}.x_1'' + 2.E\zeta..E\omega^{-1}.x_1' + x_1$

Single-Point Algorithms

In this section we relax the constant parameter condition by assuming a linear time variation—that is, constant first derivative but zero second and higher time derivatives of parameters. As may be expected, more information is needed for this new case, which is to be extracted from the system output trajectory by obtaining higher time derivatives. Explicit functions of the parameters are still possible, as well as those of their first-time derivatives. A set of three equations, one for each parameter, is formulated and numerically computed in real time together with the state estimation vector observer to yield continuous trajectories of the parameters. This is a different technique to that of augmenting the state derivative vector with the parameter derivatives—instead of driving these derivatives with some function of the error between the system and model output, we provide an explicit function that should aid successful and speedy convergence to actual parameter values and provide continuous tracking. This should hold even when the parameters are changing rapidly compared to the system's natural frequency or time constant.

These are classified according to the order of the parameter variation used in the derivation—that is, constant, first-order polynomial (constant u' but u''=0), second-order polynomial (constant u'' but u'''=0), and so forth.

Algorithm 3: Constant parameters. Consider using the first to fourth time derivatives at a single point. Given the second-order system:

$$\omega^{-2} x'' + 2.2.\zeta.\omega^{-1}.x' + x = u \quad (29)$$

Differentiate with respect to t and divide by x'':

$$\omega^{-2} x'''/ x'' + 2.\zeta.\omega^{-1} + x'/ x'' = 0 \quad (30)$$

and differentiate with respect to t again to give:

$$\omega^{-2}.[(x''.x'''' - x'''^2) / x''^2] + 0 + [(x''^2 - x'.x''') / x''^2] = 0 \quad (4\text{-}15)$$

We get expressions for estimated ω, estimated ζ using (3-14), and estimated u:

$E\omega^2 = [x''.x'''' - x'''^2] / [x'.x''' - x''^2]$

$E\zeta = -[E\omega^{-2} x''' + x'] / [2.E\omega^{-1}.x'']$

$Eu = E\omega^{-2}.x'' + 2.E\zeta^{-1}.x' + x$

Algorithm 4: First-order parameters. Let the first-time derivative of u to be non zero. For simplicity, assume that both a and b (the coefficients of x'' and x' to make symbol manipulation easier) to be constant and hence disappear on first differentiation. The extra information needed for u' to be non zero is extracted from the fifth time derivative of the trajectory.

$$a.x'' + b.x' + x = u \quad (31)$$

Differentiate wrt to t and assume u' is non zero to give:

$$a.x''' + b.x'' + x' = u' \quad (32)$$

Differentiate again and set u'' = 0 gives:

$$a.x'''' + b.x''' + x'' = 0 \quad (33)$$

Divide by x''' to isolate b:

$$a.x''''/x''' + b + x''/x''' = 0 \quad (34)$$

Differentiate again to eliminate b:

$$a.(x'''''.x'' - x''''^2)/x'''^2 + (x'''^2 - x'' . x'''')/x'''^2 = 0 \quad (35)$$

Re-arranging for a gives:

$$a = (x'' . x'''' - x'''^2)/(x''''' . x''' - x''''^2) \quad (36)$$

Solve for *b* by substituting a from equation 36 into equation 34:

$$b = -x''/x''' - a.x''''/x'''$$

Substituting for a and manipulating gives:

$$b = (x''.x''''' - x'''.x'''')/(x'''''^2 - x''' . x''''') \quad (37)$$

We can now substitute these values for *a* and *b* into equation 31 to solve for u,

$$u = a.x'' + b.x' + x$$

Results and Discussion

Algorithm 3 Using Constant Parameters: This algorithm uses a single time point but two further time derivatives compared to Algorithm 1. The filter cascade is increased by one again to provide a continuous estimate of the fourth time derivative x''''. The separation problem disappears altogether now to provide a continuous estimate of all parameters at each point on the trajectory. Program 3 [Al-Dabass et al., 1999a] was run, and the result of the estimation shows fast and accurate convergence.

Discussion

Estimated values for constants parameters were close to the desired set values. The derived algorithms estimated ω, ζ, and u for a good range of values: ω from 1 to 10, ζ between +/- (0.01 to 1), and u between +/- (0.5-40), and gave accurate estimates. Estimation errors decreased as ω increased, particularly for small ζ (less than 0.5), where oscillation provided wide variation in the variables to decrease errors. The differences between the (simulated) system time derivatives (x, x', and x'') and their estimates from the filter cascade depended on G (the cut-off frequency): high G provided more accurate estimation of derivatives but made the algorithms prone to noise and vice versa. Another disadvantage of high G from the simulation point of view is that simulation time increased considerably due to the integration routine adapting to ever smaller steps. The algorithms provided progressively faster convergence, with Algorithm 3 being the fastest to converge.

Results

Further results to those shown earlier are obtained to test the case when one or two of the input causal parameters were changing. Figure 13 shows the results when u is changing and the result of tracking it using algorithms 3 a—d 4—algorithm 3 results showing large oscillations while those for algorithm 4 show much smoother tracking. Figure 14 shows the results of both algorithms tracking the other input **ω**. Again algorithm 4 is producing a far smoother estimate than the large oscillatory output of algorithm 3.

Comments

For a given range of parameters, the algorithms worked well, being able to estimate the two

*Figure 13. The **u** causal parameter (smooth trace) being tracked using algorithm 3 (top sinusoid-like trace) and algorithm 4 (gentle wavy immediately below u trace); the black trace is the node output trajectory*

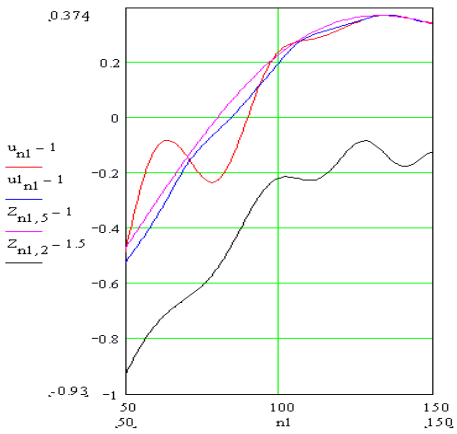
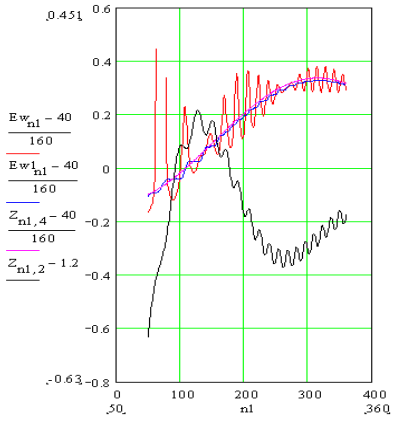


Figure 14. The ω causal parameter (smooth trace) being tracked using algorithm 3 (top jagged trace) and the new algorithm (entwined with ω trace); the bottom trace is the node output trajectory



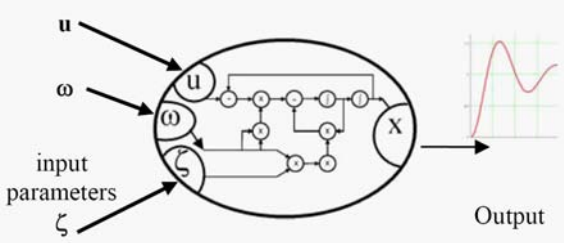
causal parameters u and omega with their temporal behaviour—that is, track them while they are changing. The first-order algorithm worked better than the constant one; Figure 13 shows a comparison of the two algorithms tracking omega. Algorithms of higher order than first showed marginal improvement, but in certain cases showed a deteriorating behaviour; Figure 14 shows a third-order algorithm deviating quite markedly from the true trajectory compared to a first-order algorithm. This is likely to be due to an accumulation of errors in higher derivative values used in the former algorithm.

KNOWLEDGE MINING FOR ECONOMIC SIGNAL PROCESSING APPLICATIONS

A special sixth-order dynamical model is proposed to simulate the behaviour of complex signals. The model consists of a two-layer hierarchy of second-order dynamics, two of whose parameters are themselves second order. Given the trajectory of the actual complex signal, a recurrent hybrid algorithm is derived to estimate the parameters of the model. Results show good performance of the algorithm in tracking the model parameters online. Suggestions for future directions are given.

The algorithms derived earlier combine estimates of a given trajectory time derivatives, using data from several points on the trajectory, with explicit static non-linear functions to provide continuous parameter estimation in real time. For time varying parameters, the time separation between the points on the trajectory directly influences the estimation accuracy. This due to the assumption of constant parameters used in the derivation is no longer valid, and accuracy deteriorates with increasing rate of parameter variation. This is termed the separation effect, which can only be eliminated if all data is obtained from a single time point. An algorithm was derived and proved, as expected, to be the most successful in coping with high rates of parameter variation. Accurate track-

Figure 15. Hybrid integral-recurrent net model of a second-order system



ing of parameters when two of the parameters were varying simultaneously still proved difficult. The constant parameters assumption in the derivation is seen as the fundamental cause here.

Sixth-Order Models of Compound Signals

Hierarchical Second-Order Models: The signal trajectory is more complicated than can be represented by a simple second-order differential model. In this case each parameter may itself be modelled as having dynamics, which may or may not be oscillatory. One such case is where two of the three parameters have second-order characteristics, as shown in Figure 15.

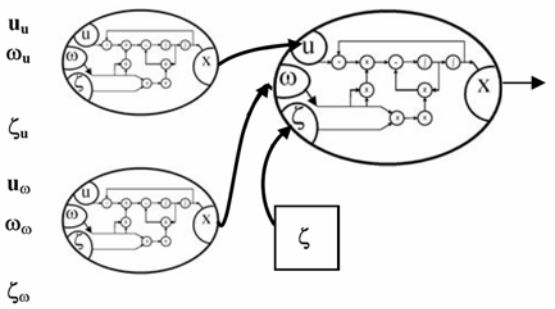
The second-order model of a node in a given layer in the hierarchy is given by:

$$\omega^{-2}\, x'' + 2.\, \zeta.\omega^{-1}.x' + x = u$$

To model complicated signals, let both u and omega have their own second-order dynamics. The input **u** is the output of the following second-order system:

$$\omega_u^{-2}\, \mathbf{u''} + 2.\, \zeta_u.\omega_u^{-1}.\mathbf{u'} + \mathbf{u} = u_u$$

Figure 16. Two of the input parameters of the signal are time varying and modelled with second-order dynamics.



The natural frequency **ω** is the output of the following second-order system:

$$\omega_\omega^{-2}\, \boldsymbol{\omega''} + 2.\, \zeta_\omega.\omega_\omega^{-1}.\boldsymbol{\omega'} + \boldsymbol{\omega} = u_\omega$$

Thus the behaviour trajectory is generated by the following sixth-order vector differential equation (using Runge Kutta in Mathcad for this example) (see Figure 17).

Where x1 and x2 represent the x and x', x3 and x4 represent u and u', and x5 and x6 represent ω and ω' respectively. To generate the trajectory shown in Figure 18, the following values were used: for the u subsystem, u started from 0 aiming at uu=1 at a rate of $\omega_u = 5$ rad/s with $\zeta_u = 0.3$. For the ω subsystem, ω started from 4 rad/s aiming at $u_\omega = 32$ rad/s at a rate of $\omega_\omega = 4$ rad/s with $\zeta_\omega = 0.1$. The resulting compound trajectory of x (red), together with the trajectories for u (top dotted) and ω (bottom dotted) are shown in the graph in Figure 18.

Results and Discussion

Mathcad routines were set up to generate the input u as a second-order system with its own parameters of natural frequency, damping ratio, and input. The input subsystem damping

Figure 17. Simulation vector of a sixth-order trajectory (top two rows) with u (rows 3 and 4) and omega (rows 5 and 6) of the signal having second-order dynamics

$$D(t, x) := \begin{bmatrix} x_2 \\ x_5 \cdot x_5 \cdot x_3 - 2 \cdot z \cdot x_5 \cdot x_2 - x_5 \cdot x_5 \cdot x_1 \\ x_4 \\ (wu \cdot wu \cdot uu) - 2 \cdot zu \cdot wu \cdot x_4 - wu \cdot wu \cdot x_3 \\ x_6 \\ (ww \cdot ww \cdot uw) - 2 \cdot zw \cdot ww \cdot x_6 - ww \cdot ww \cdot x_5 \end{bmatrix}$$

ratio was set to 0.05 to generate an oscillatory behaviour for long enough to test the parameter tracking algorithm thoroughly. The frequency of the input was set to 10 radians per second; the frequency of the main signal, on the other hand, started from 20 and aimed at 80 with a peak of about 130 radians. The derivative generation cascade was increased by one to produce the fifth time derivative.

Tracking Two Parameters: For a given range of parameters, the algorithm worked well, being able to estimate the two input parameters u and omega with their time varying behaviour—that is, track them while they are changing. The first-order algorithm worked better than the constant one; Figure 19 shows a comparison of the two algorithms tracking omega. Algorithms of higher order than first showed marginal improvement, but in certain cases showed a deteriorating behaviour; Figure 20 shows a third-order algorithm deviating quite markedly from the true trajectory compared to a first-order algorithm. This is likely to be due to an accumulation of errors in higher derivative values used in the former algorithm.

CONCLUSION

A model for hybrid logic nets was put forward to model the complex behaviour of economic systems. To estimate the values of the causal parameters, a number of parameter knowledge mining algorithms were presented. Two of the algorithms used multiple points from the trajectory, three for the first algorithm and two points for the second. Two single-point algorithms were presented: one that assumed constant parameters and used higher time derivatives of the trajectory (up to fourth), and a second

Figure 18. Simulated trajectory of a complex signal of a sixth (oscillating trace), with two variable inputs: u (top) and omega (bottom)

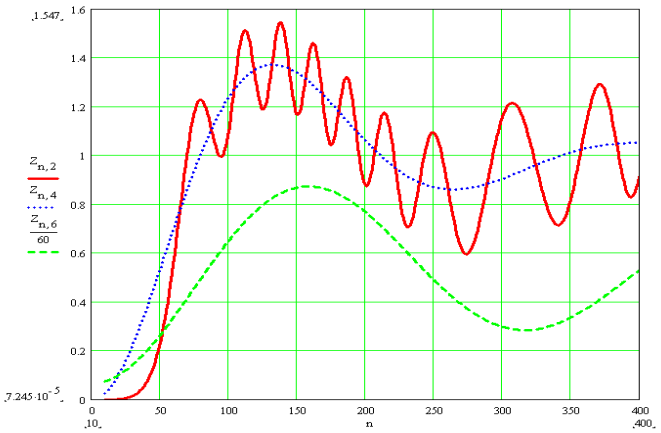


Figure 19. Estimated Omega assuming constant parameters (Ew, very jagged trace) and first-order parameters (Ew1, jagged trace) compared to actual (smooth trace)

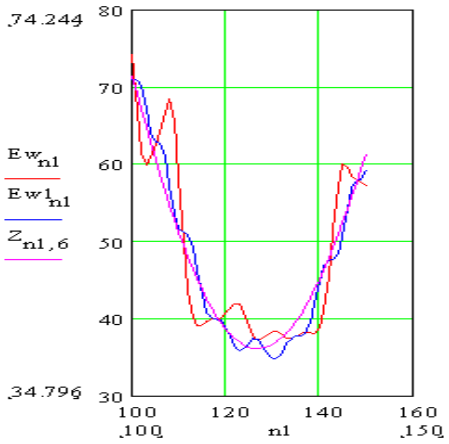
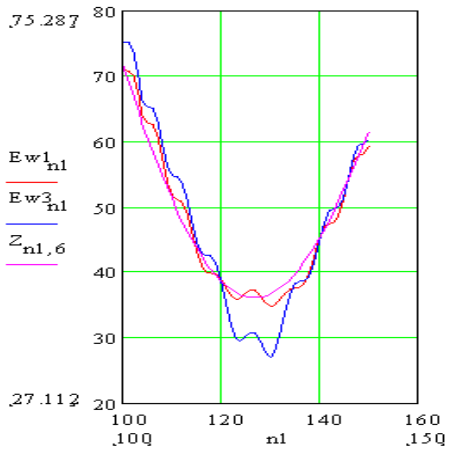


Figure 20. Deterioration of accuracy of estimated Omega with higher-order algorithms: third-order estimate (Ew3, very jagged trace) is worse than first-order estimate (Ew1, jagged), actual (smooth)



algorithm that used additional information from a fifth time derivative of the trajectory to allow one of the parameters, the input parameter u, to have a non zero first-order time derivative.

The fourth algorithm was tested for its ability to track the input parameter for a reduced order model. The test involved the generation of a lightly damped second-order recurrent net. The results showed the algorithm maintaining good tracking over an extended period of time. This algorithm proved to be far superior to the third algorithm, which relied on the assumption of constant input in the derivation.

REFERENCES

Al-Dabass, D., Zreiba, A., Evans, D. J., & Sivayoganathan, K. (1999a). Simulation of three parameter estimation algorithms for pattern recognition architecture. *Proceedings of the UK Simulation Society IM'99 Conference* (pp. 170-176), St Catharine's College, UK.

Al-Dabass, D., Zreiba, A., Evans, D.J., & Sivayoganathan, K. (1999b, October 29). Simulation of noise sensitivity of parameter estimation algorithms. *Proceedings of the Simulation'99 Workshop* (pp. 32-35), London.

Al-Dabass, D. (2001a, July 31). A Kalman observer computational model for metaphor based creativity. *Proceedings of ICCBR2001,* Simon Fraser University, Canada. Retrieved from http://ducati.doc.ntu.ac.uk/uksim/dad/webpagepapers/Paper-1.doc

Al-Dabass, D., Zreiba, A., Evans, D. J., & Sivayoganathan, K. (2002a). Parameter estimation algorithms for hierarchical distributed systems. *International Journal of Computer Mathematics, 79*(1), 65-88.

Al-Dabass, D., Evans, D. J., & Sivayoganathan, K. (2002b, May 12-17). Derivative abduction using a recurrent network architecture for parameter tracking algorithms. *Proceedings of the IEEE 2002 Joint International Confer-*

ence on Neural Networks, World Congress on Computational Intelligence (pp. 1570-1575), Hawaii.

Al-Dabass, D., Zreiba, A., Evans, D. J., & Sivayoganathan, K. (2002c). Parameter estimation algorithms for hierarchical distributed systems. *International Journal of Computer Mathematics, 79*(1), 65-88.

Al-Dabass, D., Evans, D. J., & Sivayoganathan, K. (2002d, May 12-17). Derivative abduction using a recurrent network architecture for parameter tracking algorithms. *Proceedings of the IEEE 2002 Joint International Conference on Neural Networks, World Congress on Computational Intelligence* (IJCNN02) (pp. 1570-1574), Honolulu, Hawaii.

Al-Dabass, D., Evans, D. J., & Sivayoganathan, K. (2002e, August 7-10). A recurrent network architecture for non-linear parameter tracking algorithms. *Proceedings of the 2nd International Conference on Neural, Parallel and Scientific Computations,* Atlanta.

Al-Dabass, D. (2002f, August 2-6). Modelling the complexity of concept dynamics. *Proceedings of the 47th Meeting of the International Society for the Systems Sciences,* Shanghai, China.

Al-Dabass, D., Evans, D. J., & Sivayoganathan, K. (2003). Signal parameter tracking algorithms using hybrid recurrent networks. *International Journal of Computer Mathematics, 80*(10), 1313-1322.

Bailey, S., Grossman, R. L., Gu, L., & Hanley, D. (1996). A data intensive approach to path planning and mode management for hybrid systems. In R. Alur, T. A. Henzigner, & E. Sontag (Eds.), *Hybrid Systems III, Proceedings of the DIMACS Workshop on Verification and Control of Hybrid Systems* (pp. 485-495). Berlin: Springer-Verlag (LNCS 1066).

Baltagi, B. (2005). *Economic analysis of panel data.* New York: Wiley.

Berndt, D. J., & Clifford, J. (1996). Finding patterns in time series: A dynamic programming approach. In U. M. Fayyad, G. Piatetsky-Shapiro, P., Smyth, & R. Uthurusamy (Eds.), *Advances in knowledge discovery and data mining* (pp. 229-248). Boston: AAAI Press/ MIT Press.

Bovet, D. P., & Crescenzi, P. (1994). *Introduction to the theory of complexity*. Englewood Cliffs, NJ: Prentice-Hall.

Cant R., Churchill, J., & Al-Dabass, D. (2001). Using hard and soft artificial intelligence algorithms to simulate human go playing techniques. *International Journal of Simulation, 2*(1), 31-49.

Cawley, P. (1984). The reduction of bias error in transfer function estimates using FFT-based analysers. *Journal of Vibration, Acoustics, Stress and Reliability in Design,* 29-35.

Close, C. M., & Fredrick, D. K. (1994). *Modelling and analysis of dynamic systems* (2nd ed.). New York: John Wiley & Sons.

Dewolf, D., & Wiberg, D. (1993). An ordinary differential-equation technique for continuous time parameter estimation. *IEEE Transactions of Automated Control, 38*(4), 514-528.

Doyle, E. (2005). *The economic system.* New York: Wiley.

Gersch, W. (1974). Least squares estimates of structural system parameters using covariance function data. *IEEE Transactions of Automated Control, 19*(6).

Kailath, T. (1978). *Lectures on linear least-squares estimation.* CISM Courses and Lectures No. 140. New York: Spring-Verlag.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of SAME: Journal of Basic Engineering (Series D), 82,* 35-45.

Koop, G. (2005). *The analysis of economic data.* New York: Wiley.

Mannila, H., Toivonen, H., & Verkamo, I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery, 1,* 259-289.

Man, Z. (1995). Parameter-estimation of continuous linear systems using functional approximation. *Computers and Electrical Engineering, 21*(3), 183-187.

Papazoglou, M., & Ribbers, P. (2005). *E-business: Organizational and technical foundations.* New York: Wiley.

Ren, M., & Al-Dabass, D. (1995). An associative memory artificial neural network system. *Proceedings of the European Chinese Automation Conference* (pp. 91-96), London.

Ren, M., Al-Dabass, D., & Su, D. (1996). A three-layer hierarchy for representing Chinese characters. *Proceedings of Expert Systems '96, 6th Annual Technical Conference of the BCS Specialist Group on Expert Systems* (pp. 137-146), Cambridge, UK.

Ren, M., & Al-Dabass, D. (2001). Simulation of fuzzy possibilistic algorithms for recognising Chinese characters. *International Journal of Simulation, 2*(1), 1-13.

Samuelson, W. (2005). *Managerial economics.* New York: Wiley.

Schank, R. C., & Colby, K. M. (Eds.). (1973). *Computer models of thought and language.* San Francisco: Freeman & Co.

Seveance, F. L. (2001). *Systems modelling and simulation, an introduction.* New York: John Wiley & Sons.

Chapter XIII
The Grid for Nature-Inspired Computing and Complex Simulations

Riccardo Boero
University of Surrey, UK
Università di Torino, Italy

ABSTRACT

This chapter deals with the usage of grid technologies for nature-inspired algorithms and complex simulations. After shortly introducing the grid and its technological state of the art, some features are pointed out in order to set the boundaries of the applicability of such new technology to the matters of interest. Then two paragraphs show some possible usages of grid technologies. The first one introduces the master-worker paradigm as a conceptual and technological scheme that helps in solving issues related to dynamic optimisation via nature-inspired algorithms and in exploring the parameters space of complex simulations. The following paragraph concerns two other points: the possibility to distribute agents of agent-based simulations using multi-agent systems; and the boundaries, architectures, and advantages in distributing parts of complex simulations which are heavy from the computational point of view. The chapter, as a whole, acts as a guide presenting applicative ideas and tools to exploit grid technological solutions for the considered purposes.

INTRODUCTION

Among the many novelties developed in ICT, the *grid* has emerged as one of the most radical and promising, aiming to modify the established standards in computing.

The basic idea of the grid is, as well known, the possibility to develop and implement a widespread network of computing services, easily accessible from anywhere, taking as a model the structure of power grids for distributing electricity. In such kind of information infra-

structure, it should not be relevant where the computing power is geographically located and computing therefore becomes a commodity.

The grid is a new paradigm that makes possible already known computing paradigms such as parallel and distributed computing in a more general framework based on common standards and on grid services.[1] Such services are not just useful for performing better computations in terms of time, but also to share any kind of resources (for instance databases and scientific tools) among any kind of virtual organisation.

While the social and organisational impact of the new concepts implied by the grid (in particular the transformation of computing in a commodity) are still to be explored and they actually belong more to possible future scenarios than to real ones, it is worth noting the state of the art of the technologies developed in the field.

Since the main ideas of the grid have been presented to the scientific community, a decade characterised by many research projects has passed away and it is nowadays possible to exploit the first elements of such infrastructure: many grids have been developed by different scientific communities, and a considerable amount of software is available to develop new grids and services.

Thus, grid computing is not a dream but something real, though it is worth adding that the usage of the grid is nowadays not *easy*. In fact, referring to the current issues in exploiting the new technology, the actual phase of development and evolution must be carefully considered: after less than a decade of development, the technology is not usable by every potential end user because of the software available, which is mainly middleware that must be managed by professionals.

But even if the problem of easiness of usage does not affect every potential user (e.g., institutions and organizations where computing departments have human resources with the needed skills), other issues are now preventing the fast diffusion of the grid, mainly its lack of standards and the few fields for which it has been used and thus for which it has shown its sound advantages. These issues are nowadays considered critical and in fact new standards, such as the open grid services architecture—OGSA (Foster, Kesselman, Nick, & Tuecke, 2002; Foster, Kesselman, & Tuecke, 2004) and the open grid services infrastructure—OGSI (Tuecke et al., 2003), and many new projects are under fast development and diffusion.

This chapter considers both available technologies and already started projects, but it is not a generic introduction on the theme[2]: it deals with possible usages of the technology on specific topics.

In fact, while grid technologies can be applied in every field of research and for any kind of purpose characterised by the need of an intensive usage of computing power, here the focus is on the exploitation of the tool in the overlapping fields of nature-inspired computing and complexity, aiming to answer research and organisational questions.

Considering such kind of aim, it is worth underlining how the mentioned fields of research are overlapping and closely connected, and moreover how they call for contributions from several other research fields, from distributed artificial intelligence to operational research.

Moreover, another important issue should be considered: grid technologies, even if thought and designed to radically modify the computing paradigm for all possible kinds of purpose, have been mainly developed and applied in some fields which are intrinsically different from the ones we are talking about.

For instance, the largest part of projects, software, infrastructure, and research funds

spent and developed in Europe in the last years in the field of grid computing relate to the research work of physicists and in particular to experiments that will be made possible by the new collider (the Large Hadron Collider—LHC) that will start to be ready in 2007 at CERN in Geneva.

The need for computing in this case is largely modular: from making available different services and tools through the network to the distribution of the vast experimental data collected, each activity is mainly managed by the execution of several tasks across the grid, and those tasks are completely independent or slightly dependent in the sense that some tasks need the results obtained by others to be executed.

In our fields of interest (i.e., nature-inspired computing and applications coming from the field of complexity science), that is generally not the case because computing tasks are strictly dependent on one another. In other words, in the largest part of the cases now considered by the grid, computational tasks interact just at the beginning and at the end of the execution, while in the direction of applying such technologies for our purposes, the need to have computing tasks interacting along their execution must be considered.

Summarising the state of the art in grid computing, it raises two main issues concerning researchers and professionals wanting to apply it to the topics here focussed:

1. Grid technological solutions are still in a stage of development: depending on that, the knowledge to exploit them is highly expensive to learn because of the lack of standardisation and because of the lack of efforts in preparing *end user* tools; the development of such technologies is led by physicists, the needs of whom are just partially common to ours, and thus very few projects already developed can be considered useful examples on how to solve our needs, and consequently, an even more expensive effort is due to apply the technology.
2. A relevant amount of grid technologies (i.e., the ones that deal with sharing computational resources) are characterised by aiming to distribute computing weights on networked resources, separating them in different and almost independent tasks, capable of interacting with each other shortly and mainly by coordination means, and that structure is possible because of the modularity of the kind of computations that are distributed: when considering exploiting such technology for highly complex computations, neither modular nor separable, the implementing paradigm of the computations must be changed and adapted to the new programming and computing environment, or new approaches to grid computing must be developed to fix the missing characteristics of the mainstream in grid projects.

For these reasons, the following paragraphs explain two kinds of approaches: the first one shows how a computing paradigm already available in the technological universe of the grid can be effectively exploited to solve some particular problems mainly in the field of nature-inspired algorithms, but sometimes also in complex simulations, while the second one shows other possible means to exploit the grid in complex simulations, pointing out the constraints now posed by the technology.

THE MASTER-WORKER PARADIGM

The master-worker paradigm is based on a simple computing architecture: there is one

single node of the network that has the role of master—that is, to coordinate other nodes and to give tasks to be executed—while the other nodes are workers, in the sense that they execute received tasks. Each worker can be a single computer or a pool of them. The communication among master and workers happens at the beginning when tasks are sent from master to workers, and at the end when each worker, as soon as it has terminated the task, communicates the results to the master. There is not communication among workers.

The master-worker architecture is one of the simplest that can be found in the field of grid computing, and it is not very different from the ones which aim to efficiently provide specific computing services, such as NetSolve (Casanova & Dongarra, 1997) and NEOS (More, Czyzyj, & Mesnier, 1998). The mentioned examples are a particular case of the computing paradigm we are talking about: their aim is to distribute the computing power of networked computers to users who need a specific kind of service—that is, the solution of optimisation problems (NEOS) or generic computing problems (NetSolve)—but all must be rigorously formalised in a mathematical language.

The basic idea of these two projects was the building of an effective service for optimisation, where end users can write the mathematical problem with their software of choice (e.g., Mathematica and MATLAB) and then access the service as if it were unique: users do not know and do not care about the underlying implementation and computing infrastructure, but they ask a service to a provider that effectively answers managing the distribution of networked resources and so forth.

Those examples exploit the master-worker paradigm because there is a master computer that coordinates and distributes the computing weights among the networked resources, being able to perform complicated algorithms of optimisation in a very short time.

In other words the mentioned examples do not look immediately as instances of the master-worker paradigm but dedicated services, while their implementation exploits the paradigm.

Our claim here is that the master-worker paradigm can be similarly exploited in the fields of nature-inspired computing and of complex simulations, and particularly for two purposes.

Considering for instance that NEOS is a service for mathematical optimisation, a first application is to build a similar service implementing the kinds of optimisation provided by nature-inspired optimisation algorithms such as genetic algorithms, neural networks, and so forth.

The master-worker model in this context appears to be a promising enhancement of the parallel search process of the solution space typical of many nature-inspired algorithms, allowing advantages from the point of view of execution speed and perhaps results quality.

The problem, in this case, is the formalisation of the optimisation problem: while in mathematical language it is possible to simply express problems of optimisation, nature-inspired optimisation algorithms are particularly useful, not just when standard mathematical optimisation algorithms are not effective because of the definition of the problem space or because of the computational weight, but even when problems cannot be expressed in mathematical language.

Thus, even if a general service just for function optimisation exploiting genetic algorithms can be conceived and made (e.g., Nakada, Matsuoka, & Sekiguchi, 2004), it is not possible to build a generic service for Genetic Algorithms, but it is possible to build several specific services exploiting genetic algorithms to solve specific problems.

This kind of application is the first one in which the master-worker paradigm appears deeply effective to solve computational problems in our fields of interest.

Imagine, as an example, of being interested in the development of a complex simulation of the productive organisation of a manufacturing plant. After having built a model capable of describing all the manufacturing phases, flows and time lags of materials, and so on, exploiting the most appropriate modelling paradigm for the case (for instance making a process simulation), further suppose that the problem of interest concerns the organisation and management of the production flow in order to optimise the plant output, considering the complexity and variety of the flow of orders that are demanding different versions and different lots of goods.

If the technological complexity of the manufacturing phases does not allow the finding of a simple rule to optimise the organisation, different optimal solutions can be dynamically discovered using genetic algorithms combined with the simulation: several organisational solutions can be explored and evolved by the algorithm to find the organisational solution that optimise the simulated plant with a particular set of orders.

If the flow of orders is frequently updated and modified, the problem of dynamic optimisation can be too heavy even for a powerful computer, in particular if the optimisation system has to organise the production line in real time and thus the computing speed becomes a relevant constrain for the productive exploitation of the model.

With a small cluster of computers, highly efficient outcomes will be reached by distributing the simulations among them, having the master managing the genetic algorithm and the workers simulating the possible solutions to evaluate their fitness values. The grid moreover allows the possibility of sharing resources coming from different organisations, and thus the need for a cluster is not to be intended as the need of a proprietary and dedicated computing system, but just to have access to the grid. The different instances of *workers* applications can in fact be spread and distributed efficiently through the technology of grid services.

The kind of computational improvement that such solution can bring is easy to be evaluated: the theoretically bounded maximal improvement conceivable is the sum of the computational power of the nodes exploited—that is, if each computer has the same characteristics, to halve optimisation time when using two computers instead of one.

But in reality the amount of improvement is slightly less than the theoretical maximum because of network communication between master and workers that can slow down the whole optimisation process.

In recent experiments conducted with the colleagues (Gianluigi Ferraris, Matteo Morini, and Michele Sonnessa) of the Networked Modelling Team at the University of Torino on a complex model of a textile plant, we have found how some small modifications to the standard genetic algorithm can help. In fact, creating a memory to avoid re-exploring already met productive configurations and condensing communication to transmit, each time, multiple possible solutions to be explored by workers means reaching better improvement in computational time (for instance, the doubling of computational power that would mean a theoretical bound of 50% improvement, in our tries meant a 41% improvement in real computational time).

A second case in which the master-worker paradigm helps is the problem of parameters sweeping. In the fields of research and design, it is more common everyday to exploit models that consider a large amount of parameters; and while in standard mathematical models the parameters space is generally characterised by properties that let the user simplify the problem

of exploring that space, in complex simulations that is not possible because of the intrinsic complexity of the model.

In those cases, it would be necessary to run the simulation with all possible configurations of parameters creating a very long and computing-intensive task.

With a master creating parameters configuration, sending them to workers, and collecting the results, such process becomes fast and possible.[3]

In conclusion, the master-worker paradigm appears to be effective for two purposes: optimisation led by nature-inspired algorithms with distributed fitness evaluation, and parameters space sweeping of complex simulations.

The technical solutions for implementing the master-worker computing architecture are many: it is possible to exploit middleware that enables such kinds of computing tasks, and among them one of the most reliable and diffused solution is the CONDOR project[4] (used with NIMROD for distributed parametric modelling[5]), or it is possible to easily develop specific solutions from scratch, for instance exploiting high-level programming libraries such as Java-RMI, XML-Rpc, Corba, or recent grid middleware such as the Globus Toolkit[6] and grid services, and so forth.

The Distribution of Agents and Databases

Available grid technologies are useful for the specific but very relevant purposes just shown in the paragraph above. The exploitation of the new computing infrastructure for other applications related to nature-inspired computing and complex simulations is a main issue for the development of the grid and even for enlarging the universe of possibilities of our research techniques, but such a dream is particularly far off for scholars and practitioners of complex simulations.

Whilst for people involved in exploiting nature-inspired computational algorithms the grid gives the master-worker paradigm and many tools to realise it, those interested in realising complex simulations on networked resources faces several problems, the most important of which is the difficulty to exploit intra-execution communication (a feature needed because of the lack of modularity in computations).

The only technology now available and diffused to achieve that purpose is the so-called message passing interface—MPI (for an overview, see Foster, Kohr, Krishnaiyer, & Choudhary, 1997; for an application in grid context, see Bal, Casanova, Dongarra, & Matsuoka, 2004, pp. 482-483). It is a low-level way of programming that enables the possibility of accessing memory areas located in different computers, achieving the possibility of transmitting information from a task in a computer to another in a different machine during execution.

But the implementation of MPI in a simulation is problematic: past years of evolution in the field of software for simulation have followed the criterion of making libraries and platforms that facilitate end users in the realisation of simulations, by hiding low-level details and adopting easier solutions like multi-platform and half-interpreted languages such as Java; therefore the need to go back to low-level programming means several disadvantages.

One is the impossibility to exploit any of the model realised in past years. A second disadvantage is the uselessness of the software now widespread in the community. The third and last one is the need for deeper programming skills.

In other words, the MPI solution is too expensive to be accepted because it generates more practical problems than the ones it solves, and to be adopted it implies relevant costs.

Is it thus completely impossible to generally exploit available grid technologies to improve

computing performances in the field of complex simulations (except, of course, the already discussed case of parameters space sweeping)?

The answer should be positive, and this chapter should end with the hope of future technological developments. But even if that positive answer is largely correct, we present here two possible solutions that base their technological feasibility on exploiting tools developed outside the research on the grid, that still need to be fully explored and diffused in the community, but that, despite the premises, provide advantages similar to the ones of grid technologies.

Firstly, considering just a particular kind of complex simulations—such as agent-based models (ABMs) and not process simulation or system dynamics simulation, it is interesting to note how one of its overlapping research fields, that of multi-agent systems (MASs) belonging to computer science, has developed some tools that can be really useful.

MASs in fact have been developed in recent years by computer scientists aiming to exploit concepts from distributed artificial intelligence to make new software services capable of exploiting networked resources. The term agent in MAS refers to software agents capable of reaching some goals requested by the user, interacting with each other, finding and exploiting resources where they are. An agent, for instance, in a MAS is a piece of software that manages the personal agenda of an individual and matches its entries with the ones of the people the individual wants to meet, even if they belong to different organisations.

Remembering that in ABM the term agent has a complete different meaning, and recalling the fact that the unity of the system modelled is a single one and not a collection, it is worth noting how agents in MAS are generally capable of doing what practitioners of ABM would like to do in grid environments—that is, intra-execution communication among network distributed software.

Exploiting available MAS platforms (for instance the Magenta Multi-Agent Platform[7] or JADE[8]) for distributed agent-based modelling is thus a possibility, avoiding the costs of using low-level solutions (e.g., MPI) or the time needed to develop new platforms from scratch, but it implies some risks due to the adoption of a few important concepts from that field.

One of the most important issues that the exploitation of MAS platform implies for ABM is the fact that MASs are conceived as open systems and that, for guaranteeing the interoperability between agents developed in different times and locations, the communication infrastructure used by MAS agents is strictly formalised.

Agents in MAS have formal ontologies and semantics, concepts of which ABM practitioners are generally unaware (in the sense that they are not used to implementing such concepts in their agents).

In other words, in MAS, even if new agents are developed with completely new functions and tasks, they can communicate each other and with older ones as long as they share the ontology and the semantics of communication, and this simple rule makes MAS open to development and new functionalities.

From the ABM point of view, this technological constrain poses the disadvantage of changing the way systems are usually modelled (developing formalised communications even in ABM), but permits the exploitation of MAS platforms and of distributed computation, besides the possibility to fruitfully interact with the computer science community.

Moreover, getting back to the proposed solutions of further exploitation of grid technologies for complex simulations, a more general scheme can be proposed.

It is in fact possible to exploit the grid to delocalise the *less dynamic* parts of simulations, not only agents. A good example could be the case of an agent-based simulation that considers geographical data. The integration of an ABM with a GIS (geographical information system) tool can in fact be made in several ways. A widespread approach is the one called *coupling* (see Maguire & Batty, 2005, for a comprehensive introduction about GIS and modelling). In such kinds of models, the simulation does not contain the geographical data, but is executed along with a GIS application that provides the needed information on run time.

An example can help in making the point clearer: imagine preparing a simulation of transport companies delivering goods of different kinds and weights via different means. The territory considered is wide; transportation companies are spread in different locations; the transportation infrastructure is complex, determining delivery costs, timing, traffic queues, and so forth.

While the simulation deals with the modelling of transportation companies and of their decision-making process (i.e., which transportation mean should be used and with which route, considering the quality and the quantity of the good and the target location), the geographical information and constraints are held by the GIS application, which provides data when asked by the simulation.

The coupling solution is often adopted because the task of modelling geographical data *inside* a simulation would be too expensive, in terms of programming work and in terms of computational weight.

Geographical data and constraints are a good example of a part of a complex simulation that can be distributed on the grid through grid services or just by accessing a simple and dedicated remote service: a GIS is a special kind of database that is a service not directly involved in the simulation, posing just constraints or determining the parameters space. Any service with such characteristics could be a good candidate for network distribution, improving the simulation performance in terms of time of execution and exploiting specialised software for each task (in our example, a simulation platform for the model and a GIS server as the database service).

CONCLUSION

In this chapter we have briefly introduced some important grid concepts and debated them, focusing on their possible exploitation in the fields of nature-inspired computing and of complex simulations.

With examples we have suggested the possible melting pot between concepts and technologies: the distribution of computational weights through grid services (and via grid-based architectures and implementations) can increase the performance of intrinsically parallel algorithms such as many inspired by nature and of complex simulations.

An improvement in the field of complex simulations can be reached even by adopting other technologies (generally not considered as belonging to the grid) for distributing pieces of the model on networked resources. The technological implementation is easier when the distributed elements are only slightly interacting with the others. The performance is better when the distributed elements are the heaviest from the point of view of computational weights. In other words, because of the technological state of the art in grid computing, the "rule" for exploiting networked resources in our fields of interest is to distribute the parts with less interaction and more computational weight.

Although a chapter dealing with the grid inevitably comes with an expiration time due to the fact that the grid is now under a fast process of development, the thoughts presented here along with the technical solutions have a more enduring nature: nature-inspired computing and complex simulation research areas are, in a similar way, still under strong development and diffusion, and their methodological implications are not still completely clear and shared. If the adoption of a new tool in a research program is never a neutral choice because it introduces a new language and pragmatics, the opposite is also true: new ideas can be introduced in the tool when it meets a scientific community that has not developed it.

For the grid case, it is the same, but with the advantage of its state of development. Our community is surely not the main one involved in the development of the computing infrastructure of the future, but it is an important opportunity to work towards finding the tool to fulfil our needs with its promising potential.

ACKNOWLEDGMENT

The research work presented here has been funded by the Progetto Lagrange—Fondazione CRT and by the research centre C.S.P., in Torino, Italy. The author also thanks the L.I.A.S.E.S. at the University of Torino for its technical support and colleagues Gianluigi Ferraris, Matteo Morini, and Michele Sonnessa for their help and enlightening comments. The usual disclaimers apply.

REFERENCES

Abramson, D., Sosic, R., Giddy, J., & Hall, B. (1995). Nimrod: A tool for performing parametrised simulations using distributed workstations. *Proceedings of the 5th IEEE International Symposium on High Performance Distributed Computing* (pp. 112-121). Los Alamitos, CA: IEEE Computer Society Press.

Bal, H., Casanova, H., Dongarra, J., & Matsuoka, S. (2004). Application-level tools. In I. Foster, C. Kesselman, & S. Tuecke (Eds.), *The grid: Blueprint for a new computing infrastructure* (2nd ed., pp. 463-490). San Francisco: Morgan Kauffmann.

Bellifemine, F, Caire, G., Poggi, A., & Rimassa, G. (2003). JADE, a white paper. *Exp in Search of Innovation, 3*(3), 6-19.

Casanova, H., & Dongarra, J. (1997). NetSolve: A network server for solving computational science problems. *International Journal of Supercomputer Application and High Performance Computing, 11*(3), 212-223.

Edwards, P., Preece, A., Pignotti, E., Polhill, G., & Gotts, N. (2005). Lessons learnt from deployment of a social simulation tool to the semantic grid. *Proceedings of the 1st International Conference on eSocial Science*, Manchester UK.

Foster, I., Kohr, D. R., Krishnaiyer, R., & Choudhary, A. (1997). A library-based approach to task parallelism in a data-parallel language. *Journal of Parallel and Distributed Computing, 45*(2), 148-158.

Foster, I., Kesselman, C., & Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications, 15*(3), pp. 200-222.

Foster, I., Kesselman, C., Nick, J., & Tuecke, S. (2002. *The physiology of the grid: An Open Grid Services Architecture for distributed systems integration.* Open Grid Service Infrastructure Working Group, Global Grid Fo-

rum. Retrieved from www.globus.org/alliance/publications/papers/ogsa.pdf

Foster, I., Kesselman, C., & Tuecke, S. (2004). The open grid services architecture. In I. Foster & C. Kesselman (Eds.), *The Grid: Blueprint for a new computing infrastructure* (2nd ed., pp. 215-258). San Francisco: Morgan Kauffmann.

Foster, I., & Kesselman, C. (2004a). *The grid: Blueprint for a new computing infrastructure* (2nd ed.). San Francisco: Morgan Kauffmann.

Foster, I., & Kesselman, C. (2004b). Concepts and architecture. In I. Foster & C. Kesselman (Eds.), *The grid: Blueprint for a new computing infrastructure* (2nd ed., pp. 37-64). San Francisco: Morgan Kauffmann.

Foster, I. (2005). Globus Toolkit version 4: Software for service-oriented systems. *Proceedings of the IFIP International Conference on Network and Parallel Computing* (pp. 2-13). Berlin: Springer-Verlag (LNCS 3779).

Frey, J., Tannenbaum, T., Foster, I., Livny, M., & Tuecke, S. (2002). Condor-G: A computation management agent for multi-institutional Grids. *Cluster Computing, 5*(3), 237-246.

Gotts, N., Polhill, G., Law, A.N.R., & Izquierdo, I.R. (2003, April 7-11). Dynamics of imitation in a land use simulation. *Proceedings of the AISB 2nd International Symposium on Imitation in Animals and Artefacts* (pp. 39-46), University of Wales, Aberystwyth.

Himoff, J. (2004, December). Magenta technology: Taking multi-agent systems to enterprises. *AgentLink News, 16,* 15.

Levine, D., & Wirt, M. (2004). Interactivity with scalability: Infrastructure for multiplayer games. In I. Foster & C. Kesselman (Eds.), *The Grid: Blueprint for a new computing infrastructure* (2nd ed., pp. 167-178). San Francisco: Morgan Kauffmann.

Maguire, D., & Batty, M. (Eds.). (2005). *GIS, spatial analysis, and modelling.* Redlands, CA: ESRI Press.

More, J., Czyzyj, J., & Mesnier, M. (1998). The NEOS server. *IEEE Journal on Computational Science and Engineering, 5,* 68-75.

Nakada, H., Matsuoka, S., & Sekiguchi, S. (2004). jPoP: *A parallelisation framework for combinatorial optimisation problems*. Retrieved from http://ninf.apgrid.org/jpop

Pignotti, E., Edwards, P., Preece, A., Gotts, N., & Polhill, G. (2004). FEARLUS-G: A semantic grid service for land-use modelling. *Proceedings of the ECAI-04 Workshop on Semantic Intelligent Middleware for the Web & the Grid,* Valencia. Amsterdam, The Netherlands: IOS Press.

Polhill, G., Gotts, N., & Law, A.N.R. (2001). Imitative versus non-imitative strategies in a land use simulation. *Cybernetics and Systems, 32*(1-2), 285-307.

Smarr, L. (2004). Grids in context. In I. Foster & C. Kesselman (Eds.), *The grid: Blueprint for a new computing infrastructure* (2nd ed., pp. 3-12). San Francisco: Morgan Kauffmann.

Thain, D., & Livny, M. (2004). Building reliable clients and services. In I. Foster & C. Kesselman (Eds.), *The grid: Blueprint for a new computing infrastructure* (2nd ed., pp. 285-318). San Francisco: Morgan Kauffmann.

KEY TERMS

Grid: A new computing infrastructure that takes inspiration from power grids and that aims to make available computing services,

such as computer power and data storage, as a commodity accessible from anywhere.

Master-Worker Paradigm: A computing architecture operating on a network. A single computer is called the master and sends jobs to be executed by other computers, called workers. The communication between master and workers generally happens just at the beginning and at the end of each job.

Message Passing Interface: A standard interface that allows access to memory areas of distant computers. Generally implemented by low-level programming languages, it also permits the communication between processes in execution on different computers.

Open Grid Services Architecture: A set of principles defined by standards organisations for developing Grid middleware capable of allowing grid systems based on Web services.

ENDNOTES

1 A grid service is, technologically speaking, a Web service adapted to be compliant with the standard grid architecture presented in Foster et al., 2002, 2004.

2 For an introduction to the grid, see Foster, Kesselman, & Tuecke, 2001; Smarr, 2004; Foster & Kesselman 2004b.

3 Very similarly to a parameters space sweeping application, some projects have been developed not to explore the entire parametric space, but just some configurations considered as research hypotheses to be verified. FEARLUS-G (see Pignotti, Edwards, Preece, Gotts, & Polhill, 2004; Edwards, Preece, Pignotti, Polhill, & Gotts, 2005), the grid extension of the FEARLUS agent-based model (see Polhill, Gotts, & Law, 2001; Gotts, Polhill, Law, & Izquierdo, 2003), is an example of that approach applied to verify hypotheses-simulating land-use dynamics.

4 The homepage of the Condor project is http://www.cs.wisc.edu/condor. For an introduction to the usage of Condor across several networks belonging to different virtual organisations, see Frey, Tannenbaum, Foster, Livny, and Tuecke (2002). For an example of a master-worker implementation via Condor, see Thain and Livny (2004, pp. 304-307).

5 For reference, see Abramson, Sosic, Giddy, and Hall (1995).

6 An introduction to the Globus Toolkit is available in Foster (2005), while an application of the tool is presented in Levine and Wirt (2004). For software and the official documentation, visit http://www.globus.org

7 For a brief introduction to the tool, see Himoff (2004). For complete reference, see http://www.magenta-technology.com.

8 For an introduction to JADE, see Bellifemine, Caire, Poggi, and Rimassa (2003). For complete reference, see http://jade.tilab.com

Section III

Economics

Chapter XIV
Agent-Based Computational Economics

Charlotte Bruun
Aalborg University, Denmark

ABSTRACT

This chapter argues that the economic system is best perceived as a complex adaptive system, and as such, the traditional analytical methods of economics are not optimal for its study. Agent-based computational economics (ACE) studies the economic system from the bottom up and recognizes interaction between autonomous agents as the central mechanism in generating the self-organizing features of economic systems. Besides a discussion of this new economic methodology, a short how-to introduction is given, and the problem of constraining economics as a science within the ACE approach is raised. It is argued that ACE should be perceived as a new methodological approach to the study of economic systems rather than a new approach to economics, and that the use of ACE should be anchored in existing economic theory.

INTRODUCTION

Terming a specific approach to economics agent-based may appear paradoxical. Isn't human behavior the foundation of economics—and shouldn't all economic theory be based on agents behavior in some sense? This, at least, is what conventional economic theory has been claiming since the 1970s. In this introduction we shall argue that agent-based computational economics (ACE) allows agents, and especially their interaction, a more pivotal role than does conventional microeconomics or microfounded theory. There is a difference between microeconomics and agent-based economics: in the latter you are not satisfied with understanding exactly how a single agent acts in economic markets; you are primarily interested in the system view that arises when you observe the interaction between a number of

agents. From observing an isolated agent, it is impossible to foresee what happens when a multitude of agents interact; it cannot be deduced. This adds importance to the computational part. In agent-based economics the computer is not merely used as a giant calculator finding analytical or numerical solutions, but it is used as a central part in a new methodological approach to economics.

THE ECONOMY AS A COMPLEX ADAPTIVE SYSTEM

The economy may be described as a complex adaptive system—that is, a system where complexity arises because of the way a large number of agents interact. Complexity thus stems from the fact that the economy is a large composite system. What we observe as the economy is the result of millions of agents interacting. We know the output of this system as growth rates, inflation rates, unemployment rates, and so forth, but how do we get from the description of our agents to these aggregate magnitudes, and can we say anything about the aggregate magnitudes in their own right?

As other sciences dealing with large composite systems, economics has developed a tradition of dealing with two levels: the microlevel and the macrolevel. Microeconomics takes as its starting point the behavior of individual agents, whereas macroeconomics theorizes about relations between aggregate magnitudes. The problem is that unless one is willing to make very restrictive assumptions, it has proven to be impossible to unite the two levels. This has resulted in assumptions of homogeneity and constructions as the representative agent—a construction that among others has been heavily criticized by Kirman (1992).

The apparent impossibility of uniting micro and macro is particularly crucial in economics since we have developed a tradition of demanding microfoundation of macroeconomics. In reality this means a dismissal of macroeconomics altogether, and an ignorance of the fact that important characteristics of the system may arise in the interaction part. If Keynes' conception of effective demand (Keynes, 1936) is something that arises in the interaction, then the possible lack of aggregate demand is dismissed from the outset since it cannot be microfounded.

Complexity science now offers a way out of this situation. Rather than starting with either a single isolated agent or aggregate magnitudes, complexity science suggests focusing on the interaction between agents. Recognizing that what turns large composite systems into systems and not just collections is the interaction between the parts, it seems apparent to start with the interaction.

Traditional microeconomic models also have interaction, it could be claimed, but this is really pseudo-interaction since here an agent either interacts with the aggregate whole or all are simultaneously interacting with all others (Rosser, 1999). That is, we really do not have interaction taking place in time and space. In general equilibrium models, the pseudo-interaction is obtained by having a controller signal out a price vector to agents, and collecting information on demand and supply given this price vector. Being able to handle heterogeneity, time, and space, the sort of interaction suggested by complexity science comes much closer to the kinds of interaction that is actually taking place in economic systems.

As a consequence of allowing economic agents to actually interact, we must expect our system to display a less predictable behavior—that is, we must expect very complex dynamics. In their introduction to the first Santa Fe workshop on "The Economy as an Evolving Complex System," Arthur, Durlauf, and Lane (1997) characterize complex dynamic systems

as having dispersed interaction among heterogeneous agents acting locally on each other in some space. This characterization must also be expected to hold the other way around—that is, all systems having dispersed interaction among heterogeneous agents must be expected to display complex dynamics, and thus economists must learn to live with complexity.

Besides being a complex dynamic system as described above, the economy is also adaptive. Adaptivity exists on two levels: the system may be capable of adapting to external factors without individuals being aware of this adaptation taking place, and agents may be capable of adapting to the system. The economic system has adaptation on both levels, and the fact that economic agents are intelligent and form opinions about the system of which they are a part adds to the complexity of the system. Using their conception of the system, they form expectations concerning the future developments of the system, and these expectations affect their current behavior. Since the behavior of a given agent affects the outcome of the system as a whole, expectation formation must take into account the behavior of the other agents in the system—that is, expectation formation turns into a guessing game. This makes it difficult for the individual agent to form expectations about the behavior of the system, and it makes it difficult for the economic theorist to model what is going on. Rosser (2001) talks about the uncertainty arising from group dynamics—an uncertainty that cannot be reduced to risk.

What the complexity approach suggests is that in order to understand the economy as a complex adaptive system, we must accept interaction as an activity taking place in time and space without a central controller. Even if agents start out being homogenous, they cannot all be in the same place at the same time, and thus they experience different histories and inevitably evolve into heterogeneous agents.

Schelling (1978) was among the first to apply a complexity approach to social sciences. He argued that economic systems are particularly complex because, besides a large number of locally interacting agents, economic systems are characterized by a lot of relations that must hold in the aggregate, but do not necessarily hold for each individual. This means that fallacies of composition are easily committed within economics. Schelling used the example of musical chairs: no matter how children playing the game behave, there will always be a chair missing—this is a macroproperty of the system that must hold. Therefore we cannot understand the system by observing a single child and aggregate from there. Changes in microbehavior may not have the expected effect—weightlifting and increased aggression in the behavior of children does not affect the macroproperty of the system and thus cannot improve the performance of the children as a group.

In economics we often refer to such discrepancy between parts of the system and the system as a whole as paradoxes; the paradox of thrift is just one among a number of paradoxes. For an individual economic agent, expenses may be larger than income, but this cannot be true of the whole system, thus an attempt by all agents to increase saving is bound to fail. This implies that there is important feedback from the macro to the microlevel, and that the whole may be more (or less!) than the sum of its parts. We cannot all get rich by saving our money. When an agent makes a purchase, he sets off a number of activities because his expense will generate an income somewhere else in the system, and this income may set off additional expenses.

We have argued that the economic system should be perceived as a complex adaptive system, and that within economics, complexity arises from at least three different sources: (1) from the fact that the economy is a large

composite system, (2) from the fact that economic agents adapt their behavior to the system, and (3) from the fact that economics is characterized by a lot of relations that must hold in the aggregate, but need not hold for the individual agent. But isn't all this complexity bad news, one might ask? A high degree of complexity, however, does not mean chaos.

An interesting characteristic of complex adaptive systems is that they have a capacity to self-organize and adapt to changing environments. Despite the lack of a global controller, complex adaptive systems appear to perform quite well. Considering the chain of events a single agent may cause at the supermarket, it is surprising how limited serious market failures in economic systems are. Many economists are puzzled why we have market failures like unemployment at all—but from a complexity perspective, the big puzzle becomes why markets function so well.

Economic theory has tended to put the rationality of economic systems with the agents(market economies work because agents are rational optimizers. This assumption is what makes behavior predictable. Communicating through price signals, this is what makes it possible for a global controller to find the optimal price vector in general equilibrium systems. However, if just one agent does not use his mind very carefully at the supermarket, we can no longer prove analytically that it will work (Arthur, 1995). It is no longer possible for the remaining agents to calculate each other's expectations, and we are back with the uncertainty of Rosser (2001).

Then what about insects living in colonies, ants and bees? They also manage to organize rather complex systems(does that imply that they are as rational as economic agents are? The Santa Fe Institute has developed the idea of Swarm intelligence to describe how complex adaptive systems self-organize (by means of an intelligence that is related to the system rather than to the individual agent (Bonabeau, Dorigo, & Theraulaz, 1999). They claim that there is no need to invoke individual complexity to explain complex collective behavior. Dumb bees may organize into a very smart beehive. From a complex adaptive systems perspective, one would consider a social insect colony or an economic system a decentralized problem-solving system, comprising many relatively simple interacting entities. It is interesting that such systems solve problems in a very flexible and robust way. Flexible in the sense that they have capacity to adapt to new environments, and robust in the sense that they are not distracted by poor performance of some agents.

Whether the economy is perceived as an equilibrium system or a complex adaptive system matters a great deal(especially if we purpose to use our knowledge of its functioning for making policy recommendations. The picture of throwing a rock or throwing a bird can be used. If the economy is best perceived as a static equilibrium system, policy intervention is like throwing a rock and we can foresee its trajectory. If the economy is more like a living system, it is like throwing a bird, and policy intervention becomes much more difficult(it is much harder to get an idea of where the bird is going to end up. Thus the complexity approach may help us understand the system on the one hand, but on the other hand it warns us about the predictability of consequences when we try to control the system. Believing that we are throwing a rock, when actually we are throwing a bird, can make the system worse off.

This is not an argument against any kind of policy intervention, but rather an argument against the dream of fine-tuning the system. Sometimes the system may behave so badly that we can be confident that "throwing a bird" is going to be helpful. This may be easier to understand if we stop thinking of economic

policy as the Gs of our comparative static models and start thinking of policy in terms of design. Can we promote or design institutions that may assist the economic system in self-organizing?

METHODOLOGY: A SYNTHETIC APPROACH

If the economy is best understood as a complex adaptive system, it implies not only a shift in perspective, but also a shift in methodology. Since the aggregate behavior of the system which arises from the interaction between a multitude of agents can be expected to display very complex dynamics, it becomes in vain, or at least very cumbersome, to study the system using deductive analysis. Rather than deductive analysis, ACE researchers synthesize—that is, they try to understand economic processes by synthetically creating them.[1]

The synthetic approach to model building is a bottom-up approach, where a model is built up by simple components that are assembled into a working system and simulated using computers (synthesis by simulation). For an economic model the components will typically be a number of agents, and the model is assembled by letting agents interact in a market. In economics we use the term agent-based computational economics for such a modeling approach. This is opposite to an analytical top-down approach where you start with a complete system, which is decomposed.

Dealing with artificial life, Bonabeau and Theraulaz (1994) set the difference between the two approaches at the edge: in a top-down approach you have to start with manifestation of life; in a bottom-up approach you synthesize more and more complex behaviors, hoping to capture the nature of some aspect of life. Transferring the analogy to economics, we do not want to start out defining market equilibrium and deducing the conditions under which equilibrium will exist. Instead we want to study the likely outcome of the interaction of agents.

In a synthetic bottom-up approach, you need not make any assumption about what the "up" is going to look like; the *up* is what emerges when units interact. The concept of emergence is very important to a synthetic methodology—one may claim that it constitutes the whole purpose of synthesizing. Aggregate behaviors or structures emerge out of self-organization rather than being imposed or assumed. Emergent properties cannot be explained by *built-in* microproperties—it is what makes the whole more than the sum of its parts. In systems with emergent properties, we cannot use simple aggregation; we will just end up committing fallacies of composition. Since emergence is a result of the agents' interaction, emergent phenomena cannot be expected to be analytically tractable. The only thing that we can do is to observe what happens. "We can learn about emergent properties only inductively" (Lane, 1993).

Thus the power of a synthetic approach is to find emergent properties internal to the system rather than relying on external explanations—that is, the approach may help economists explain business cycles from within the system rather than relying on exogenous chocks to the system. Its weakness is that the explanatory power of synthetic models is not very high. The space of exploration is huge, and constraints lack, and thus we easily end up with one thing after the other.[2]

As a young science, complexity science is in a process of defining the concept of emergence.[3] It is still not clear what demands should be met for a phenomenon to be characterized as an emergent phenomenon. We need not enter this discussion here, but we may clarify the concept of emergence by listing some ex-

amples of what emergent phenomena we may hope to find within economics.

- **Equilibrium:** Has already been used as an example of an emergent phenomenon; a tendency for prices, demand, and supply to meet in such a way that markets clear, without imposing the result in advance, may be an emergent feature of economic systems.
- **Self-Organization:** One might also be less ambitious and say that survival of agents in our economic model is an emergent feature (i.e., the fact that the system does not break down completely, but manages to self-organize to some extent)d.
- **Business Cycles:** The idea that economic activity has a tendency to move in cycles is very old, but economics still has problems coming up with a plausible explanation. The reason for this difficulty may be that the cycles are really an emergent phenomenon. This would not be a very surprising discovery since many other complex adaptive, or living, systems display cyclical behavior.
- **Institution Formation:** One way of dealing with uncertainty is to form institutions (Heiner, 1983), and the appearance of institutions may be perceived as emergent phenomena.
- **Skewed Distributions:** Distributions of income, wealth, changes in asset prices, and so forth may show characteristics that may be emergent phenomena. We shall return to this example later.

An important question is whether a synthetic approach can stand on its own. It is hard to find discussions of this within social sciences. To strengthen the approach, Dawson (2004), working within psychology, puts synthesis into a framework of synthesis, emergence, and analysis (SEA). Once a synthetic model has been built and is working, it needs to be run in order to observe emergent properties of the model. Parameter space needs to be explored in order to find out exactly under which circumstances an observed phenomenon arises.

The third step in Dawson's suggested approach is analysis; the difficult—and interesting—work starts when an attempt is made to generate theories of regularities that emerge from what we synthesize. In order to do this we need statistical testing of model output. Rosser (1999), however, complains that techniques for analytical testing of output data is missing and suggests that it will be necessary to rethink the nature of empirical testing. This must be done in such a way that we can take advantage of the fact that the output data comes from a simulated model and not from the real world. If interpreting simulated data is as difficult as interpreting real-world data, there really is no point in doing the simulation.

It should be evident by now that although the approach is agent-based, it would be wrong to term it methodological individualism or reductionism. The starting point is individualistic, but it is an important point that agents do not make their decision in isolation from their surroundings. Interaction causes institutions to emerge and these institutions affect human behavior. As an example one could use the emergence of a medium of exchange (Clower & Howitt, 2000). Economic theory seems to have given up the idea of microfounding monetary theory. It is simply too self-contradictory to argue why rational optimizing agents should demand money that is not needed in a general equilibrium setting. ACE models can demonstrate how a medium of exchange may emerge from interaction between agents. But an important factor in

this emergence is that the agents' preference for the goods chosen to act as medium of exchange is affected—that is, their preferences are not isolated from the preferences of other agents, and methodological individualism is violated.

Related to the discussion of methodological individualism is the discussion already touched upon—how much attention should ACE modelers pay to the behavior of agents. The description of human behavior must be consistent with the system; if the economic theoretician cannot deduce how the system works, then neither can the agent. This is the complete negation of the rational expectations hypothesis. Thus agents must use inductive reasoning rather than deductive. But inductive reasoning can also be quite complex, and we need to ask whether very complex modeling of human behavior is necessary for the emergence we search for to occur. This discussion is raised by Arthur et al. (1997):

Overrating the cognition is just another error deriving from methodological individualism, the very bedrock of standard economic theory. How individual agents decide what to do may not matter very much.

Economic systems may be much closer to anthills and beehives than we like to think, but that does not mean that we are not more intelligent than ants and bees. It just means that the overriding intelligence is not with the agent but with the system. What matters is not how we make decisions, but how we interact. If the point of interest is the system or macroeconomic view, there is a point for ACE researchers to find the simplest behavior that can generate the emerging phenomenon searched for, rather than behavior closest to actual human behavior.

BUILDING AN AGENT-BASED COMPUTATIONAL MODEL

Building an agent-based computational model gives you a sense of playing God in your own silicon world. You define a number of agents with characterizing variables, a set of decision rules, and an environment in which interaction takes place. Finally you let the agents act, and watch what happens. There are a lot of choices to be made during the modeling phase, and the purpose of this section is to discuss some of these choices.

The first choice to be made is what programming language to use. You may either start from scratch in any programming language, use a matrix-based mathematical environment (e.g., Matlab), or you may choose a framework prepared for agent-based simulations like Swarm or Repast. Generally, it is a good idea to choose object-oriented languages, since the approach of object-oriented software comes very close to agent-based modeling. Both approaches turn to decentralization to deal with complexity. For larger, more complex models, you easily run into constraints if a mathematical environment is chosen (remember that your model may grow as your research progress), but on the other hand there is a larger effort involved in choosing to program everything from scratch. Frameworks prepared to agent-based simulation have been developed in order to overcome this problem, but even in this case, the model builder must make a big investment. An important side effect of using a framework is access to models already built within the framework.

It is a good idea to choose programming language before settling the details of the model, since there may be help to find during the modeling process. Searching through existing models where source code is available is rec-

ommended, since you may be able to reuse much of the code. Searching through existing models also make choices about institutional features more deliberate: when do you choose the usual institutional settings (if a convention exists), and where does your model require a different setting?[4]

For modeling, especially systems with non-emergent macroproperties, a three-step approach to model building is recommendable. The three steps are macroanalysis, microanalysis, and simulated interaction. This three-step approach is very much in line with a synthetic approach as described by Auyang (2004):

Synthetic analysis encompasses two perspectives, looking at the system on its own level and looking at the level of its constituents. It includes two kinds of explanations. Macroexplanations develop scientific concepts and theories for composite systems without mentioning their constituents…However, for a full understanding of the system including their composition, macroexplanations are necessary but not sufficient. For this we also need microexplanations that connect the properties delineated in macroexplanations to the properties of the constituents. Microexplanation depends on macroexplanation, which first set out what needs microexplanation.

Thus, following the three-step approach, you first need to consider whether there are macro-bindings on your system—that is, detect any tautologies relevant to your system. Are you dealing with a closed economic system where, in the aggregate, expenditure must equal receipts? Does the system have money, and in what form is money to be implemented, for example as a *numeraire* in bookkeeping entries or as a fixed stock of currency. Detecting tautologies and non-emergent macrophenomena does not mean that you have to model them yet—detection is meant to assist you in the following steps.

Once you have an idea of the macroproperties of your system, you may consider how an individual agent acts. In any specification of economic behavior, simplification is necessary; the question here is what should guide simplification. We suggest starting out with macroanalysis rather than microanalysis in order to allow knowledge of macroproperties to guide simplification choices. Going the other way from micro to macro involves a higher degree of complexity and the risk of committing fallacies of composition. If the macroanalysis tells us that a given aspect of decision making is not all that important to the macroproperties of a system, we can be more careless in designing the given decision rule. But the decision rule may still be important from a macroperspective, since it may be necessary for closing the model.

The third and final step is simulation: building a model that combines micro and macro perspectives by letting agents interact in a closed world. To model a closed system, or at least a system where openness is controlled, is important in order to enforce the feedback from macro to micro.

Following the ideas of object-oriented programming, it is a good idea to think about agents (objects) as being characterized by a number of state variables (e.g., wealth, consumption) and having a number of decision rules or methods (e.g., how much and what to consume). As an example you may have a main program where you call an agent, who then acts according to decision rules and as a result of the action, state variables are changed. The action of an agent will often imply activating other agents' decision rules and changes in other agents' state variables. If Agent1 is called by the main program and asked to perform decision to con-

sume, in a completely decentralized model the agent needs to find some other agent to buy the consumption goods from. Agent1 then calls Agent2 and evokes decision rules by this agent in order to negotiate a trade. If a trade is negotiated, goods need to be transferred from Agent2 to Agent1 and money needs to be transferred from Agent1 to Agent2. Normally it is chosen to let control go back to Agent1 for more decision rules to be evoked, and finally back to the main program—that is, Agent2 is not allowed to react to the decrease in his stock of goods or the increase in money until later when the main program picks Agent2 for action.

Notice that in the above example, the two agents have to negotiate the trade locally without help from a centralized market maker. Within agent-based modeling, many different representations of what happens on a market exist. Instead of contacting another agent directly, one could have allowed Agent1 to put up whatever he wanted to sell as a list of offered goods at a certain reservation price. In buying goods he could search through the list to see if what he wanted to buy was offered for sale below his reservation price. Here we have a centralized institution: the list of goods put up for sale, but no centralized mechanism finding equilibrium prices. If, when he gets to act again, Agent1 realizes that nobody purchased the goods he put up for sale, he may try to lower the price demanded. Use of agents for Internet trading has extended research in auction theory, a development that ACE researchers may take advantage of.

As should be clear by now, the idea of agent-based simulation is very simple. However, when you start assembling your agents into an agent-based model, you may be overwhelmed by the number of seemingly unimportant institutional settings you have to consider. How exactly do agents interact, and in what order do they get to interact? Are the state variables of the agents updated immediately after they have acted (sequential or asynchronous updating), or do you wait until all agents have acted in order to update simultaneously (synchronous updating)? Huberman and Glance (1993) have illustrated how important this choice is within the field of evolutionary games. You also need to consider whether to model a fixed number of agents, or whether agents enter and exit, for example, through a bankruptcy mechanism. The number of agents you model may also turn out to be a crucial parameter in your simulation.

Whereas conventional economic theory has a hard time dealing with time and space, these important concepts come naturally to agent-based simulations. You may construct your model so that not only time matters in specifying the updating rule, but time and space also become important constraints in the behavior of agents. A cellular automaton type construction is often used to introduce space in the model, but other alternatives are also available. You need to consider whether all agents communicate with each other at a given point of time, or whether communication is constrained so that agents can only communicate locally. If space is modeled, do agents stay put in space or do they move around, for example in search of trading partners.

Within agent-based economics a number of different techniques for constructing agents can be found. Generally we have chosen to use the term decision rules, since most of the different techniques can be formulated as decision rules. ACE models often involve some form of learning, where decision rules change during the simulation. Learning processes can be represented using, for example, genetic algorithms or neural networks. In adopting such approaches in economics, it is however important to always keep in mind that we are dealing with human

agents. It may be straightforward to translate the fitness of genetic algorithms into some utility concept, but is this really the way you want to describe human behavior?

Once you have a running model, it is important to go back and check for the importance of some of the institutional choices you made. You may find that your results depend crucially on, for example, simultaneous updating, and this gives you an idea of the robustness and generality of your model. Also the number of agents may be important. If model results remain the same whether you operate with a population of 100 or 1,000, you may as well stick with 100 agents—but it is important to check for results with 1,000 agents.

Finally you check the importance of behavioral parameters. Parameter space is typically too large to be explored completely, so it is necessary to use experimental methods (e.g., Monte Carlo) for finding the most important parameters. Now the fun is over, and you need to go into the analysis part of Dawson's framework. Here it is important, once again, to look at other simulation models and compare results to search for possible generalizations.

CONSTRAINING THE APPROACH

Going from the traditional analytical methodologies to the synthetic agent-based approach presents economists with a lot of freedom we have never had before—too much freedom some may add, for what is to define economics as a science, once we allow ourselves to go beyond the study of rational optimizing behavior? Does anything go, and if so, how can we progress as a science?

If one contributor uses ACE to demonstrate the likely occurrence of market equilibrium in an unregulated market while another contributor uses it to get the opposite result(that state intervention is necessary for markets to behave reasonably(who is more right, and how is economics as a science to deal with these results?

As a new approach to economics, ACE researchers have to deal with such problems. We should enjoy the newly won freedom, but we should also use it deliberately. Agent-based economics is not a new approach to economics; it is a new methodological approach to the study of economic systems. This implies that we do not need to start from scratch when we apply the method, but want to apply the method where it is needed(that is, where economic theory or economic controversies are stuck. By anchoring ACE in existing economic theory, we stand a better chance of progressing as a science.

For economics as a science, at least three different justifications for agent-based computational models can be presented:

1. **Models that attempt to reproduce theoretical results that are analytically tractable:** The purpose here is mainly to check the realism of the approach while constraints remain the same.
2. **Models that reproduce theory that is not analytically tractable:** Here the purpose is to check the consistency of theories that have not been formalized, and to extend such theories to involve both microeconomic and macroeconomic aspects.
3. **Models that reproduce empirical findings:** Here the purpose is to understand what generates empirical findings.

Reproducing Results of Analytically Tractable Theory

Why use ACE models to reproduce theoretical models that are analytically tractable? When Adam Smith introduced his invisible hand, it was not the result of a desire to have some divine mechanism in his economic theory—his

purpose was not to prove the existence of God, but to illustrate the self-organizing capacity of a market economy. When economic theory became mathematical, an important goal was to replace the invisible hand, first with a proof of existence of equilibrium, and later Herbert E. Scarf hoped to find a numerical solving mechanism that could actually find the equilibrium. Scarf's (1973) work led the way for the widespread use of computable general equilibrium models we witness today. But have we really come that far from Adam Smith's invisible hand? Haven't we just replaced one *deus ex machina,* the invisible hand, with another, the solving mechanism?

ACE models allow us to replace the *deus ex machina* with something that is a lot closer to what actually happens at markets. Rather than assuring self-organization of market economies by imposing an auctioneer, self-organization is a result of interaction. Such a hypothetical result of emergent equilibria could be used to dismiss the claim for necessity of simulation once the realism of the analytical approach has been stated. Another approach is, as noted by Tesfatsion (2002), that once we have the result of emerging equilibria, we can start relaxing assumptions, thus replicating the research program of general equilibrium theory. Once we move beyond the scope of analytically tractable models, however, we lack the constraining factor. One model relaxes assumptions on competitive markets, another introduces transaction cost, a third model introduces bounded rationality, and so forth—how do we generalize the information we get from such a multitude of models?

Reproducing Theories that are not Analytically Tractable

Using ACE to formalize theories that it has not been possible to analyze using formalistic mathematics has obvious advantages. Much macroeconomic theory was dismissed in the 1970s as a result of requirement of microfoundation (and formalization)—but if they were attempts to theorize about emergent phenomena, we cannot expect to be able to deduce such theoretical propositions analytically. Within an ACE approach we can now check macroeconomic theories within a constrained environment given a well-defined microbehavior—that is, we can check their consistency. Agent-based models have a great potential in this area; they can unify micro- and macroeconomics in a way that has not been possible before. Within this approach ACE models should be perceived as tools for theory development.

ACE models also have a potential as tools for economic policy as discussed by Rosser (1999). Governments may be perceived as assistant to the self-organizing process, for example by providing the institutional structure within which self-organization takes place, or as a necessary institution for rectifying possible self-destructive aspects of complex adaptive systems (e.g., a tendency to create a very skewed distribution of wealth).

Reproducing Empirical Findings

Empirical data are often emphasized as the constraining factor in artificial life and agent-based approaches (Bonabeau & Theraulaz, 1994). For many aspects of economics, it is however difficult to obtain the relevant data—especially considering the fact that human behavior is involved. How and why do we make the decisions we make? In one sense empirical data can, however, be considered the constraining factor of agent-based computational economics, namely when models are developed in order to explain the emergence of aggregate empirical findings—that is, explaining economic stylized facts.

ACE models in particular have proved to be useful for explaining the existence of power law distributions in cases where normal (bell curve) distributions were to be expected (Simon, 1955). Pareto observed a tendency towards a skewed distribution of wealth, where 20% of the population holds 80% of the wealth. Agent-based models have produced similar skewed distributions (Epstein & Axtell, 1996; Bruun & Luna, 2000). ACE models have also been able to produce the "fat tails" of the stock market (Lux & Sornette, 2002), the size distribution of cities (Krugman, 1996), and the size distribution of firms (Axtell, 1999). An important task for ACE is to generalize results on power law distributions, as well as using the method to explain other stylized facts.

CONCLUSION

Agent-based computational economics has been presented as a new methodological approach to the study of economic systems. Its strength is its ability to combine micro- and macroeconomics, and thus overcome one of the biggest hurdles of economics. It allows economists to pose new questions and repose old questions in a less restrictive setting.

Conceiving the economy as a complex adaptive system implies focusing on the self-organizing features of economic systems where a concept as equilibrium becomes an emergent phenomenon that does not rely on exogenous mechanisms or *deus ex machina*. This also means asking the questions "Why does the system function so well?" and "Can we possibly assist it in performing even better?", rather than asking why the system does not function perfectly and which obstacles should be removed.

The main theoretical drawback of the approach is that it becomes very hard to define and delimitate economics as a science. In decentralized agent-based models, assuming rational optimizing agents no longer suffices as the defining feature of economic theory. Thus it is no longer clear what should constraint economics as a science. This may not be a problem; other social sciences survive without such defining constraints, and economics is after all just another social science!

A more practical drawback of the approach is that it cannot provide us with the same sort of policy recommendation that macroeconometric models have provided. Policy recommendation based on ACE models have a qualitative rather than a quantitative nature. Focusing on the self-organizing features of economic systems does not necessarily imply a laissez-faire approach to economic policy—an important question to ACE policy modelers is what institutional setting is best in supporting self-organizing features of the system. Thus the design or policy aspect of ACE will often be about building up institutions, whereas recommendations based on general equilibrium theory will often be about the removal of institutions.

ACE has been presented, not as a new approach to economics, but as a new methodological approach to the study of economic systems. Ace is not going to resolve economic controversies, but is hopefully going to qualify discussions and help us locate the roots of controversies. Newcomers to the field should be aware of this—and keep focus on their purpose in doing ACE. Only by anchoring the approach in existing economic theory, and existing economic controversies, can we confront the risk of simulation fetishism with one model entering the scene after the other.

REFERENCES

Arthur, B. W. (1995). Complexity in economic and financial markets. *Complexity, 1*(1), 20-25.

Arthur, B. W., Durlauf, S. N., & Lane, D. (1997). Introduction. The economy as an evolving complex system II. In B. W. Arthur, S. N. Durlauf, & D. Lane (Eds.), *SFI studies in the sciences of complexity* (vol. XXVII). Reading, MA: Addison-Wesley.

Auyang, S. Y. (2004). *Synthetic analysis of complex systems 1—Theories*. Department of History and Philosophy of Science, University of Sydney, Australia. Retrieved from http://www.usyd.edu.au/su/hps/newevents/Auyang1.html

Axtell, R. (1999). *The emergence of firms in a population of agents—Local increasing returns, unstable Nash equilibria, and power law size distributions.* Brookings Institute.

Bonabeau, E., & Theraulaz, G. (1994). Why do we need artificial life? *Artificial Life, 1*(3), 303-325.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). Swarm intelligence—From natural to artificial systems. *Santa Fe Institute Studies in the Science of Complexity.* Oxford, UK: Oxford University Press.

Bruun, C. (2002). Prospect for an economics framework for swarm. In F. Luna & A. Perrone (Eds.), *Agent-based methods in economics and finance—Simulations in swarm.* Boston: Kluwer.

Bruun, C., & Luna, F. (2000). Endogenous growth in a swarm economy. In F. Luna & B. Stefansson (Eds.), *Economic simulations in swarm: Agent-based modeling and object oriented programming.* Boston: Kluwer.

Clower, R., & Howitt, P. (2000). The emergence of economic organization. *Journal of Economic Behavior and Organization, 41,* 55-84.

Dawson, M. R. W. (2004). *Minds and machines: Connectionism and psychological modeling.* Blackwell.

Epstein, J. M. (2005). *Remarks on the foundations of agent-based generative social science.* CSED Working Paper No. 41, Brookings Institute.

Epstein, J. M., & Axtell, R. (1996). *Growing artificial societies—Social science from the bottom up.* Cambridge, MA: MIT Press.

Heiner, R. A. (1983). The origin of predictable behavior. *American Economic Review, 73,* 560-595

Holland, J. H. (1998). *Emergence—From chaos to order.* Oxford, UK: Oxford University Press.

Huberman, B. A., & Glance, N. S. (1993). Evolutionary games and computer simulations. *Proceedings of the National Academy of Sciences, 90,* 7716-7718.

Keynes, J. M. (1936). The general theory of employment, interest and money. *The collected writings of John Mayneard Keynes* (vol. VII). Macmillan. (Cambridge University Press, 1973).

Kirman, A. (1992). Whom or what does the representative agent represent? *Journal of Economic Perspectives, 6*(2), 117-136.

Krugman, P. R. (1996). *The self-organizing economy.* Blackwell.

Lane, D.A. (1993). Artificial worlds and economics, part I and II. *Journal of Evolutionary Economics, 3,* 89-107, 177-197.

Lux, T., & Sornette, D. (2002). On rational bubbles and fat tails. *Journal of Money, Credit and Banking, 34,* 589-610.

Putnam, H. (1988). Much ado about not very much. *Daedalus, 117*(1), 269-281.

Rosser, J. B. Jr. (1999). On the complexities of complex economic dynamics. *Journal of Economic Perspectives, 13*(4), 169-192.

Rosser, J. B. Jr. (2001). Alternative Keynesian and post Keynesian perspectives on uncertainty and expectations. *Journal of Post Keynesian Economics, 23*(4), 545-566.

Scarf H. E. with the collaboration of Hansen, T. (1973). *The computation of economic equilibria.* New Haven, CT: Yale University Press.

Schelling, T. C. (1978). *Micromotives and macrobehavior.* W. W. Norton & Company.

Simon, H. A. (1955). On a class of skew distribution functions. *Biometrika, 4*(2/3), 425-440.

Tesfatsion, L. (2002). Agent-based computational economics: Growing economies from the bottom up. *Artificial Life, 8*(1), 55-82.

KEY TERMS

Agent-Based Computational Economics: Acknowledging the economy as a complex adaptive system, agent-based computational economics (ACE) is the computational study of economies as collections of interacting agents. By letting autonomous agents interact *in silico,* the overall aim is to represent self-organizing features of real-world economic systems in a more realistic way than has hitherto been possible. But ACE may be applied to many different aspects of economics and should be perceived as a new methodological approach to the study of economic systems rather than a new approach to economics. Focus may be placed on the individual agent and its interaction with its surroundings (e.g., learning by the agent), or on the system resulting from the interaction of many agents (e.g., learning by the system).

General Equilibrium Theory: As agent-based computational economics (ACE), general equilibrium theory is a bottom-up approach to the study of economic systems. But in opposition to ACE, general equilibrium theory makes use of methodological individualism and reductionism. Taking as its starting point endowments, preferences, and technologies of economic agents, general equilibrium theory uses an auctioneer to calculate a vector of prices that will clear all markets. Methodological individualism allows general equilibrium theory to regard preferences as given, and the existence of an auctioneer prevents trading at *false* prices, since agents are only allowed to interact through the auctioneer. A number of fundamental problems arise when the ideal general equilibrium model meets the real world. Among these is the problem of making room for money in a setting where there is no need for a medium of exchange and where any good can take the role as unit of account.

Keynesian Macroeconomics: Keynesian macroeconomics takes as its starting point aggregate magnitudes and their interrelations. Of pivotal importance to the ideas of Keynes is his definition of national (money) income as the earnings of the factors of production, which must be equal to the cost of production. From this follows the equivalence of saving and investment; both are defined as the excess of income over consumption. This definition implies that consumers will always hold enough money to buy the product. However, this does *not* imply that supply creates its own demand in the sense that lack of aggregate demand cannot cause unemployment (Say's law). Entrepreneurs decide the level of production, and they will not produce beyond their own investment demand plus expected consumption demand. If, at the level of full employment, consumers in the aggregate are expected to save more than entrepreneurs wish to invest, entrepreneurs

will reduce production until expected savings equals expected investment. By, in this way, arguing in terms of aggregate magnitudes, Keynes depicts a different economy than the one known from neoclassical or general equilibrium theory—an economy where unemployment due to a lack of aggregate demand is not only possible but also likely.

Macrofoundation: If for example methodological individualism is violated in a composite system, simple aggregation is no longer possible, and studying macroproperties directly may be the only way to learn about certain aspects of the system. This may be characterized as the macrofoundation of the system.

Methodological Individualism: Methodological individualism's purpose is to explain and understand macro phenomena as the aggregation of decisions by individuals—the whole is nothing but the sum of its parts. This may also be described as reductionism since methodological individualism allows the whole to be reduced to the sum of its parts, and thus the parts that make up the system may be studied in isolation. Methodological individualism is an essential part of general equilibrium theory and in the demand for microfoundation in economics.

Microoundation: Making use of methodological individualism, the claim for microfoundation states that any macroeconomic phenomena must be shown to result from individual (rational) actions.

Rational Expectations Hypothesis: Rational expectations hypothesis is the idea that economic agents use all available information, including information on economic relations, in forming expectations about the future. Rational expectations may also be denoted "model consistent expectations" since consistency between the economic model and the model used by agents in expectation formation is required. Economic agents are just as well informed as the economist (or even econometrician) building the model.

ENDNOTES

1 Tesfatsion (2002) notes that economists prefer to think of their ACE models as representations rather than syntheses, whereas Epstein (2005) talks about a generative science—But the methodology remains the same.

2 This phrase originates from Putnam (1988) discussing artificial intelligence.

3 On the other hand, Holland (1998) wrote a book entitle *Emergence* without trying to define it.

4 How ACE can avoid "the tower of Babel" effect by stealing code from each other is discussed in Bruun (2002).

Chapter XV
Data Gathering to Build and Validate Small-Scale Social Models for Simulation

Juliette Rouchier
GREQAM, CNRS, France

ABSTRACT

This chapter discusses two different approaches that gather empirical data and link them to modeling and simulations with agent-based systems: experimental economics which built reproducible settings and quantitatively defined indicators, and companion modeling which accompanies observed social groups when they negotiate over renewable resource issues. Both developed techniques have different epistemological posture which lead them to promote diverse data comparison and model validation. They both have small limitation. The chapter wishes to put forward that, although both approaches have different goals, some evolutions in research protocol could enhance qualities of both. Some of these evolutions have already started to be used by researchers.

MODELING AND THE SEARCH FOR REALISM

Social simulation using multi-agent paradigm (sometimes called *agent-based simulation* or *simulation with agents*) has developed quickly in the last fifteen years. The tool offers numerous new possibilities to represent rationality and structures of interaction, to take into account for heterogeneity in rationality and perception (Kirman, 1997; Bousquet, Cambier, Mullon, Morand, Quensière, & Pavè, 1993) or social relations (Moss & Edmonds, 2005; Rouchier, 2004), and test a variety of learning models for agents with procedural rationality (Simon, 1969).

The discipline of social simulation with agents started from two distinct communities at least. One was the economics community, with regular seminars at the Santa Fe Institute called "Economics as an Evolving Complex System" (Anderson, Arrow, & Pines, 1988). Another branch was started in Europe with the book *Artificial Societies* (Gilbert & Conte, 1994).

The first developments were quite close to artificial life (Langton, 1991). Researchers were trying to build credible global behavior from local actions, and were qualitatively inspired by social facts. For example, some would deal with emergence of hierarchies based on hypothesis by archaeologists (Doran et al., 1995), and would study which parameters have an influence on the existence and stability of the emerging phenomena. They were producing global behaviors that resemble the type of general structures that can be observed in society, and it was an important first step to be able to "grow artificial society" (Epstein & Axtell, 1992). Hence, most simulations of complexity were used as "black boxes" (Simon, 1969): the influence of the changes of parameters would be studied in relation with some global observation parameters, without following necessarily step-by-step processes to understand the internal mechanisms. There was no strong request for validation at that time, and all representations put in a model as well as evaluation processes relied on the expertise of the researcher. The issue of realistically understanding the influence of the numerous parameters of these complex systems, however, became more and more central (Gilbert & Abbott, 2005; Edmonds et al., 2003), and implied the apparition of new methods for building and validating models.

Nowadays, it has become a norm to assess results with actually comparable data and to build the hypotheses themselves by offering some empirical facts to justify for the model construction. Diverse sets of data can be used—from surveys to observations in real settings, and a large number of applications have searched for the right use of empirical data in the building of models (Moss & Edmonds, 2005). In this chapter, we focus on two trends that explore the representation of small-scale interaction settings with multi-agent simulations, caring about both protocols to gather data and assess the results of their assumptions on rationality, and involve human subjects in the process.

Some researchers want to perform quantitative validation for their simulations, and try to use their models to establish positive scientific results. The method of statistical comparison with outside world data is very often used (Moss & Edmonds, 2005). In the context of experimental economics, it is possible to acquire a lot of very precise behavioral data on microbehavior in the context of well-defined choice. A systematic comparison of artificial and real agents' behavior is, for some, a good way to assess the validity of the cognitive models they build for agents. A Popperian approach can be identified in these protocols, where scientists position themselves as outside observers of a social system on which they can draw objective refutable theories.

Others, at the opposite end of the validation approach, question the need to switch from qualitative validation to quantitative validation and look for scientific methods where the validation would not be abstracted from the social use they assign to social sciences. The second approach presented here is called companion modeling (some recent papers also refer to the stakeholder approach) and is rather embedded in an instrumentalist epistemology (Zammito, 2004), referring to the Duhem-Quine thesis. Followers are conscious of the fact that their hypotheses about rationality and structure of systems are social construct (Berger & Luckman, 1969), and instead of confining the use of the model to an analytic situation, they use it as a device for communication among those very actors who have been mimicked in the system.

Both approaches have different goals and invented settings that are suited to their general understanding of knowledge of society. We present here those two protocols, which define a rather high methodological standard for validation of simulation models. They have two

aspects in common: (1) they can be applied mainly to small-scale studies, and (2) they can be related to *empirical research* since they include at the same time the gathering of data of *real behaviors*, the building of the model with observed data, and the test of their assumption through a comeback on the gathered data. The differences and limits of both approaches will be presented, and some opening remarks will be made.

SIMULATIONS AND ECONOMIC EXPERIMENTS

Abstractly Studying Decentralization

There is a contemporary general issue in economics: to construct a theory of behavior for economic actions (Kirman, 1997). Researchers belonging to the trend called ACE (agent-based computational economics) have used multi-agent simulations (or *agent-based simulations*) either to test models that they had inferred from the observation of economic activities or, using genetic algorithm, to discover some optimal algorithms. The idea is to describe the rationality of agents in a really decentralized setting, with imperfect information, and look at the evolution of the global data so that to judge the relevance of the model (Tesfatsion, 2006). Up until recently, those focusing on the behavior of individuals and history of transactions rather than optimality were quite rare, and their hypotheses were based on long-term experience in micro-economics (Kirman & Vriend, 2001).

Other economists focus on a more precise description of rationality: for now more than 40 years, experimentalists have been testing the possibility of implementing archetypical economic settings in laboratories (Smith, 2002). They make human subjects play in a game, where choices faced are similar to those described by economic theory. Those subjects' preferences are voluntarily controlled by monetary incentives (and their rational action is computable by the one who organizes the experiment); the amount of information they get is limited and carefully defined. The issue is to judge if subjects do behave in a perfectly rational way, or if they show some behavioral features that are not predicted by theory.

The experiment has one advantage, which is to be perfectly *controlled* and *reproducible.* All information circulation and actions are strictly limited and recorded with the help of computers. The situation is hence different from real life, where many situations are different from ideally expected equilibrium, but it is impossible to decide if the lack of information or the lack of rationality is responsible. The experimenter does not have access to the calculus led by the subject, but can make subtle changes in distributed information, to understand better the way it is used.

The results have been gathered for some time, and a huge amount of relevant descriptions on how humans treat information in economic contexts has been gathered (Camerer, 2003), concerning for example: cognitive limitations, need for reciprocity and altruist behaviors, consumer attitude, public good games, and non-cooperative games. Most of the time, results are compared to theoretical expectations, and when real behavior differs from optimal model, interpretation has to be given.

To explain the deviations from theoretical behaviors, positive models are proposed as dynamical learning in context based on equations (Fudenberg & Levine, 1999). These models are now tested thanks to simulations. Designing these simulations is pretty easy because the settings for experiments were already ready to be transposed to produce a distributed model. All proposed equations and hypotheses must be turned into algorithms, and when the simulation

has run, quantitative data can serve as benchmarks, and using the same indicators, help in selecting models (Brenner & Vriend, 2005). The main part of produced research is dealing with learning in markets (Duffy, 2001; Rouchier & Robin, 2006; Brewer, Huang, Nelson, & Plott, 2002), but some were concerned with other topics, like cooperation (Janssen & Ahn, 2003). For an excellent review on the topic, one can refer to John Duffy's chapter in the forthcoming handbook for computational economics (Duffy, 2006).

Examples and Critics

An application of the abstract model referred to as EWA (*e*xperience *w*eighted *a*ttraction learning model) by Ho and Camerer can be found in Chong, Camerer, and Ho (2005). The paper is a study of the calibration of the model with some data collected through experiments. The model is considered as very good and pure, since it only depends on one variable to adjust. The authors show that it is rather good for estimating learning by individuals, and that they can characterize different type of players (sophisticated—who realize that others are learning—or not) by looking at their results. However, one can note that in the paper, the authors only consider rather simple games with two agents. Plus, the fit of simulated and experimental data is not so good. Some people tried to use their model in simulations using a higher number of agents and found it rather impossible to choose the adequate parameters to fit their own data. This corresponds pretty well to the analysis by Thomas Brenner, who tries to make a complete review of the relevant use of learning models depending on the type of model that is built and shows that for the moment, each has very limited application (Brenner, 2006).

Going further, Janssen and Ahn (2003) showed that in the context of a public good provision game, it is not possible to adapt existing individual learning models to both individual and global data. Using a large set of experimental results from diverse researchers with varying payoffs in the same game, they ran simulations using two types of models: best-response model with signaling and EWA. Both models could match human results much better (statistically) than a perfect rationality calculus. EWA was proven to be good to match individual trajectories, but was very poor in finding global results; best-response was good for the global results but could not describe individual trajectories well.

A recent paper (Lelec, 2005) presents reimplementation of a set of simulations based on experiments on ultimatum game, with a reinforcement learning model proposed by Roth and Erev (1995). The paper shows that reinforcement learning is extremely sensitive to parameters settings. A very annoying result is that different parameters settings can lead to simulation results that are difficult to discriminate. Hence which model is right is difficult to choose when comparing with human subject experiments. The indicators that are usual for experimenters do not always allow discriminating, even for rather simple systems. This can also be found in the case of the reproduction of a market with artificial agents, where the speed of convergence is similar for a lot of different simulations and does not allow the decision of which model is best (Rouchier & Robin, 2006).

Duffy (2001), in a complete research, reveals all these issues and looks for experiments to go further in understanding human rationality in context. His example was an attempt to lead human subjects, and then artificial agents, to speculate. In his first experiments, he recorded behaviors, asked his subjects about their logic, and produced hypotheses on learning. From these assumptions he built an algorithm and led simulations. He insists on saying that, to compare, he used the same number of agents and the same time-length as in real experiments (which is rarely done). Since he could not

match data for individual agents' trajectories and global results, he mixed agents and subjects. Then all results, including human behaviors, changed a lot. In no case could he get his artificial agents to act like humans globally; only individual behaviors would qualitatively match human ones.

The Need for New Protocols

The comparison of experiment results and simulation data seems quite easy to realize at first sight, due to the perfect reproducibility of experiments and the possibility to produce simulation settings where the representation of information circulation is very close to the one of experiments.

The idea of building a general algorithm that mimics human learning is apparently not a reasonable short-term aim in the contemporary context and might never be. When it comes to understanding better the structures that emerge through iterative learning, it seems clear that several issues are at stake. There is a need to rely on a strict methodology to build indicators and link results to each initial parameter of the simulation. To know the context of learning should enable a decision about which learning model is better in different cases (Brenner, 2006). One can note that for the moment, representation of intelligence for humans is indeed very little inspired by some of the great theories of mind that have inspired the apparition of Artificial Intelligence as a field. Maybe the fact that all reasoning is represented through linear equation-based models is a problem. The composition of mind as an emergence of different interacting sub-computers is a hypothesis that maybe should not be ignored when representing learning, even in economics (Minsky, 1985; Bateson, 1972).

When writing about the importance of experiments, Smith (2002) stresses the fact that no experiment can actually destroy or prove a theory, just ask new questions and identify application boundaries for theories. This is a rather usual assertion nowadays, and corresponds roughly to what is referred to as the Duhem-Quine thesis: experiments embed so many hypotheses that one result cannot be taken into account to refute or assess a specific aspect. Simulations can be seen as having a similar status, and are just a way to test the coherence of hypothesis (or cognitive consistence) and show their limits, from which new assumptions should be expressed. The question of "truth" of a complex model seems so impossible to evaluate that many researchers now consider that it should not be used on its own as an abstract object that can more or less prove anything, but only considered in its context of use (Barreteau, Bousquet, Millier, & Weber, 2004). This is similar to the efficiency paradigm of the instrumentalist approach, also said to be emphasized by Quine (Zammito, 2004). A rather different protocol for research with simulations and human subjects—usually referred to as *companion modeling*—has evolved in that sense; it enables accumulation of knowledge on human behavior through the confrontation of simulated and real behaviors.

COMPANION MODELING

Resource Management and Complexity

Simulations have been widely developed for the representation of social-ecological systems for management application, with social entities interacting with the resource and with each other through the resource. Resource management issues are usually grounded in reality and imply superposing several points of view on a system. One can list for example three different issues that are relevant in these studies (Janssen & Ostrom, 2006): how to solve social

dilemma and achieve coordination with sometimes competing interests (close to economics), the analysis of how humans deal with uncertainty (rather psychological issue), and the influence of social networks on human activity (sociology). Each of these elements is still a question under study per se, where knowledge about how to represent humans' actions is very limited. Hence, to be analyzed and represented, resource management issues need the insight from different sciences when it is dealt with in context. And it is rather clear that depending on the main point of view taken on the system—sociology, geography, or economics—different conclusions will be drawn. It is even possible to prove the influence of points of view with a multi-agent system containing different models with agents acting differently according to the point of view that is chosen (and conclusions on emergent phenomena hence differ) (Bousquet, 1994).

Another issue when dealing with resource management is that the problems are not abstractly arising. They are usually situated within a political context, where the researcher himself has to be conscious that, as a knowledge builder, he is an actor in a global social process (Berger & Luckman, 1966). In this sense there is no possibility to really believe in the position of scientist as an outside expert who would define the problem on its own and bring optimal solutions to the population. Legitimacy of any representation or solution has to be recognized by the group under study (Barreteau, Le Page, & D'Aquino, 2003).

It would be all the more difficult to pretend to be an expert than to actually predict with accuracy the future of a relation between man and nature. We saw in the preceding section that the evaluation of simple behavior models is almost impossible with econometrics, even when one compares with experimental data (Janssen & Ostrom, 2006). With data gathered in vivo, and considering the permanent technical and institutional innovation that takes place, the importance of each factor gets so confused that it seems almost dangerous to pretend to evaluate quantitatively the future. Contemporary decision making in resource management is certainly due to relying on adaptive policymaking rather than top-down imposed plans (Holling, 1978), and there is a need for methodology sustaining this research. The *companion modeling* approach, using simulation as a qualitative disputable representation of reality, has been developed in such a post-normal direction (Funtowicz, Martinez-Alier, Mundo, & Ravetz, 1999), where the scientist is just one of other actors in the research for solutions.

Methodology and Applications

Companion modeling is one among other approaches involving stakeholders in building and evaluating models, and less heavy approaches can be applied (Taylor, 2004). Its characteristic is the great care given to make sure the issue predefined by researchers can be appropriated as a hypothesis by users and often discarded in the definition of new research objects. It has been for the moment dedicated to policymaking for organizing sustainable resource management. Sustainability implies that individuals that are going to use the new institution or infrastructure will be able (and willing) to do so. To describe environment and society as accurately as possible, diverse points of view have to be taken into account in the model when considering the complex and often conflict-based situations—from scientific experts to local stakeholders (Barreteau & Bousquet, 2000).

Coming as an expert in the field and getting individuals to express their points of view is not an easy task, and the usual methodology used by those referring to companion modeling is: build a first analysis to roughly identify the relevant stakeholders for management or con-

flict resolution, involve them in discussions, and present them with a first-draft model. To present a model, the use of a role-playing game is the main tool. It represents an easy way to discuss rules and interactions with a group, but also to make sure that all participants of the meeting understand the core of the model, to be able to use it for a session. After the game it is usual to gather suggestions about new scenarios and run subsequent simulations, giving rise to new discussions about the model and requests for change. This represents the beginning of the protocol, and usually researchers have acquired, at this stage, a lot of crucial information on the real representation of local actors, and of interactions and decision-making dynamics that exist on the field (Etienne, Le Page, & Cohen, 2003).

To make sure that stakeholders are not manipulated by researchers and do not reveal just what was present as an understatement in the model and game, time is a good ally. The revision of the model is presented a few months later, in new sessions where stakeholders themselves propose new people to involve in the discussion, and focus the discussion more clearly on the aspects they feel more relevant in the conflict. The repetition of sessions (usually three at least) of play, scenario building, and comments, helps a stuck or unclear situation to evolve. A better understanding of reasons of institutional blockage arises, and solutions emerge through discussion.

It has been shown that the model does increase the ability of individuals to communicate and find viable solutions. First, when discussing the model, most social hierarchies can stay out, and people can discuss the model a bit more openly (with limits) (Daré & Barreteau, 2003). Second, in any type of negotiation, there is a need for an intermediate object to position discussions (especially when the issue is not clearly delineated as in these initial situations), and the model is very efficient to serve this role (Bousquet et al., 2002). Eventually, the fact that the model has been co-constructed implies a higher level of legitimacy than if it had been imposed; during the long intervals, stakeholders understand the modeling process and are able to formulate their own hypothesis and understanding of the situation, and to use the meetings as a confrontation period. The process in itself gives roots to the enforcement of the decided policy since relevant individuals get involved. This is the main validation point of the model that is built: it is perfectly legitimate for users, and useful as a tool for action on the considered group.

For the moment this process has been shown successful for numerous applications. In Senegal it helped to secure irrigation systems in several villages (Barreteau & Bousquet, 2000) and solve conflict threatening herders (D'Aquino, Le Page, Bousquet, & Bah, 2003). In Holland, organization of traffic could be better managed (Duijn, Immers, Waaldijk, & Stoelhorst, 2003). In Vietnam, it sustained decisions taken for land-use changes (Castella, Trung, & Boissau, 2005). In Thailand, it helped create local decision-processes for land-use and revealed issues of access to credit for farmers (Barnaud et al., submitted). Discussions on sylvopastoral management in France have been also greatly sustained through the use of the method (Etienne, 2003).

In this last example, as well as the irrigation example of Senegal, the games that had been created through negotiations have been transformed so to match educational purposes and help explore group behaviors in various contexts. Observing diverse audiences while they are playing can indeed offer much information about the processes that lead to the apparition of new institutions (D'Aquino et al., 2003).

While the process got generalized, new issues arose since various relations to RP games and simulation (each having a distinct role in the process of elicitation and resolution) could be

observed. For example, associated sociological research was led to understand individual learning processes in Thailand, and enabled anticipation of the future impact of the decision made in the context of the game (Patamadit & Bousquet, 2005).

Limits

Companion modeling produces models that are validated in the sense that they are accepted and efficient for leading local policy changes; it is an excellent tool to reveal the precise shape of interest conflict; it also reveals new questions and hence helps furthering social theoretic research. However there are still some limitations in the method in its existing form. The first and most obvious one is the lack of accumulation of a knowledge that could be generalized to more than one situation. The following remarks are examples that might be irrelevant to the practitioners, but that are striking when reading this literature.

Companion modeling has been applied to a lot of cases where it has revealed hidden reasons for conflicts. However, there has been little accumulation of knowledge on a relation between the reality of conflict and its main expressions. One can imagine that there are various forms of conflicts with the same base, and maybe certain types of draft models could be more adapted than others facing given situations. In every case, this time-consuming activity treats one situation, but even if it can be compared to others in terms of resolution, authors never tell. From the list that has been drawn, a more general view on how to help the appropriation of the process by actors could be valuable. The transmission of this knowledge has been done through formation of researchers, but could it not be shared via usual standards of transmission?

To attain this aim, one necessity seems to be the reproduction of conflict situation in the laboratory, as was the case for two RPGs: a subsequent model was adapted to match various audiences and tested. A study of produced patterns and the possibility to produce simulation models could be interesting. Reproducibility of the experience is the strength of experimental economics, who established standards to describe settings and results. This is also what made it so easy to turn into simulations. Maybe it could be a good way to judge the findings of companion modeling to offer readers an archetypical game they could challenge in another context. Since research is now being led to join both protocols, maybe proposals will arise in that direction.

One last remark concerns the lack of use of psychological models in simulations (Janssen & Ostrom, 2006). The assumption is that points of view have to be captured and revealed, since they have an impact on the evolution of the system. If they were written in the simulation models following psychological representation of exploring and learning from experience, it could maybe be possible to delineate psychological effects from those linked to the conflict setting itself.

CONCLUSION

To build and evaluate agent-based models, one has to make hypotheses and assess them by comparing with data at two levels: the internal rationality of individual agents, and the emerging patterns that can be observed (for individual behavior and global trends in the group). In this chapter two approaches were presented that use almost similar settings, controlling at the same time simulated and human behaviors in the elaboration of the model: simulations linked with economic experiments, and companion modeling.

In the first one, human subjects have to accept a predefined problem and choose with

their rationality, facing a situation they have no way to question. What is observed is a highly constrained behavior, and the tested models can hence give presumably very general models on human rationality. For the moment, although some experimentalists have tried to actually create a questionnaire understand subjects' internal processes (Nyarko & Schotter, 2002), simulating researchers are not using these techniques to elaborate their assumptions. There is a danger in staying so far from an evaluation by the concerned actors: simulation is for the moment very unreliable to demonstrate the quality of an assumption of its uniqueness and cannot by itself be used in such a Popperian approach.

The second approach wants to spot rationality by allowing the actors themselves to build the model and accept it when all assumptions and global behaviors have been analyzed by the actors. Validation comes from the fact that the model is useful for the resolution of social dilemma in the context of resource management, as an acceptable and legitimate representation. The understanding of processes is more at the center of research than in the case of economics, and the model is not considered as an external truth. The problem in this very self-conscious instrumentalist approach is that it lacks accumulation of knowledge: by giving account of actors' interpretation of simulations only in their context of production, like companion modeling has mostly done, it was certainly possible to miss numerous patterns that could be more generalized.

One can suppose that intermediate approaches could be chosen: a companion modeling where only a (possibly large) selection of abstract games would be proposed, and where actors would have to select from among them which one best fits their conflict situation, and then behave accordingly and build agents that would be useful in other contexts; an experimental approach emphasizing more a Turing-like situation (Turing, 1950) where human subjects would have to recognize some agents as behaving like they themselves do. Simulations involving human participants look like a key feature of future research. It will certainly have to add more tools than just gaming and experiments—such as videogames, questionnaires, comments on agents and other players—and find ways to accumulate the information on actions and interpretation of rational humans. The sole use of a complex system and analysis of the parameters does not seem to be enough when facing the sets of simulated data of high complexity.

ACKNOWLEDGMENT

I wish to thank JSPS (Japan Society for the Promotion of Science), which gave me a grant, and Toru Ishida, who welcomed me in his lab for the year 2005. Many thanks to anonymous reviewers who made some comments and to Jean-Philippe Rennard who is a very efficient editor. Many thanks to experimental economists (Stephane Robin, Fabrice Lelec, Toshiji Kawagoe) and companion modelers (Francois Bousquet, Olivier Barreteau) for their explanations and comments on the chapter. The elements stated in this chapter are, however, my own responsibility.

REFERENCES

Anderson, P. W., Arrow, K. J., & Pines, D. (Eds.). (1988, September). The economy as an evolving complex system. *Proceedings of the Global Economy Workshop,* Santa Fe, NM. Santa Fe: Addison-Wesley.

Barreteau, O., & Bousquet, F. (2000). SHADOC: A multi-agent model to tackle viability of irrigated systems. *Annals of Operations Research, 94,* 139-162.

Barreteau, O. (2003). The joint use of role-playing games and models regarding negotiation processes: Characterization of associations. *Journal of Artificial Societies and Social Simulation, 6*(3). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/2/3.html/

Barreteau, O., Le Page, C., & D'Aquino, P. (2003). Role-playing games, models and negotiation processes. *Journal of Artificial Societies and Social Simulation, 6*(3). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/2/10.html/

Barreteau, O., Bousquet, F., Millier C., & Weber, J. (2004). Suitability of multi-agent simulations to study irrigation system viability: Applications to case studies in the Senegal River valley. *Agricultural Systems, 80,* 255-275.

Bateson, G. (1972). *Steps to an ecology of mind.* New York: Balantine.

Bousquet, F., Cambier, C., Mullon, C., Morand, P., Quensière, J., & Pavè, A. (1993). Simulating the interaction between a society and a renewable resource. *Journal of Biological Systems, 1,* 199-214.

Bousquet, F. (1994). *Des milieux, des poissons, des hommes: Étude par simulations multi-agents.* Paris: Orstom Editions.

Bousquet, F., Barreteau, O., D'Aquino, P., Etienne, M., Boissau, S., Aubert, S., et al. (2002). Multi-agent systems and role games: Collective learning processes for ecosystem management. In M. Janssen (Ed.), *Complexity and ecosystem management: The theory and practice of multi-agent approaches*. Cheltenham: Edward Elgar.

Brenner, T., & Vriend, N. J. (2005). On the behavior of proposers in ultimatum games. *Journal of Economic Behavior and Organization.*

Brewer, P., Huang, M., Nelson, B., & Plott C. (2002). On the behavioral foundations of the law of supply and demand: Human convergence and robot randomness. *Experimental Economics, 5,* 179-208.

Burger, P., & Luckman, T. (1966). *The social construction of reality: A treatise in the sociology of knowledge.* New York: Doubleday.

Castella, J., Trung, T., & Boissau, S. (2005). Participatory simulation of land-use changes in the northern mountains of Vietnam: The combined use of an agent-based model, a role-playing game, and a geographic information system. *Ecology and Society, 10*(1), 27. Retrieved November 15, 2005, from http://www.ecologyandsociety.org/vol10/iss1/art27/

Camerer, C. (2003). *Behavioral game theory: Experiments on strategic interaction*. Princeton, NJ: Princeton University Press.

Chong, J.-K., Camerer, C., & Ho, T.-H. (2005). A learning-based model of repeated games with incomplete information. *Games and Economic Behavior, 55*(2), 340-371.

D'Aquino, P., Le Page, C., Bousquet, F., & Bah, A. (2003). Using self-designed role-playing games and a multi-agent system to empower a local decision-making process for land use management: The SelfCormas experiment in Senegal. *JASSS, 6*(3). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/3/5.html/

Daré, W., & Barreteau, O. (2003). A role-playing game in irrigated system negotiation: Between play and reality. *JASSS, 6*(3). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/3/6.html/

Doran, J., & Palmer, M. (1995). The EOS project: Integrating two models of Paleolithic social change. In N. Gilbert & R. Conte (Eds.),

Artificial societies: The computer simulation of social life (pp. 103-125). London: UCL Press.

Duffy, J. (2001). Learning to speculate: Experiments with artificial and real agents. *Journal of Economic Dynamics and Control, 25,* 295-319.

Duffy J. (2006). Agent-based models and human subject experiments. In K. L. Judd & J. Tesfatsion (Eds.), *Handbook of computational economics* (Vol. 2). Amsterdam: North-Holland.

Duijn, M., Immers, L. H, Waaldijk, F. A, & Stoelhorst, H. J. (2003). Gaming approach route 26: A combination of computer simulation, design tools and social interaction. *Journal of Artificial Societies and Social Simulation, 6*(3). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/3/7.html/

Etienne, M. (2003). SYLVOPAST: A multiple target role-playing game to assess negotiation processes in sylvopastoral management planning. *Journal of Artificial Societies and Social Simulation, 6*(2). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/2/5.html

Etienne, M., Le Page, C., & Cohen, M. (2003). A step-by-step approach to building land management scenarios based on multiple viewpoints on multi-agent system simulations. *Journal of Artificial Societies and Social Simulation, 6*(2). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/2/2.html/

Fudenberg, D., & Levine, D.K. (1999). *The theory of learning in games.* Cambridge, MA: MIT Press.

Funtowicz, S. O., Martinez-Alier, J., Mundo, G., & Ravetz, J. R. (1999). *Information tools for environmental policy under conditions of complexity.* European Environment Agency.

Gilbert, N., & Abbott, A. (2005). Introduction. *American Journal of Sociology, 110*(4), 859-863.

Holling, C. S. (1978). *Adaptive environmental assessment and management.* London: John Wiley & Sons.

Langton, C. G. (Ed.). (1991). *Artificial life 1.* Redwood City, CA: Addison-Wesley.

Janssen, M., & Ahn, T. K. (2003). Adaptation vs. anticipation in public-good games. In J. Rouchier, B. Edmonds, & D. Hales (Eds.), *Proceedings of Model to Model Workshop, GREQAM,* Marseille, France (pp. 16-35).

Janssen, M., & Ostrom, E. (2006) Governing social-ecological systems. In K. L. Judd & L. Tesfatsion (Eds.), *Handbook of computational economics* (Vol. 2). Amsterdam: North-Holland.

Kirman, A. (1997). The economy as an interactive system. In W. B. Arthur, S. Durlauf, & D. Lane (Eds.), *The economy as an evolving complex system. Studies in the sciences of complexity XXVII* (no. 2). Santa Fe: Addison-Wesley.

Kirman, A., & Vriend, N. (2001). Evolving market structure: An ACE model of price dispersion and loyalty. *Journal of Economic Dynamics and Control, 25*(3-4), 459-502.

Lelec, F. (2005, July). Inertia and learning in the ultimatum game: Explaining experimental behaviors as a weakly learning agents system convergence. *Proceedings of AESCS,* Tokyo.

Minsky, M. (1985). *The society of mind.* New York: Simon & Schuster.

Moss, S., & Edmonds, B. (2005). Sociology and simulation: Statistical and qualitative cross-validation. *American Journal of Sociology, 110*(4), 1095-1131.

Nyarko, Y., & Schotter, A. (2002). An experimental study of belief learning using elicited belief. *Econometrica, 70,* 971-1005.

Patamadit, I., & Bousquet, F. (2005). The Thai traditional learning process in folk culture: Implications for the companion modeling approach. In F. Bousquet, G. Trébuil, & B. Hardy (Eds.), *Companion modeling and multi-agent systems for integrated natural resource management in Asia*. Metro Manila, Philippines: IRRI Press.

Roth, A. E., & Erev, I. (1995). Learning in extensive-form games: Experimental data and simple dynamic models in intermediate term. *Games and Economic Behavior. Special Issue: Nobel Symposium, 8,* 164-212.

Rouchier, J. (2004). Re-implementation of a multi-agent model aimed at sustaining experimental economic research: The case of simulations with emerging speculation. *Journal of Artificial Societies and Social Simulation, 6*(4). Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/6/4/7.html/

Rouchier, J., & Robin, S. (2006). Information perception and price dynamics in a continuous double auction. *Simulation and Gaming, 37,* 195-208.

Simon, H. (1969). *The sciences of the artificial.* Cambridge, MA: MIT Press.

Smith, V. (2002). Method in experiment: Rhetoric and reality. *Experimental Economics, 5,* 91-110.

Taylor, R. (2003). *Agent-based modeling incorporating qualitative and quantitative methods: A case study investigating the impact of e-commerce upon the value chain.* Doctoral thesis, Manchester University, UK.

Tesfatsion, L. (2006). Agent-based computational economics: A constructive approach to economic theory. In K. L. Judd & L. Tesfatsion (Eds.), *Handbook of computational economics* (Vol. 2). Amsterdam: North-Holland.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind, 59,* 433-460.

Zammito, J.H. (2004). *A nice derangement of epistemes: Post-positivism in the study of science from Quine to Latour.* Chicago: The University of Chicago Press.

KEY TERMS

Emergence: A phenomena is said to be emerging if it is the result of the parallel action of individuals of agents who did not have this phenomena as an objective. For example: individuals wish to go from A to B. A traffic jam appears, it was not wanted, nor predictable by the agent before it happened; it emerges from the presence of a number of individuals at the same time, going to the same place. Such an emergence can be detected by the people who are in it or not. Global warming is an emerging phenomenon that we can witness.

Experiment: Experiments in economics were created imitating psychological experiments. Individuals are gathered in a place where they are made to interact through computers so to eliminate any non-controlled information circulation. They are given a task for which economic theory predicts an optimal behavior. Actual behaviors are then observed, and the divergence between theory and actions is analyzed.

Parameters: Agent-based models are made of equations that are based on values—numbers or symbols. When a simulation is run, one set of such parameters is used. For example the number of agents in the system, the size of the grid on which the agents evolve, the rules that define how the resource is renewed, the time

that each agent has for communication in a time-step, and so forth. Since a complex system cannot be analyzed analytically, what is studied is its sensitivity to each parameter. If the system displays similar behavior when one parameter varies within a certain range, one can say that this behavior is robust regarding this parameter. If the system's behavior is highly correlated to the change of a parameter, then the model is considered dependent on this parameter.

Protocols: A protocol is the organization of the capture of data in empirical research. In the case of experiments, it will include the way individuals are selected, how the instructions are transmitted, as well as the task that the participants have to achieve in the experiment. In the case of experiments, it will include the type of study that is led before organizing meetings, the number of meetings and participants, the choice of the stakeholders, the tools used for communication, and the number of modifications of these tools.

Rationality: An individual's rationality is the process of decision that organizes its actions. Rationality has to be separated from *perfect rationality*, which is an economics concept defined by the ability to choose the best action in a situation. Rationality can be suboptimal (especially in the case of *bounded-rationality* as defined by Simon). For an artificial agent, rationality is defined by an algorithm which associates information gathered by the agent to a choice of action.

Role-Playing Game: Role-playing games are games that are organized around a scenario where each player takes a role that partly defines its abilities and motivation, and where a story is commonly built from the original script. When the aim is just recreational, the master of the game usually makes up in his mind an environmental accident to stimulate the imagination of players. When the role-playing game is used to sustain discussions in a group, events are either: logically deduced from the definition of the environment dynamics, or from scenarios that are proposed by players or have been observed in the represented setting.

Setting: A setting is made up of the whole system of communication, role, and timing in which the agents or individuals are immerged. It is the artificial organization (in a simulation, a game, or an experiment) that represents the institution that is studied.

Stakeholder: In a decision-making process, a stakeholder is one of those involved in the situation as it is analyzed by organizers, and who have an interest which is at that moment in conflict with others. Along the mediation process, the definition of situation can change and some new actors can be involved. Sometimes people are identified as stakeholders, but do not recognize themselves as such and do not wish to participate in common decision making.

Validation: Facing any kind of simulation, there is a necessity to define a way to assess the quality of the model in its representation of quality. In the case of simulation models, it is important to first check that the implemented model corresponds to the model description (*internal validation* or *verification*). Then one has to see if the model has some structural equivalence with the observed real world, to make sure that it is interesting to use the simulation model to understand, predict, or exemplify this reality. It is common to consider that depending on the aim of the modeling process, the model cannot be validated in the same manner.

Chapter XVI
Modeling Qualitative Development

Andreas Pyka
University of Augsburg, Germany

ABSTRACT

This chapter introduces agent-based modeling as a methodology to study qualitative change in economic systems. The need to focus on qualitative developments is derived from evolutionary economics, where the quantitative orientation of mainstream economic approaches is strongly criticized. It is shown that agent-based models can cope with the challenges of an evolutionary setting and fulfill the requirements of modeling qualitative change. In particular agent-based models allow a detailed representation of knowledge and the underlying dynamics, which are considered the major driving force of economic growth and development. The chapter also gives an illustrative example of an agent-based model of innovation processes organized in networks of actors.

INTRODUCTION

The tremendous development of and easy access to computational power within the last 30 years has led to the widespread use of numerical approaches in almost all scientific disciplines. While the engineering sciences focused on the applied use of simulation techniques from the very beginning, in the social sciences most of the early examples of numerical approaches were purely theoretical.

There are two reasons for this. First, since the middle of the 20th century, starting with economics, equilibrium-oriented analytical techniques flourished and were developed to a highly sophisticated level. This led to the widely shared view that within the elegant and formal framework of linear analysis offered by neoclassical economics, the social sciences could reach a level of accuracy not previously thought to be possible.

Second, within the same period, new phenomena of structural change exerted a strong influence on the social and economic realms. Despite the mainstream neoclassical successes in shifting the social sciences to a more mathematical foundation, an increasing dissatisfaction with this approach emerged. For example,

by the 1970s the benchmark of atomistic competition in neoclassical economics had already been replaced by the idea of monopolistic and oligopolistic structures. A similar development emphasizing positive feedback effects and increasing returns to scale caused by innovation led to the attribute "new" in macroeconomic growth theory in the 1980s.

In addition to these stepwise renewals of mainstream methodology, an increasingly larger group claimed that the general toolbox of economic theory, emphasizing rational behavior and equilibrium, is no longer suitable for the analysis of complex social and economic changes. In a speech at the International Conference on Complex Systems organized by the New England Complex Systems Institute in 2000, Kenneth Arrow stated that until the 1980s, the "sea of truth" in economics lay in simplicity, whereas since then it has become recognized that "the sea of truth lies in complexity." Adequate tools have to include the heterogeneous composition of agents (e.g., Saviotti, 1996), the possibility of multilevel feedback effects (e.g., Cantner & Pyka, 1998), and a realistic representation of dynamic processes in historical time (e.g., Arthur, 1988). These requirements are congruent with the possibilities offered by simulation approaches. It is not surprising that within economics the first numerical exercises were within evolutionary economics, where phenomena of qualitative change and development are at the front of the research program.

The first generation simulation models were highly stylized and did not focus on empirical phenomena. Instead, they were designed to analyze the logic of dynamic economic and social processes, exploring the possibilities of complex systems behavior. However, since the end of the 1990s, more and more specific simulation models aiming at empirically observed phenomena have been developed focusing on the interaction of heterogeneous actors responsible for qualitative change and development processes. Modelers have had to wrestle with an unavoidable trade-off between the demands of a general theoretical approach and the descriptive accuracy required to model a particular phenomenon. A new class of simulation models has shown to be well adapted to this challenge, basically by shifting outwards this trade-off (e.g., Gilbert & Troitzsch, 1999): so-called agent-based models are increasingly used for the modeling of socioeconomic developments.

This chapter deals with the changed requirements for modeling caused by the necessity to focus on qualitative developments which is generally highlighted within evolutionary economics and the possibilities given by agent-based models. The next section is concerned with the importance of an analysis of qualitative development, and it is shown that evolutionary economics is offering an adequate framework for this. A focus on agent-based-modeling as *the* tool that allows incorporating endogenously caused development processes follows, and the next section gives an illustrative example. Finally, the whole story is summarized.

QUALITATIVE CHANGE IN AN EVOLUTIONARY ECONOMICS PERSPECTIVE

When concerned with the examination of change and development within industrialized economies, economists usually focus on the movement of certain variables they consider a good description of the basic effects of economic growth and development. In mainstream economics, the phenomenon of economic development is, for example, empirically analyzed on the macro level as the improvement of total factor productivity in time, which lowers prices and leads to the growth of incomes. Accordingly, most often the GDP per capita is used as an indicator describing economic development

in a quantitative fashion. Although it is impressing to observe the growth of income in economies over a long time span, this indicator does not give any idea about the structural and qualitative dimensions underlying economic development. This becomes even more obvious on the sectoral level, where the analysis is most often restricted to long-run equilibrium structure describing the number of firms in a particular industry without putting emphasis on, for example, the organizational forms to be observed in the interaction patterns encompassing not only competition, but increasingly also cooperative interactions leading to important network structures (Ahrweiler, Gilbert, & Pyka, 2001). By restricting their analysis on the quantitative dimension, the economic mainstream implicitly confines itself to the analysis of a system characterized by a constant set of activities basically neglecting innovation processes: "Economic growth can be described at the macro-economic level, but it can never be explained at that level … Economic growth results from the interaction of a variety of actors who create and use technology and demanding costumers" (Eliasson & Carlsson, 2004).

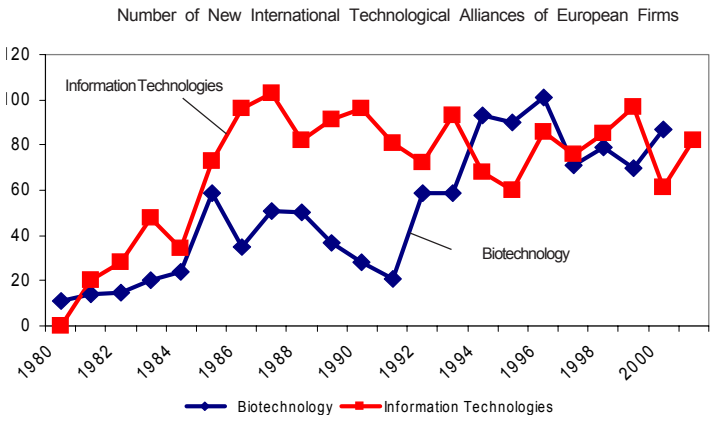
However, in less orthodox economic approaches it is argued, and it is indeed also one of Schumpeter's major contributions, that economic development does also include prominently qualitative changes, not only as an outcome but also as an essential ingredient which justifies us to speak of transformation processes going on. Qualitative change manifests itself basically via innovation of different categories of which technological innovation very likely is among the most important ones. Qualitative change is the transformation of an economic system, characterized by a set of components and interactions into another system, with different components and different interrelationships (e.g., Saviotti, 1996). An analysis of qualitative change therefore necessarily has to include the actors, their activities, and the objects that are responsible for the ongoing economic development. An example for the significance of qualitative changes referring to the network example above can be found in Figure 1, which displays the increasing importance of collaborative R&D in knowledge-intensive industries. What strikes immediately is that collaborative R&D obviously is not only a temporary phenomenon, as stated in equilibrium-oriented neoclassical transaction costs theory, but very likely a persistent phenomenon of increasing importance (Pyka, 2002). Of course there are many other variables which also reflect the importance of the qualitative dimensions of economic development, for example, on a macro level the changing composition of the employment structure (*Fourastier Hypothesis*), on a meso level the regional specialization patterns, or on a micro level the obsolescence of old and the emergence of new knowledge like the biotechnology revolution in pharmaceuticals, to name a few. By its very nature, the transformation of an economic system is a multi-facetted phenomenon. Accordingly, it is misleading to focus only on quantitative changes when analyzing the driving factors of the transformation of economic systems over time. To better understand the mechanisms and dynamics behind the observed developments, one has to explicitly include the qualitative dimensions. To achieve this, economic analysis has to consider—besides the prevailing cost-orientation—an important knowledge and learning orientation.

The following paragraphs are concerned with the implications of this knowledge orientation, which can also be considered as the heart of the matter of evolutionary economics.

Knowledge-Based Approach of Evolutionary Economics

It is beyond the scope of this contribution to discuss in detail the criticism brought forth by

Figure 1. Increasing importance of R&D collaboration (Science and Engineering Indicators, 2002)



evolutionary economics with respect to assumptions underlying the mainstream economic reasoning. A major discussion can be found, among others, in Dopfer (2001). For our purposes it is sufficient to mention three points that evolutionary economists claim to be of outstanding importance in the discussion of economic development and which are incompatible with traditional approaches. These points are also constitutive for that strand of literature within evolutionary economics which is concerned with industry evolution and technological progress—namely, the Neo-Schumpeterian approach (Hanusch & Pyka, 2006). First of all, Neo-Schumpeterian theory wants to explain how innovations emerge and diffuse over time. A specific feature of these processes is uncertainty, which cannot be treated adequately by drawing on stochastically distributions referring to the concept of risk. Therefore, the assumption of perfect rationality underlying traditional models cannot be maintained; instead the concepts of bounded and procedural rationality are invoked. Consequently, actors in Neo-Schumpeterian models are characterized by incomplete knowledge bases and capabilities. Closely connected, the second point concerns the important role heterogeneity and variety plays. Due to the assumption of perfect rationality, in traditional models homogeneous actors and technologies are analyzed. Heterogeneity as a source of learning and novelty is by and large neglected, or treated as an only temporary deviation. Finally, the third point deals with the time dimension in which learning and the emergence of novelties take place. By their very nature, these processes are truly dynamic, meaning that they occur in historical time. The possibility of irreversibility, however, does not exist in the mainstream approaches, relying on linearity and equilibrium.

Thus, traditional economic theories, summarized under the heading of incentive-based approaches, with their focus on cost-based and rational decisions, are excluding crucial aspects of actors' behaviors and interactions, which are influenced by a couple of factors lying by their very nature beyond the scope of these approaches. Although, of course, cost-benefit calculations play an important role, the actors' behavior is influenced additionally by several other factors such as learning, individual and collective motivation, trust, and so forth. It is the role of these factors the knowl-

edge-based approach of evolutionary economics explicitly takes into account.

By switching to the knowledge-based perspective, the Neo-Schumpeterian approaches have realized a decisive change in the analysis of the transformation of economic systems. In this light the introduction of novelties mutate from optimal cost-benefit considerations to collective experimental and problem-solving processes (Eliasson, 1988). The knowledge base of the actors is no longer perfect; instead a gap between the competences and difficulties (C-D gap) that are to be mastered opens up (Heiner, 1983). There are two reasons for this C-D gap when it comes to innovation: on the one hand, technological uncertainty introduces errors and surprises. On the other hand, the very nature of knowledge avoids an unrestricted access. Knowledge in general, and new technological know-how in particular, are no longer considered as freely available, but as local (technology specific), tacit (firm specific), and complex (based on a variety of technology and scientific fields). To understand and use the respective know-how, specific competences are necessary which have to be built up in a cumulative process in the course of time. Following this, knowledge and the underlying learning processes are important sources for the observed heterogeneity among agents.

Challenges for Analyzing Qualitative Change

From the discussion above we can identify two major challenges for an analysis of qualitative change. The first challenge is that a theoretical framework adequately displaying our notion of qualitative change has to incorporate concepts that comply with the notion of development of evolutionary economics. Basically this refers to path-dependencies, dynamic returns, and their interaction as constitutive ingredients for evolutionary processes.

The second challenge is that we generally have to focus on both the micro and meso levels of the economy, as to our understanding the term qualitative change refers to a changing composition and interaction of and in the economic system. In doing so, we can identify stylized facts that are considered of importance when qualitative change in an economy is considered. The most obvious ones follow.

First, an increasing importance of knowledge generation and diffusion is observed. This coins the notion of a transformation of the economy into a knowledge-based economy. Second, this is accompanied by a continuously increasing specialization, and relates to an increasing variety of products and services coexisting. Third, specialization and differentiations go hand in hand with an increasing importance of (market and non-market) interactions between the agents. Fourth, behind this increasing variety we observe innovation processes that at the same time improve efficiency and the quality. Fifth, this innovation process is driven by competition selecting between technological alternatives. Finally, the environmental constraints can be considered as filter and focusing devices either supporting or hindering the diffusion of innovations.

Once the relevance of these facts for the transformation of an economy are accepted, the research has to account for those developments adequately.

Micro- and Meso-Perspective

Obviously this aim can only be accomplished by abandoning an aggregate perspective and instead focusing on a micro- or meso-level population approach (Metcalfe, 2001). This allows for examining diverse agents, their interaction, and the knowledge-induced transformation of both. Modeling has to take into account the importance of micro-macro-micro feedback effects. In their decisions, actors obviously

consider macro constraints, but they also exert a significant influence on the altering of these constraints (Dopfer, 2001). The interrelated inspection of the meso and the micro level reflects the idea that analysis on the aggregated meso level relies on description, whereas the analysis of the micro level focuses on explanation of the phenomena found on the meso level (Dopfer, 2001).

Knowledge

Considering this will lead to a revision of standard economic models, as analysis here follows reality closely. Traditional 'production functions' include labor, capital, materials, and energy. Knowledge and technology are only external influences on production. However, recent analytical approaches have been developed allowing the explicit consideration of knowledge as well as learning as a mean for acquiring new knowledge. Improvements in the knowledge base are likely not only to increase the productive capacity of other production factors, leading to the introduction of new products as a visible outcome of the transformation process, but also to alter the organizational processes of knowledge creation, namely the interrelationships between the actors. Thus, transformation relates to a result *and* an extremely important process dimension.

A MODELING APPROACH ALLOWING FOR QUALITATIVE CHANGE: AGENT-BASED MODELING

An exploration of settings fulfilling the above requirements needs numerical techniques. Although simulation analysis comes in various flavors, most of them reflect Boulding's call that we need to develop "mathematics which is suitable to social systems, which the sort of 18th-century mathematics which we use is not" (Boulding, 1991). An increasingly growing literature today is concerned with the application of so-called agent-based models (ABMs). This approach consists of a decentralized collection of agents acting autonomously in various contexts. The massively parallel and local interactions can give rise to path dependencies, dynamic returns, and their interaction. In such an environment, global phenomena as the development and diffusion of technologies, the emergence of networks, herd-behavior, and so forth—which cause the transformation of the observed system—can be modeled. This modeling approach focuses on depicting the agents, their relationships, and the processes governing the transformation. Very broadly, the application of ABMs offers two major advantages with respect to the knowledge and learning orientation.

The first advantage of ABMs is their capability to show how collective phenomena come about, and how the interaction of the autonomous and heterogeneous agents leads to their genesis. Furthermore ABMs aim at the isolation of critical behavior in order to identify agents that more than others drive the collective result of the system. They also endeavor to single out points of time where the system exhibits qualitative rather than sheer quantitative change (Tesfatsion, 2001). In this light it becomes clear why ABMs conform to the principles of evolutionary economics (Lane, 1993a, 1993b). It is *the* modeling approach to be pursued in evolutionary settings. The second advantage lies in the possibility to use ABMs as computational laboratories to explore various institutional arrangements, various potential paths of development so as to assist and guide, for example, firms, policymakers, and so forth, in their particular decision context.

ABMs thus use methods and insights from diverse disciplines such as evolutionary economics, cognitive science, and computer science in an attempt to model the bottom-up emergence of phenomena and the top-down influence of the collective phenomena on individual behavior. The recent developments in new techniques, in particular the advent of powerful tools of computation, open up the opportunity for economists to model economic systems on a more complex basis (Tesfatsion, 2001).

In the following sections a typical example for an ABM is introduced in order to highlight the specialties of this methodology. In particular the model deals with the emergence of innovation networks where economic agents jointly develop and share at least some of their knowledge. As the focus of this chapter lies on the methodology of ABMs, we cannot go into detail with respect to the economic implications of the model, but refer instead to Ahrweiler et al. (2004), where all the economic concepts used in the model are described in detail.

AN EXAMPLE: THE SKIN MODEL

SKIN, the acronym for Simulating Knowledge in Innovation Networks, is a multi-agent model containing heterogeneous agents which act in a complex and changing environment. Its agents are innovative firms who try to sell their innovations to other agents and end users, but who also have to buy raw materials or more sophisticated inputs from other agents in order to produce their outputs. This basic model of a market is extended with a representation of the knowledge dynamics in and between the firms. Each firm tries to improve its innovation performance and its sales by improving its knowledge base through adaptation to user needs, incremental or radical learning, and cooperation and networking with other agents. In the next paragraphs, the basic elements and processes are introduced in order to illustrate exemplarily the architecture of an ABM.

The Agents

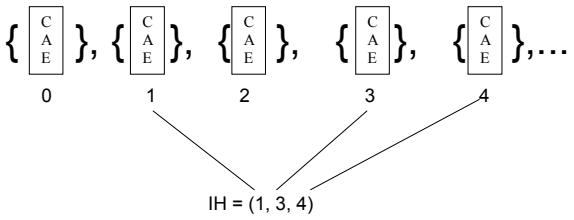
A SKIN agent is a firm with an individual knowledge base. This knowledge base is called its *kene* (Gilbert, 1997) and consists of a number of "units of knowledge." Each unit is represented as a triple, consisting of a firm's *capability C* in a scientific, technological, or business domain (e.g., biochemistry), represented by an integer randomly chosen from the range of 1..1000; its *ability A* to perform a certain application in this field (e.g., a synthesis procedure or filtering), represented by an integer randomly chosen from the range 1..10; and the *expertise level E* the firm has achieved with respect to this ability (represented by an integer randomly chosen from the range 1..10). The firm's kene is its collection of C/A/E-triples.

When it is set up, each firm has a stock of initial *capital.* It needs this capital to produce for the market and to improve its knowledge base, and it can increase its capital by selling products. The amount of capital owned by a firm is a measure of its size.

Most firms are initially given a standard amount of starting capital, but in order to model differences in firm size, a few randomly chosen firms can be given extra capital (set using the "n-big-firms" slider on the interface—see Figure 6).

Figure 2. The kene of a firm

Figure 3. Forming an innovation hypothesis



The Market

Firms apply their knowledge to create innovative products that have a chance to be successful in the market. The special focus of a firm, its potential innovation, is called an *innovation hypothesis*. In the model, the innovation hypothesis (IH) consists of a subset of the firm's kene triples.

The underlying idea for an innovation, modeled by the innovation hypothesis, is the source an agent uses for its attempts to make profits on the market. Developing the innovation hypothesis into a *product* is a mapping procedure where the capabilities and abilities of the innovation hypothesis are used to compute an index number that represents the product.

A firm's product, P, is generated from its innovation hypothesis as

$$P = (C_1 * A_1) + (C_3 * A_3) + (C_4 * A_4) + \ldots \text{ modulus } N \quad (1)$$

(where N is the total number of products ever possible within the model).

The product has a certain *quality* which is also computed from the innovation hypothesis in a similar way, but using a product of the abilities and the expertise levels for each triple in the innovation hypothesis.

In order to start production, the agent needs some raw materials or more sophisticated inputs from other agents. What exactly it needs is also determined by the underlying innovation hypothesis: the kind of material required for an input is obtained by selecting subsets from the innovation hypotheses and applying the standard mapping function (equation 2).

In order for an agent to be able to engage in production, all the inputs need to be available on the market (i.e., provided by other agents). If the inputs are not available, the agent is not able to produce and has to give up this attempt to innovate. If there is more than one supplier for a certain input, the agent will choose the one at the cheapest price and, if there are several similar offers, the one with the highest quality.

$$\text{Input 1: } (C_1 * A_1 + C_2 * A_2) \text{ modulus } N$$

$$\text{Input 2: } (C_3 * A_3 + C_4 * A_4 + C_5 * A_5) \text{ modulus } N \quad (2)$$

If the agent can go into production, it has to find a price for its own product which takes account of the input prices it is paying and a possible profit margin. While the simulation starts with assuming that all agents have a product that they can sell and with the product prices set at random, as the simulation proceeds, a price adjustment mechanism ensures that the selling price will at least equal the total cost.

An agent will then buy the requested inputs from its suppliers using its capital to do so. It produces its output and puts it on the market. Agents follow a standard pricing strategy such that if a product sells, its price will be increased, while if it does not sell, the price is reduced until

Figure 4. A firm's input requirements

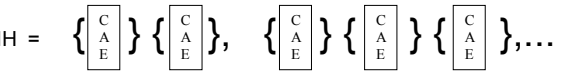
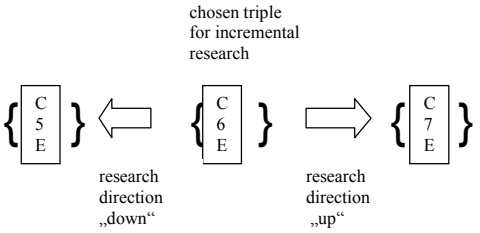


Figure 5. Incremental research



the cost of production is reached. In this way, agents are able to adapt their prices to demand.

In making a product, an agent applies the knowledge in its innovation hypothesis, and this increases its expertise. This is the way *learning by doing/using* is modeled. The expertise level of the triples in the innovation hypothesis is increased by 1 and the expertise levels of the other triples are decremented by 1. Unused triples in the kene eventually drop to an expertise level of 0 and are deleted from the kene; the corresponding abilities are "forgotten."

The Environment

Within the model, there are two world settings for the agents' environment. With the first, the model represents a closed market in which the agents trade only with each other as equals, sharing the same attributes and rules. Each agent buys its inputs from other agents and itself produces an output which must then be bought by other agents. The alternative world is a market where external sources and purchasers interact with the firm population. With this setting, the model includes some supplier firms and some customer firms. The supplier firms try to sell *raw materials*—the basic elements necessary for the production of goods—but they do not buy anything. The customer firms are 'end users' who buy products without producing anything themselves. The implementation of the model allows for switching between these settings in order to experiment with the two market conditions (see the *open-system* slider in Figure 6).

Learning and Cooperation: Improving Innovation Performance

In trying to be successful, the firms are highly dependent on their innovation hypothesis—that is, on their kene. If a product does not meet any demand, the firm has to adapt its knowledge in order to produce something else for which there are customers. In the model, a firm can choose between different ways of improving its performance, either alone or in cooperation, and either in an incremental or in a more radical fashion. All strategies have in common that they are costly: the firm has to pay a *tax* as the cost of applying an improvement strategy.

Incremental Research

If a firm's previous innovation has been successful, it will continue selling the same product. However, if the previous profit was below a certain threshold, it considers that it is time for change. If the firm still has enough capital, it will carry out incremental research (R&D in the firm's labs). Performing incremental research means that a firm tries to improve its product by altering one of its abilities chosen from the triples in its innovation hypothesis while generally sticking to its focal capabilities. This is to exploit the action space available for a certain capability. The ability in each triple is considered to be a point in the respective capability's action space. To move in the action space means to go up or down by 1 on the integer scale, thus allowing for two possible incremental *research directions*. Initially, the research direction of a firm is set at random. Later it learns to adjust to success or failure: if a move in the action space has been successful,

the firm will continue with the same research direction within the same triple; if it has been a failure, the firm will randomly select a different triple from the innovation hypothesis and try again with a random research direction on the triple's ability.

Radical Research

A firm under serious pressure that is in danger of becoming bankrupt will turn to more radical means to prevent its exit. In this situation, a firm can choose to perform radical research to explore a completely different area of market opportunities. In the model, an agent under financial pressure turns to a new innovation hypothesis after first "inventing" a new capability for its kene. This is done by randomly changing one capability in the kene for a new one and then forming an innovation hypothesis from its kene set.

Partnerships

An agent may consider partnerships and networks in order to learn from other agents, to exploit external knowledge sources. Within the model we can switch between a scenario where partnerships and networks are prohibited and a scenario where they are allowed (see Figure 6, slider *collaboration*). In the latter scenario, the decision whether and with whom to cooperate is based on mutual observations. The information a firm can gather about others is provided by a marketing feature: to advertise its product, a firm publishes the capabilities used in its innovation hypothesis. The firm's advertisement is then the basis for decisions by other firms to form or reject cooperative arrangements.

In experimenting with the model, we can choose between two different partner search strategies, both of which compare the firm's own capabilities in its innovation hypothesis and the possible partner's capabilities as seen in its advertisement. Applying the conservative strategy, a firm will be attracted by a possible partner who has similar capabilities; using a progressive strategy the attraction is based on the difference between the capability sets (see Figure 6, slider *partnership-strategy*).

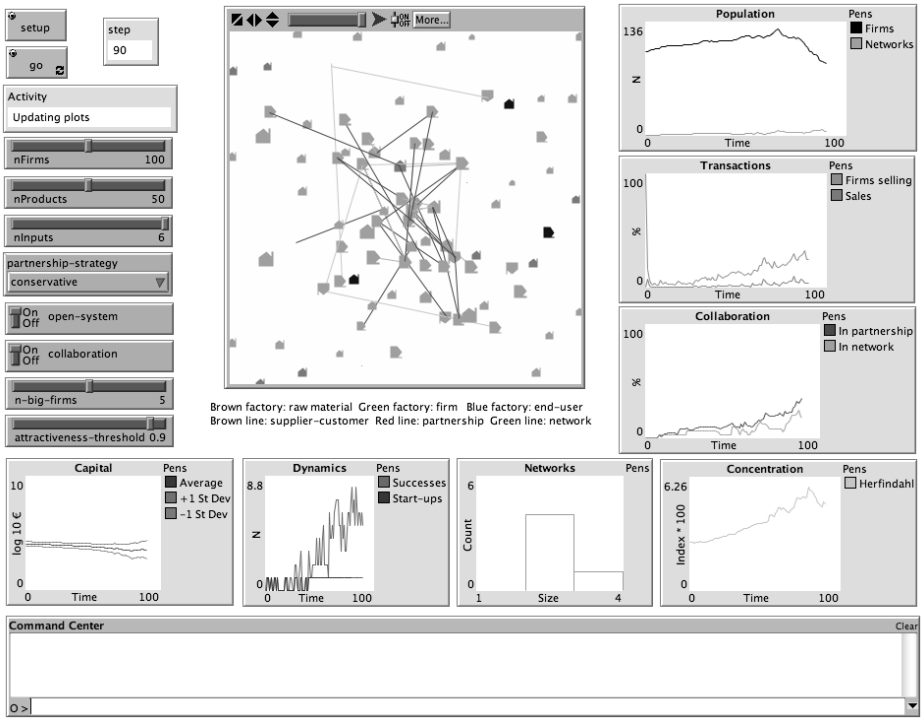
Previously good experience with former contacts generally augurs well for renewing a partnership. This is mirrored in the model: to find a partner, the firm will look at previous partners first, then at its suppliers, customers, and finally at all others. If there is a firm sufficiently attractive according to the chosen search strategy (i.e., with attractiveness above the *attractiveness threshold*), it will stop its search and offer a partnership. If the possible partner wishes to return the partnership offer, the partnership is set up.

To learn from the partner, a firm will add the triples of the partner's innovation hypothesis to its own. It will take care that it will only take triples which are different from its own triples in the innovation hypothesis: the expertise levels of the triples taken from the partner are set down to 1 in order to mirror the difficulty of integrating external knowledge. Once the knowledge transfer has been completed, each firm continues to produce its own product.

Networks

If the firm's last innovation was successful—that is, the amount of its profit in the previous round was above a threshold—and the firm has some partners at hand, it can initiate the formation of a network. This can considerably increase its profits because the network will try to create innovations as an autonomous agent in addition to those created by its members. It will distribute any rewards to its members who, in the meantime, can continue with their own attempts, thus providing a double chance for profits. However, the formation of networks is

Figure 6. The interface of the model



costly, which has two consequences: only firms with enough capital can form or join a network and no firm can be member of two networks at the same time.

Networks are *normal* agents—they get the same amount of initial capital as other firms and can engage in all the activities available to other firms. The kene of a network is the union of the triples from the innovation hypotheses of all its participants. If a network is successful, it will distribute any earnings above the amount of the initial capital to its members; if it fails and becomes bankrupt, it will be dissolved.

Start-Ups

If the sector is successful, new firms will be attracted into it. This is modeled by adding a new firm to the population when any existing firm makes a substantial profit. The new firm is a clone of the successful firm, but with its kene triples restricted to those in the successful firm's advertisement, and an expertise level of 1. This models a new firm copying the characteristics of those seen to be successful in the market. As with all firms, the kene may also be restricted because the initial capital of a start-up is limited and may not be sufficient to support the copying of the whole of the successful firm's knowledge base.

THE IMPLEMENTATION

The model has been programmed using NetLogo (http://ccl.northwestern.edu/netlogo/) and is available from the author upon request.

With a complex model such as this one, extensive experiments have to be carried out to understand its behavior and the sensitivity of the output to variations in the input parameters. Obviously such analysis is beyond the scope of this chapter. The intention is to give a detailed overview on the model's architecture in order to illustrate how the requirements for modeling knowledge dynamics and qualitative change are fulfilled in an ABM. For detailed analysis and various experiments, see Ahrweiler et al. (2001, 2004). Nevertheless, an impression of how the model performs using standard parameter settings is given in the next paragraphs.

The main graphic window of Figure 6 shows about 100 firms (represented by the small 'factory' shapes). Factory icons on their side mark firms that were created after the start of the simulation (i.e., they are start-ups), and those that are upside down are network firms—firms producing on behalf of a network, with a kene based on the union of the kenes of the network members. The size of the icons indicates the amount of capital its firm possesses.

The lines indicate partnerships, supplier relationships, and network linkages between firms. Those not involved in any relationship have been moved by the layout algorithm to the margins of the display. There are five networks (one with four members, and four with three members). The networks are shown with lines interconnecting all their members.

The display is surrounded by graphs monitoring various aggregate aspects of the system. At the top right the growth in the population of firms as start-ups are added, and the slow growth in the number of networks is shown. The graph below shows the percentage of firms that have products on the market (*Firms Selling*) and the percentage that have made a sale (*Sales*); the latter is always less than the former because some firms are unable to find customers prepared to buy at the price proposed for the product. The third graph down the right-hand side indicates the percentage of firms that are involved in either at least one partnership or in a network. The bottom right graph shows a measure of the distribution of funds in the market, the Herfindahl concentration index H_t,

$$H_t = \sum_n s_i^t$$

where s^t_i is the relative capital of firm i, which measures the distribution of capital among the firms.

The Networks histogram shows the number of firms in each network, and the Dynamics plot indicates, on the upper Successes graph, the number of firms that have exceeded the threshold of profit that indicates a successful innovation (the 'success threshold') and, on the lower Start-Ups graph, the number of new firms entering the market at each round. The Capital plot in the bottom left corner shows the average capital of the firms. The various experiments performed so far look at the development of different network structures using methods from social network analysis and graph theory.

CONCLUSION

In this chapter we attempted to provide an introduction to ABMs in economics. We began with a discussion of the main motivations that, in the last few years, led many scholars to supplement "mainstream" treatments with alternative approaches, rooted in "more realist" assumptions such as heterogeneity, interactions, bounded rationality, endogenous novelty, and so on. After presenting the building blocks shared by this class of models, an exemplary model of emerging innovation networks is introduced.

In our view, the attempt to model the aggregate dynamics of decentralized economies on the basis of a more detailed microfoundation such as the one postulated by ABMs is the

primary requirement to pursue one of the most prominent challenges in social sciences today, namely the analysis of qualitative change. Our discussion suggests that ABMs are offering an adequate framework for this, overcoming the severe restrictions which orthodox economic approaches are confronted with. By emphasizing the role of true uncertainty and irreversibility, one is able to model qualitative development as an endogenous process driven by the agents and their interactions.

ABMs allow for an explicit consideration of these characteristic features, and therefore can be considered as *the* modeling tool for the analysis of qualitative development and transformation processes. In a way, ABMs can be considered a systemic approach, allowing the consideration and integration of different social *realities* which makes them an extremely valuable tool for the analysis of social processes which can be generally considered as multifaceted phenomena.

The field of agent-based modeling in social sciences is far from its maturity. Many issues, especially methodological ones, are still debated, both on the model development side and on the model analysis side. Here, as a way of conclusion, we will try to briefly mention some of the most crucial ones.

First, on the model building and development side, one faces a huge heterogeneity in the way agents and their behavioral and interaction rules are assumed and implemented. In the relevant literature, one often deals with many structurally different ABMs addressing very similar issues. This practice, which is ultimately caused by the flexibility of programming languages and their heterogeneity, can certainly turn out to be a plus, because it might favor a better understanding of the deep causes of a given phenomenon. However, it can also generate an inherent impossibility to compare different models and to pursue a coherent procedure of model improvement.

Second, and relatedly, an agent-based modeler will often end up with an over-parameterized model. In order to limit as much as possible all critiques regarding the robustness of results to different parameterizations and initial conditions, an exhaustive exploration of both parameters' and initial conditions' sets is required.

Third, even when the foregoing critiques have been carefully considered, an agent-based modeler should be aware of the fact that all his/her results could be heavily affected by the particular sets of behavioral and interaction rules that he/she has assumed. Those rules are often kept fixed across time. This can be justified by the observation that the rules themselves typically change slower that the variables which they act upon. However, in the evolutionary spirit informing ABMs, a necessary step would be that of modeling the rules themselves as endogenously changing objects. An example here concerns learning and decision rules, which can be endogenously modified by the agents along the process.

Finally, on the normative side, it must be noticed that so far ABMs have almost exclusively addressed the issue of replication of stylized facts. However, ABMs can and should be employed to address policy issues as well. A need for increasingly normative-oriented ABMs delivering policy implications and out-of-sample predictions is nowadays strongly felt in the community. Thanks to the flexibility and the power of agent-based approaches, it is easy to conceive frameworks where policy experiments are carried out to evaluate the effectiveness of different policy measures, for a range of different institutional setups and behavioral rules.

ACKNOWLEDGMENT

For this chapter I profited very much from discussions and joint work with Petra Ahrweiler, Bernd Ebersberger, Giorgio Fagiolo, Nigel Gil-

bert, and Thomas Grebel. Obviously, the usual disclaimer remains.

REFERENCES

Arthur, W. B. (1988), Competing technologies: An overview. In G. Dosi et al. (Eds.), *Technical change and economic theory* (pp. 590-607). London: Pinter.

Ahrweiler, P., Gilbert, N., & Pyka, A. (2001). Innovation networks—A simulation approach. *Journal of Artificial Societies and Social Simulation, 4*(3). Retrieved October 2006, from http://jasss.soc.surrevy.ac.uk/4/3/8.html

Ahrweiler, P., Gilbert, N., & Pyka, A. (2004). Simulating knowledge dynamics in innovation networks. In R. Leombruni & M. Richiardi (Eds.), *Industry and Labor Dynamics: The Agent-Based Computational Economics Approach. Proceedings of the Wild@Ace 2003 Conference*. Singapore: World Scientific Press.

Boulding, K. E. (1991). What is evolutionary economics? *Journal of Evolutionary Economics, 1,* 9-17.

Cantner, U., & Pyka, A. (1998). Technological evolution—An analysis within the knowledge-based approach. *Structural Change and Economic Dynamics, 9,* 85-108.

Dopfer, K. (2001). Evolutionary economics: Framework for analysis. In K. Dopfer (Ed.), *Evolutionary economics: Program and scope*. Boston: Kluwer Academic.

Eliasson, G. (1988). Schumpeterian innovation, market structure and the stability of industrial development. In H. Hanusch (Ed.), *Evolutionary economics—Applications of Schumpeterian ideas*. Cambridge, Cambridge University Press.

Eliasson, G., & Carlsson, B. (2004). Industrial dynamics and economic growth. *Industry and Innovation, 10,* 435-455.

Gilbert, N. (1997). A simulation of the structure of academic science. *Sociological Research Online, 2*(2). Retrieved October 2006, from http://www.socresonline.org.uk/cgi-bin/abstract.pl?2/2/3.html

Gilbert, N., & Troitzsch, K. (1999). *Simulation for the social scientist.* Milton Keynes, UK: Open University Press.

Hanusch, H., & Pyka, A. (2006). *The Elgar companion to neo-Schumpeterian economics*. Cheltenham, UK: Edward Elgar.

Heiner, R. A. (1983). The origin of predictable behavior. *American Economic Review, 73,* 560-595.

Lane, D. (1993a). Artificial worlds and economics, part I. *Journal of Evolutionary Economics, 3,* 89-107.

Lane, D. (1993b). Artificial worlds and economics, part II. *Journal of Evolutionary Economics, 3,* 177-197.

Metcalfe, J. S. (2001). Evolutionary approaches to population thinking and the problem of growth and development. In K. Dopfer (Ed.), *Evolutionary economics: Program and scope* (pp. 141-164). Boston/Dordrecht/London: Kluwer Academic.

Pyka, A. (2002). Innovation networks in economics—From the incentive-based to the knowledge-based approaches. *European Journal of Innovation Management, 5*(3), 152-163.

Saviotti, P. P. (1996). *Technological evolution, variety and the economy*. Aldershot, UK: Edward Elgar.

Tesfatsion, L. (2001). Agent-based modeling of evolutionary economic systems. *IEEE Transactions on Evolutionary Computation, 5,* 1-6.

Chapter XVII
Agent-Based Modeling with Boundedly Rational Agents

Eva Ebenhöh
University of Osnabrück, Germany

Claudia Pahl-Wostl
University of Osnabrück, Germany

ABSTRACT

This chapter introduces an agent-based modeling framework for reproducing micro behavior in economic experiments. It gives an overview of the theoretical concept which forms the foundation of the framework as well as short descriptions of two exemplary models based on experimental data. The heterogeneous agents are endowed with a number of attributes like cooperativeness and employ more or less complex heuristics during their decision-making processes. The attributes help to distinguish between agents, and the heuristics distinguish between behavioral classes. Through this design, agents can be modeled to behave like real humans and their decision making is observable and traceable, features that are important when agent-based models are to be used in collaborative planning or participatory model-building processes.

INTRODUCTION

Modeling human behavior is challenging. Modelers of agent-based models face a choice and a trade-off: how simple and traceable or realistic and psychologically plausible should agent behavior be modeled. Theory does not provide much guidance in this respect since numerous and sometimes contradictory theories on human behavior exist. We chose to model agents as boundedly rational. We base their behavior on observation, both from experimental economics and field studies.

The agents in our models are characterized by a set of attributes, which have been derived from experimental data and theoretical ap-

proaches. They have expectations about their environment and the other agents. In this way, agent diversity is introduced. They exhibit boundedly rational behavior through aspiration levels and the use of simple heuristics.

This chapter introduces our modeling approach and the tool developed for our purposes which provides an environment for creating agent-based models with boundedly rational decision making. The focus lies on developing models based on economic experiments, but the tool is expandable to include field studies as well. Experimental economics provides us with a rich database of human behavior in simple, well-defined settings. The experimental results can be compared and reproduced. This allows extracting behavioral regularities from the data and using these to define micro behavior in the model. This is important because different micro behavior can lead to similar aggregate results. This chapter includes two exemplary models of economic experiments.

The software tool provides the agents with attributes and a set of heuristics that can be expanded to include new heuristics that fit better to problems which have not yet been modeled. The modeler can set parameters, like which kind of agents use which kind of heuristics. He or she can include new heuristics and learning processes, as well as new decision environments. With this tool it is possible to compare different micro behavior as well as reactions on different model framings. We expect this modeling approach to make it possible for stakeholders to identify with agent behavior and thus facilitate group model building and collaborative planning.

The remainder of this chapter is organized as follows. The section "Bounded Rationality" is a brief introduction to this theory of human behavior and its advantages for our purposes. In the section "Attributes and Heuristics," our modeling approach is described. In the section "Exemplary Models," two different models are described that reproduce data from economic experiments. These are the voluntary contribution mechanism with and without punishment, and appropriation games with and without communication. Some concluding remarks are made about the relevance of the modeling approach presented in this chapter. The appendix provides some technical details for modelers who want to use the tool.

BOUNDED RATIONALITY

The decision environments considered here are explicitly those in which the classical economic model of decision making fails to make correct predictions of actual human behavior. With classical model we refer to subjectively expected utility maximization of perfectly rational actors. The situations we are interested in include social dilemmas in which individual rationality differs from group rationality, and gift exchange situations in which gift giving seems to follow norms of reciprocity or fairness. At least in these situations the classical economic model of perfect rationality has to be enriched or complemented by theories that explain instances of cooperation and reciprocity.

Bounded rationality was introduced as an alternative to perfect rationality as the principle of human decision making (Selten, 1990). Instead of grounding a behavioral theory in optimization processes with a number of rather unrealistic assumptions on the capabilities and preferences of humans, bounded rationality is psychologically plausible (Gigerenzer & Todd, 1999; Gigerenzer & Selten, 2001).

The main idea is that learned heuristics are the basis of human behavior instead of permanent optimization. Heuristics are simple behavioral patterns that are triggered by the decision

context or environment. The usefulness of different heuristics in different circumstances is learned by experience. Heuristics are often simple and use little information, or better yet, they exploit the informational structure of the environment. This does not mean that humans are incapable of planning and full rational decision making. But the case of full rationality, trying to integrate all aspects into the decision-making process and finding the optimal solution, is a limiting case (Ostrom, 2004, p. 39). Apart from its unrealistic assumptions on the cognitive capacity of humans, optimization does not lead to satisfactory outcomes in many decision situations where uncertainties are high and/or fast reactions are required despite limited information. Optimization is applied only in situations where stakes are very high, the environment is competitive, there is much time for the decision, and information is abundant. Bounded rationality is a theory for day-to-day decision making, like buying toothpaste, as well as important high-cost situations, like buying a house. Decision contexts differ greatly, and so do heuristics needed in these situations.

One aspect of bounded rationality is the aspiration adaptation theory (Selten, 2001). According to this theory, humans have an aspiration level for a decision outcome. Above this level, an outcome is perceived as satisfactory, and below it is unsatisfactory. If a decision based on one heuristic leads to an unsatisfactory outcome and the decision can or has to be repeated, a search process for other heuristics or other choices is triggered. Another heuristic promising a better outcome is used, if it is found. If no heuristic is found that would lead to a satisfactory outcome, the aspiration level may be adapted. It can also be adapted upwards after a number of satisfactory outcomes. However, if a decision leads to a satisfactory outcome, it is repeated without any consideration of optimality. Instead of optimizing, boundedly rational humans are *satisficing*.

Whether or not the possible decisions or heuristics are actually carried out or made in thought depends on the situation. If you want to buy a house, you cannot try all possible decision alternatives. However, you can think about your aspiration level and whether or not a specific house is promising to be satisfactory. If, on the other hand, you buy toothpaste, you can probably not decide whether or not the toothpaste meets your aspiration of taste or health, but you can try another one next time. These examples refer to the decisions rather than the heuristics involved for choosing. But think again of toothpastes. A heuristic could be: Buy toothpaste with stripes. Another one could be: Buy toothpaste with green wrapping. If the third different toothpaste with green wrapping and stripes does not meet your requirements for taste or health, you can deliberately change one of the heuristics mentioned above. Maybe a friend recommended a particular brand of toothpaste and you try it, maybe you try *buy the most expensive*. Processes like these are modeled in the examples presented below.

In comparison, the standard economic model predicts a person who wants to buy a house to assign values to all possible aspects of a house and calculate utilities of the houses in question by assigning weights to the aspects and making a linear combination of values and weights. The house with the highest utility is bought. The search for a house goes on until the costs of further search equals the benefits, both of which have to be calculated in advance. The same process has to be made for toothpastes. Since such an economic behavior is rather obviously inappropriate, economists claim that humans behave *as if* they were optimizing. This is the difference between substantive and procedural rationality, first pointed out by Simon (Rubinstein, 1998). Utility maximization may lead to substantively rational behavior, like optimal outcomes, but it is not a good description of how decisions are made by actual human

beings. As a process of decision making, utility maximization is not rational. Bounded rationality seeks to alleviate this problem.

Norms and emotions can function as stopping rules for the search process. A famous example for an emotion is love as the stopping rule for the search for a mate. Somewhere out there may be the optimal mate. You never know when or if you meet him or her. If you find a satisfactory mate, love tells you to stop looking (Gigerenzer, 2001). An equally famous example for a norm is the reciprocity norm. In a private or business relationship, this norm tells you to treat the other as he or she treats you. Instead of calculating the probability for future dealings with this person and possible positive outcomes, comparing it with a short-term benefit that could be achieved by cheating now, you simply reciprocate trust with trustworthy behavior, nice acts with nice acts, and retaliate cheating with mistrust and an end to the relationship. How nice or hostile to reciprocate can be subject to a learning process. Ostrom and colleagues report that retaliation is not usually done as a grim trigger but rather as measured reaction. "Defections trigger mild reactions instead of harsh punishment. If defections continue over time, the measured response slowly moves from the point of agreement toward the Nash equilibrium" (Ostrom, Gardner, & Walker, 1994, pp. 199-200).

ATTRIBUTES AND HEURISTICS

People's behavior is diverse. In part, these differences may be explained by the history of events. We also assume people to have different dispositions to behave one way or another. These dispositions can be altered by experiences. In our models, however, the dispositions, modeled as attributes, are not changed, because of the short time horizon of the games. Expectations on the other hand are changed quickly.

Our attribute set consists of seven attributes. They are modeled as values between 0 and 1, equally distributed and independent from each other. However, if distributions or dependencies are subject of an exploration, this default can easily be changed. Our seven attributes are introduced next, and examples are given of their use in economic games. Then we introduce our heuristics approach. This can be seen as general agent behavior, which is defined and refined in specific decision environments. Examples for decision environments modeled with this approach are given in the next section.

Attributes

- **Cooperativeness:** Defines the importance of group utility compared to individual utility. A high cooperativeness implies a willingness to invest individual resources to increase the outcome of a group, even if the individual outcome may or will be less than without the investment. This attribute corresponds to cooperation as "joint gain maximization" in the social value orientation theory (McClintock, 1972) and to "impure altruism" (Andreoni, 1989). This is the main attribute deciding on how much an agent is willing to invest in the voluntary contribution mechanism in first games (see next section).
- **Conformity:** Defines norm compliance and the willingness to follow rules defined by the group. A high degree of anonymity lessens the importance of this attribute, while monitoring increases it even without sanctioning mechanisms (Burnham, 2003; Hoffman, McCabe, & Smith, 1996). This attribute is used to implement compliance to an agreed upon investment level in

appropriation games with communication (see next section).

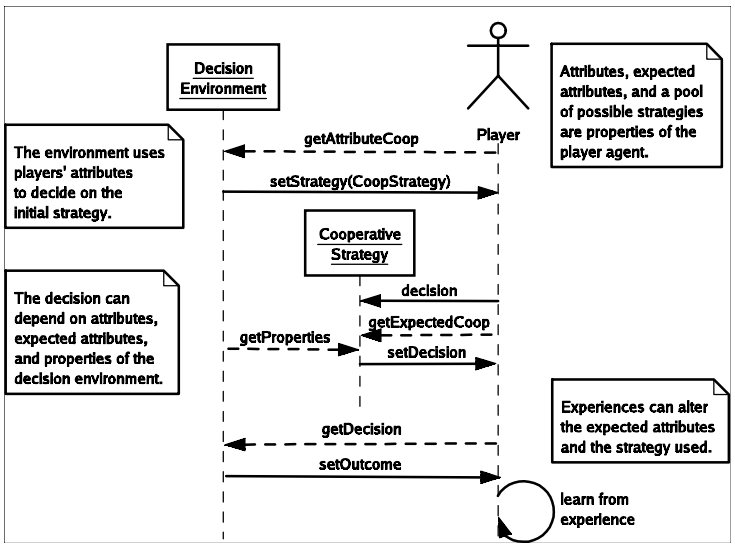
- **Fairness Concerning Others and Fairness Concerning Me:** Fairness is modeled as pure inequity aversion. Unequal outcomes are treated differently whether they are in favor of the agent or in favor of other agents. Agents with a high "fairness concerning others" are willing to sacrifice part of their resources to increase another agent's outcome that has fewer resources, even without any efficiency increase. Agents with a high "fairness concerning me," on the other hand, are willing to sacrifice resources in order to decrease another agent's outcome that has more. This differentiation follows Fehr and Schmidt (1999). "Fairness concerning others" defines how much a proposer in a dictator game[1] gives. "Fairness concerning me" defines what allocation is perceived as fair by responders in ultimatum games.[2]
- **Positive Reciprocity and Negative Reciprocity:** An agent with a high positive reciprocity is willing to reward actions by other agents perceived as friendly. An agent with a high negative reciprocity, on the other hand, is willing to punish an act perceived as hostile. With these two attributes, we model decisions based on perceived intentions as in Cox (2004). The differentiation into positive and negative reciprocity follows McCabe and Smith (2001). Positive and negative reciprocity are used by second movers in gift exchange[3] games and ultimatum games. Negative reciprocity is used for punishment decisions (see next section), positive reciprocity for rewards.
- **Risk Disposition:** In general, agents are risk averse in choices with sure gains and risk seeking in choices with sure losses (Kahneman & Tversky, 1979). This attribute is used in lottery games.

Agents also expect other agents to have attribute values. These are called expected attributes and are also used in the decision-making process. If, for instance, an agent has a high expected positive reciprocity, that implies it expects others to reciprocate nice acts, it will give more in a gift exchange game and is more willing to comply with agreements. In this way, trust is modeled, which one may have missed in the above list. High expected conformity is trust in the other's compliance to a specific norm or agreement (see next section), high expected cooperativeness is trust in the other's willingness to cooperate, and high expected positive reciprocity is trust that a gift will be returned. This reflects several theoretical concepts of trust (Nooteboom & Six, 2003, p. 4; Cook & Cooper, 2004, p. 219; Cox, 2004, p. 263). Per default, the initial values of the expected attributes are the same as the attribute values of an agent, but the expectations change within an experiment or a model, while the attributes themselves do not. Reputation formation is modeled as learning expected attributes for specific other players.

Heuristics

This subsection introduces our heuristics approach, based on the theory of bounded rationality presented above. Heuristics are used as decision mechanisms, and as search and stopping rules in an aspiration adaptation process. Heuristics usually depend on the decision environment. Some also depend on an attribute of the agent (give maximum times value of cooperativeness), some on the range of possible outcomes (give half). For some search rules, possible decision heuristics need to be ordered (try a nicer heuristic, try a more selfish heuris-

Figure 1. The interplay between an agent and a decision environment uses attributes and heuristics, which are properties of an agent. First, attribute values are used to decide on which heuristic to take, and then the heuristic is employed. It uses attributes, expected attributes, and possibly properties of the decision environment to make the decision. What an agent can learn from an experience also depends on the decision environment.



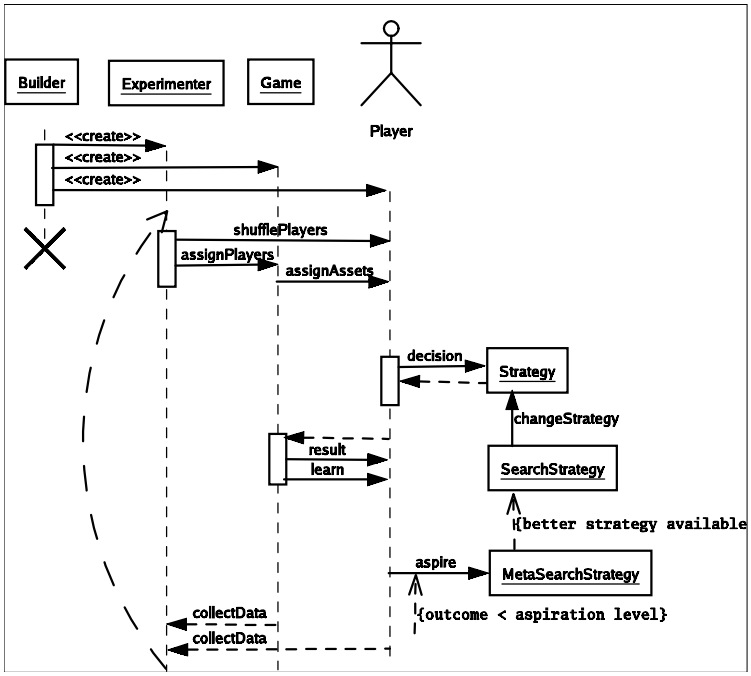
tic), some depend on the range of possible decisions (give one point more or less). Some implement a kind of social learning (when others give nothing, give nothing, too). Stopping rules refer to an aspiration level (if you are satisfied with the outcome, repeat the action). However, more extensive learning mechanisms, like case-based reasoning, can also be implemented. The role of attributes and heuristics in a decision-making process is depicted in Figure 1. In the examples presented below, the *decision environment* is a combination of experimenter and games. So far, the match between the decision environment and possible heuristics has to be made by modelers.

In this chapter we use the term strategies for simple heuristics and decision processes depending on more extensive calculations. The distinction is not clear. *Give half* is a simple heuristic, although a calculation takes place. *Calculate optimal outcome* on the other hand may involve sophisticated mathematical procedures. Our agents are in principle able to do both. So, we named the superclass of all decision processes *Strategy* rather than Heuristic.

EXEMPLARY MODELS

This section provides examples for models implemented with our approach. First, a brief description of the general framework is given. Then, two games are discussed in more detail, the voluntary contribution mechanism with punishment and appropriation games with communication effects. For further information and testing, please refer to this tool's Web site at http://www.usf.uos.de/~eebenhoe/forschung/adaptivetoolbox. There you can make model runs, alter parameters, and view the source code. The models are programmed in JAVA using the modeling environment FAMOJA (http:/

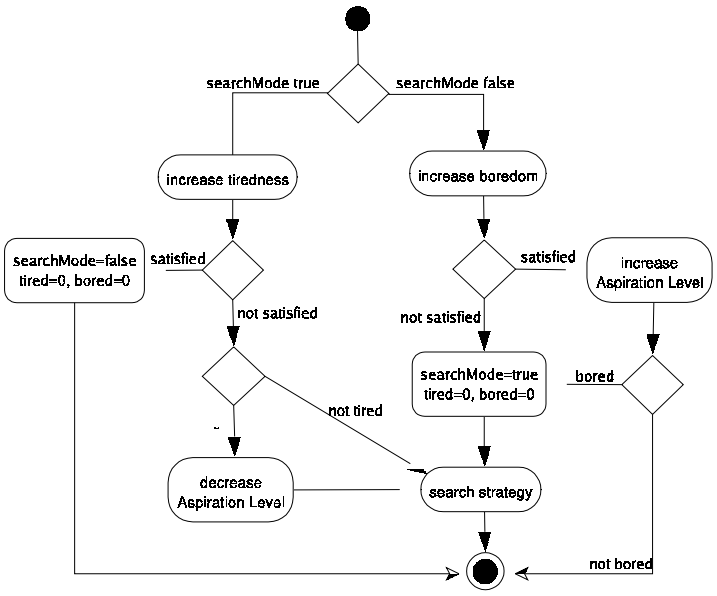
Figure 2. An experiment is structured into several phases. In the initial phase, the builder organizes the creation and setup process. In the second phase, the experimenter organizes players and games. Then players make their decisions, followed by games calculating the outcome. The last phase consists of data collection and evaluation. Players may change their strategy. Except for the creation and setup process, the phases are repeated to model successive games.



/www.famoja.net). The purpose of this section is to provide an understanding of our modeling approach and the way in which we model economic experiments. Also, the tool is described along the way. These brief descriptions will not enable you to reproduce the models. For complete descriptions including all parameters and model settings, please refer to the model Web sites.

The general model consists of an experiment builder which creates the experimenter, the games, and the player agents. The builder also sets all parameters. So, in order to change parameter values, the builder needs to know the parameter and be able to change it in the experimenter, games, or player agents. The advantage of this design is that only one class is concerned with the creation process, and all variables are bundled together. One instance of this builder class is created in order to start a model run. The experimenter organizes the experiment during the model run, it shuffles the player agents, assigns them to the games, and sets their decision strategy pool, search strategies, and meta-search strategies. The experimenter also collects data and has a number of chart objects to display data during the model run. Which data to display in which kind of charts is also defined in the builder class. The game defines the actual rules. It also sets the player agents' assets each round, calculates the outcome according to their decisions, and informs them about their outcomes, the other player agents' decisions, and the other player

Figure 3. UML diagram of the aspiration adaptation process



agents' outcomes. The player agents have a decision strategy, which makes the decision. If the aspiration level is not met, the player agent can change the decision strategy according to its meta search strategy and search strategies. The scheduling of selected events during a model run is displayed in Figure 2.

As indicated in Figure 2, player agents have an aspiration level. Whether or not the aspiration is met influences the search in the decision realm. In addition to the aspiration level, modelers can define tiredness and boredom of player agents. If an aspiration has not been met for a time, a player agent can decrease its aspiration level. If, on the other hand, the aspiration has been met for several rounds, not only is the aspiration level increased, a different strategy may be tried (see Figure 3).

Voluntary Contribution Mechanism With and Without Punishment

In the voluntary contribution mechanism, a number of players are assigned initial assets, which can be invested into a common project or kept to oneself. The total investment into the common project is increased by the experimenter and the result spread evenly among all players, regardless of their contribution. This increase constitutes a potential profit. However, investment is risky because a player does not know the other players' contributions and may receive less than he or she invested. Group optimum is reached if each player contributes all of his or her assets. There is, however, an incentive to free-ride—that is, not to contribute and profit from the others' contributions. A number of experiments have been conducted using the voluntary contribution mechanism. See Holt (1999) for a survey.

In some experiments the voluntary contribution mechanism is used as a baseline and is extended by a punishment possibility. Subjects still make their decisions anonymously, but can invest assets to punish other subjects for previous decisions. The punished subjects then have to pay more. This mechanism usually leads to an increase in the investment level and to higher

returns. See Ostrom (2004, p. 36ff) on punishment in public goods experiments. Studies incorporating the punishment mechanism in addition to the voluntary contribution mechanism include Nikiforakis (2003), and Fehr and Gächter (2000, 2002). From the latter we used the individual data to design and test the model presented here.[4] Five sessions of the following design were conducted by Fehr and Gächter. Six parallel games were played with four subjects each. Subjects do not meet another subject twice, in order to prevent reputation formation. Each subject was endowed with 20 money units, and total investment is increased by the experimenters by 60%. After six games without punishment possibilities, a sanctioning mechanism is introduced and repeated for further six games. After each voluntary contribution mechanism, subjects may pay a fee of up to 10 money units in order to punish other players, who have to pay a fine that is three times the fee. An earlier version of our model of this experiment is discussed in more detail in Pahl-Wostl and Ebenhöh (2004).

Non-cooperative game theory predicts zero investment and zero punishment regardless of the treatment (Isaac, Walker, & Thomas, 1984; Fehr & Gächter, 2000). However, in experiments of voluntary contribution mechanisms without punishment, usually overall investment starts at surprisingly high 50% of maximum and then decreases. With punishment, investment is immediately at a higher level and increases further over the course of several games. Individual investment decisions are very divergent. In the individual data of the experiment by Fehr and Gächter (2002), there is evidence for some subjects, who give the maximum of 20 money units over the course of all 12 games. There are also some who give 0 in all 6 games without punishment and only 3-5 money units in games with punishment. There are some who jump from high investment to very low investment and back, and still others who change their investment only by small amounts. For the model, we extracted the following behavioral regularities that became the heuristics in the model:

1. **With a low cooperativeness (<0.21), contribute nothing:** This is called MaximizingStrategy and is used by player agents not willing to cooperate. Actually this strategy calculates optimal outcome under the player agent's expectations. In the voluntary contribution mechanism without punishment this results to 0, but with punishment the investment is the lowest value that is expected not to provoke punishment.
2. **With a medium cooperativeness, contribute maximum times expected cooperativeness:** This heuristic is called ReciprocalExpectationStrategy and is used by player agents willing to cooperate if they expect the others to cooperate as well.
3. **With a high cooperativeness (>0.68), contribute maximum to common project if expected cooperativeness exceeds three-quarters. Contribute three-quarters if expected cooperativeness is less to indicate willingness to cooperate:** This heuristic is called CooperativeStrategy and is used by player agents willing to initialize cooperation.

High and low are parameters by which to fit the model to a specific experiment. The values given in brackets are used in order to model the experiment of Fehr and Gächter (2002). Expected cooperativeness is adjusted quickly by the individual agents to the experiences made in earlier games. This leads to changes in the decision made by the heuristics 2 and 3 without

a change of the heuristic employed. Our three strategies are not universal, even within experiments with similar designs. They do not, for instance, incorporate dependency of free-riding behavior on group size and marginal per capita return, as reported in Isaac et al. (1984). Leaving the three strategies intact, these effects can be incorporated in the model as different initial levels of cooperation, which is actually reflected in the first games of the above mentioned experiment. If initial cooperation is above 50%, expectations are usually exceeded and therefore cooperation rises; if it is below 50%, actual cooperativeness falls short of expectations and therefore cooperation decreases, mainly because of the reciprocal heuristic.

If the outcome of a game does not satisfy the agent's aspiration level, it searches for another strategy. In this case, only the three strategies mentioned above are in the search realm. Initial aspiration levels are set by the model experimenter. If they are set to very low levels, no strategy changes occur. If the level is set too high, a stable distribution of strategies is reached only after aspiration levels are reduced to realistic levels.

Changes in heuristics have been implemented using cues evaluated by a meta search strategy. Note that these cues are used only when the aspiration is not satisfied.

Cues for a more cooperative strategy:

- cooperation is higher than expected
- if one (two, three) other(s) give three-quarters or more

Cues for a more selfish strategy:

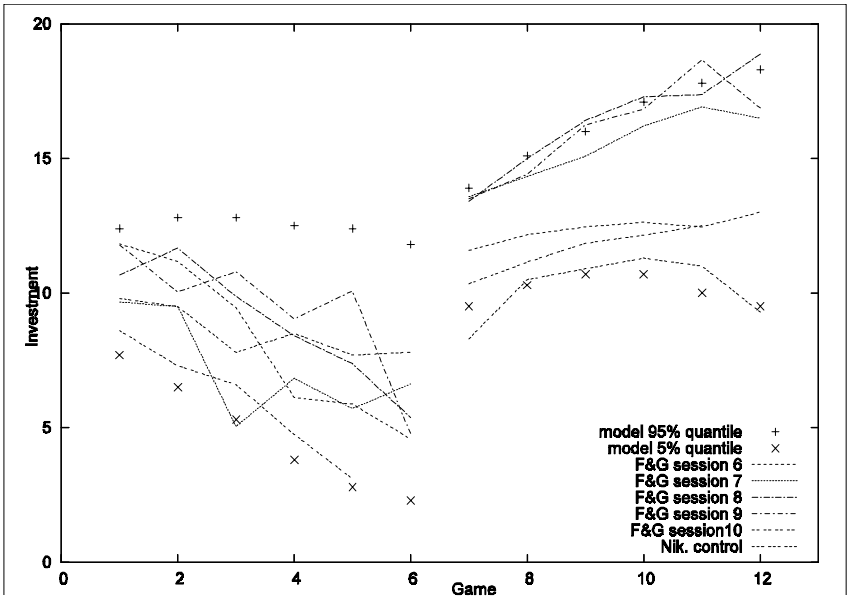
- cooperation falls short of expectations
- one (two, three) other(s) give zero
- outcome is less than investment (counts for two cues)

Different kinds of agents (with different attributes) may be biased into one direction or the other. A cooperative player agent needs more cues indicating non-cooperation than a less cooperative player agent. For a cooperative player agent (cooperativeness>0.68), three more cues for defection than for cooperation have to be encountered in order to switch to a more selfish strategy (first from cooperative strategy to reciprocal, then from reciprocal to maximizing), and one more cue for cooperation than for defection is enough to move towards cooperation. All other player agents need one more of either cue in order to change their behavior.

In addition, in punishment games, the actual punishment is of course another *cue* for switching towards more cooperative behavior. However, if punishment is taken into account, MaximizingStrategy can also lead to high investment. Except for this reasoning, the strategies do not change in the punishment treatment. In order to reproduce the data, switching from no-punishment to punishment treatment (or vice versa) leads to higher (lower) expectations of others' investment levels. ReciprocalExpectationStrategy can lead to high investment levels because of raised expectations. This leads us to an interesting result. Without punishment, the stable investment level of zero money units can be reached with two different strategies (1 and 2) if aspiration levels are not too high. Not all agents use MaximizingStrategy. Some stick to ReciprocalExpectationStrategy and can therefore adjust very quickly to changes. With punishment, all three strategies can survive the adaptation process.

The results of the experiment by Fehr and Gächter (2002) are shown in Figure 4. They are compared to model results of a model run with 100 parallel models of six parallel games each.

Figure 4. Mean investments of a model run with 100 parallel games compared to the results of the experiment by Fehr and Gächter (2002) and Nikiforakis (2004). Treatment is first six games voluntary contribution mechanism, and then six games with punishment opportunity.



The dots indicate the 5%-quantile and 95%-quantile. The lines show data from experimental sessions by Fehr and Gächter (2002) and a control experiment by Nikiforakis (2004). All sessions except for session 10 are conducted with six parallel games in the strangers' treatment; first six games without punishment opportunity and then six games with punishment opportunity. Session 10 was conducted with only five parallel games and five successive games each. The model variance seems to be close to the variance of the experiments. The mean of 100 model runs compared to the mean of the experimental results by Fehr and Gächter (2002) yields a χ^2 of 2.12 which does not allow rejecting the hypothesis of a common distribution. However, this was the data used to calibrate the model. The data from Nikiforakis (2004) is reproduced less well with a χ^2 of 13.86, due to the games with punishment. However, this was only one session, and the results of the games with punishment opportunity are quite different from the mean of the sessions of Fehr and Gächter (2002).

Validation of agent-based models should also comprise a qualitative validation of the micro behavior (Moss & Edmonds, 2005). Two possibilities to do so in this case are expert knowledge and questionnaires: Do experimenters of these models judge the agent behavior to be a valid representation of subject behavior? Do questionnaires filled in by subjects indicate the usage of heuristics similar to those described here? In this case, the description of subject behavior from Fehr and Gächter (2002) based on post-experimental questionnaires was used in addition to the individual data sets of that experiment, but no ex post validation has been made.

Appropriation With and Without Communication

The second example is a model of appropriation experiments by Ostrom et al. (1994, Chapters 5 and 7).[5] The setting is an eight-person game with two markets. Market 1 is an outside opportunity with a constant and fixed return per invested token. Market 2 on the other hand is a common pool resource with a negative quadratic function, where the return per token depends on the total number of tokens invested. In the parameterization implemented here, the function is:

$$y = 23 * \Sigma x_i - 0.25 * (\Sigma x_i)^2$$

where x_i is the number of tokens invested by player i in this round and y is the total return of market 2, which is divided proportionally among the players who have invested in market 2. The actual payment is \$0.01 per token return of market 2, and \$0.05 for each token invested in market 1. The players were endowed every round with 25 tokens each. There have also been experiments with 10-token endowment which are not discussed here.

From a theoretical viewpoint, there are three important points in this setting. Group optimum is a total investment in market 2 of 36 tokens, 4.5 tokens per subject. Zero rent is at 72 tokens total investment in market 2, 9 tokens per subject. The Nash strategy would be to invest 8 tokens per subject, because the ninth token would not yield more in market 2 than in market 1 (assuming all others also invest one more token). For a theoretical discussion, see Ostrom et al. (1994, pp. 109-115).

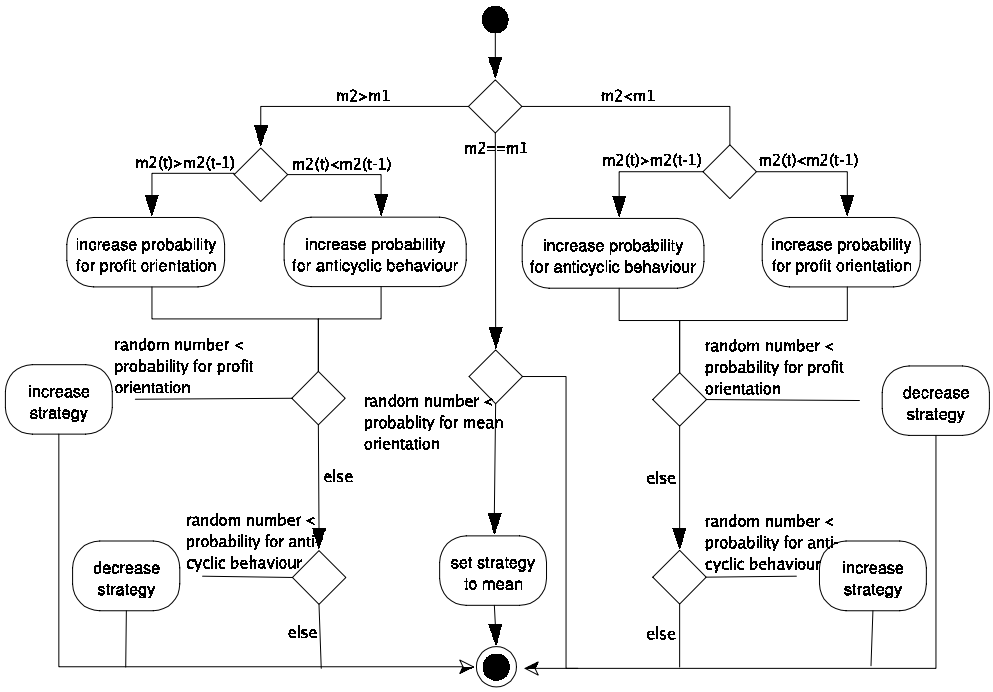
These experiments have been conducted with varying settings including probabilistic deterioration of the common pool resource, different sanctioning mechanisms, and communication. Here, only the baseline experiment and one-shot communication are presented. Usually, the game was repeated 20 to 30 times.

A major result of the baseline experiment is a pulsing pattern of the investment level, which tends to increase until token returns of market 2 fall below token returns of market 1, then decreases again. There was no symmetry in this pattern across experiments. However, the variance usually decreased over the course of several games, but there was no stability. Furthermore, individual investment never followed a pure Nash strategy. Overall, mean investment levels were rather close to the Nash equilibrium and far from the optimum. For a discussion on the results, see Ostrom et al. (1994, pp. 115-120, 151-153).

Looking at individual investment decisions reveals four different decision patterns. The most obvious pattern is to invest more (less) in market 2 next round, when token returns have been higher (lower) in market 2 than in market 1 this round, and vice versa. In the model this is called the profit-oriented strategy. However, assuming other player agents react, profit-oriented leads to anti-cyclic investment behavior: if return from market 2 was low, we can assume others invest less in market 2 and invest more ourselves. This can be considered as defection. Investing less when profit from market 2 was high in expectation that others will increase investment can be seen as cooperation, or merely as safe play. Some subjects seem to analyze the investment trend: if investment is increasing over two rounds, they drop investment, and if it is decreasing, they increase it. The fourth pattern is orientation on the mean of the previous round.

All four patterns have been implemented and can in principle be used by each player agent. These patterns are implemented as meta search strategies. However, depending on the attributes, some agents have a greater affinity to profit-oriented, nice anti-cyclic, defecting

Figure 5. UML activity diagram of the decision search process in appropriation games without communication



anti-cyclic, trend-oriented, or mean-oriented behavior. This is modeled by making the probabilities for the different behaviors depend on the agents' attributes. The general decision search process is depicted in Figure 5. For the exact implementation and the parameter configuration, please refer to the Web site of this model.

The actual decision heuristics are simply the possible integer values of investment decisions in market 2. That is, in the 25 token design, there are 26 decision heuristics, 0 to 25. Search heuristics are modeled as an increase or decrease of the investment decision by 1, 2, 5, 10, or 15 tokens. Whether or not to increase or decrease is decided by the meta search heuristic. As before, strategy search is only necessary if the aspiration level is not met. Aspiration levels are set to levels between $1.20 and $1.70. Also as before, initial strategies are set by the model experimenter, according to agents' cooperativeness, to 5, 10, 15, or 25. The agent's preferred strategy (choice) is also set to 5, 10, 15, or 25 following a different rule:

- Initial strategy:
  - 5 with cooperativeness>0.63
  - 10 with 0.42<cooperativeness<0.63
  - 15 with 0.17<cooperativeness<042
  - 25 with cooperativeness<0.17
- Choice strategy:
  - 5 with cooperativeness>0.5 or fairness concerning me<0.5
  - else equal to initial strategy

Communication effects have also been implemented. Communication in the model has two purposes. First, agents broadcast their choice strategy and indicate their willingness to follow the strategy mentioned most often. Second, expected conformity and cooperativeness

Figure 6. Scheduling of events during the communication process

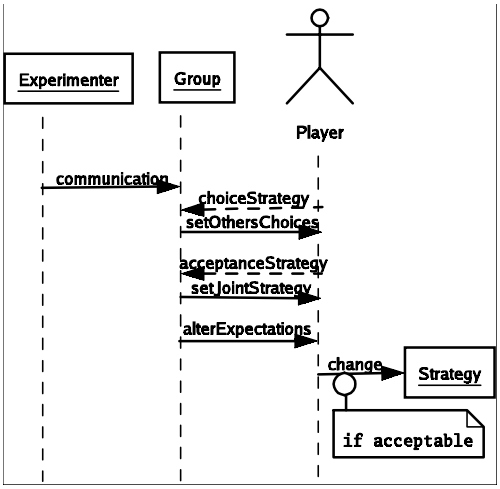
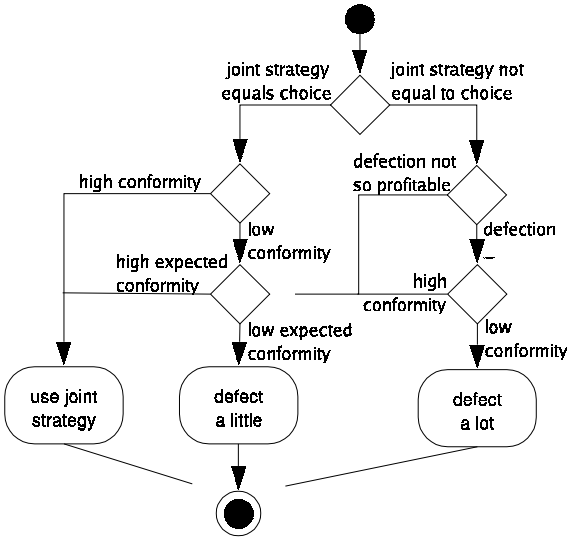


Figure 7. UML diagram of the decision whether or not to comply with a joint strategy



is altered (see Figure 6). Ostrom (2004, pp. 33-34) summarizes findings on communication in public goods experiments and stresses that information about optimal strategies is indeed discussed, but not the primary source of the observed increase in cooperation after communication. Instead, an increase in trust seems to be the most important aspect of face-to-face communication. After face-to-face communications, subjects could predict others' cooperation levels significantly better than without the chance to see each other (Ostrom, 2004, p. 51). In our model this is reflected by setting the expected cooperativeness as well as the expected conformity of player agents to the mean of the previously expected values and the actual values of the other player agents. In addition, the player agents' strategy is set to the joint strategy, which is the strategy mentioned most often, if they are willing to follow it. This willingness is determined by their choice strategy, their conformity, and their expected conformity (see Figure 7).

Fitting the model to a specific experiment is, of course, possible. However, the experiments reported by Ostrom et al. (1994) differ in actual investment levels. The qualitative result is that experiments without communication usually start with over appropriation; investment levels then follow a pulsing pattern with a tendency for the variance to decrease. This is replicated in our model (see Figure 8). Communication tends to increase returns, but not always to the maximum possible. In one-shot communication experiments, cooperation breaks down more often than not after some rounds. This too is replicated in the model. The model results, however, depend strongly on group composition.

In order to validate this model, relative changes in total investment from one round to the next were compared statistically (see Figure 9). Five values from the three baseline experiments fall outside the 5%- and 95%-quantiles, as should be the case. However, the model does not reproduce the apparent effect of decreasing variability in later games compared to earlier games. In order to cross-

Figure 8. Result of a specific model run without communication and with a one-shot communication after round 10

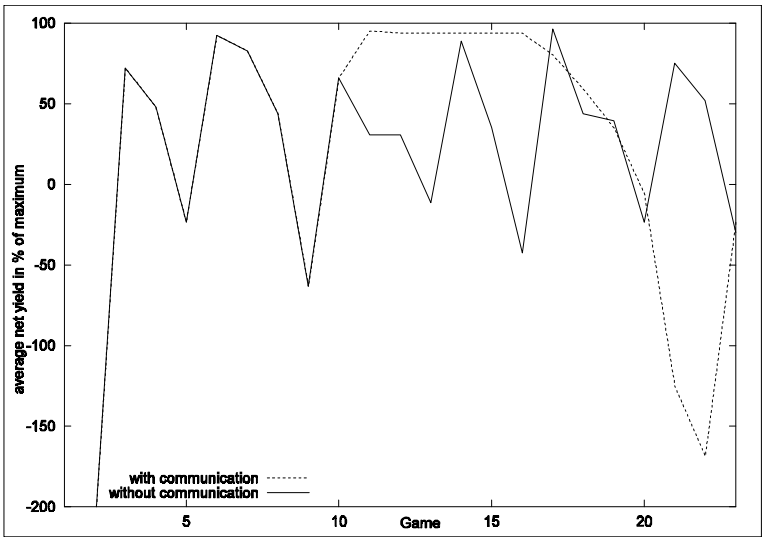
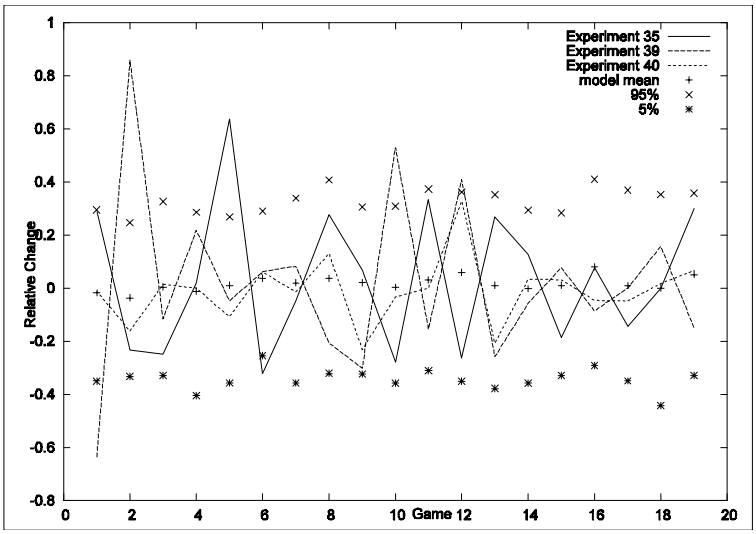


Figure 9. Relative changes in total investment of the three baseline model runs compared with model data from 100 parallel games



validate the model, the micro behavior must also be validated (Moss & Edmonds, 2005). Aside from data analysis which was the basis for the model micro behavior, questionnaires can be used. In post-experimental questionnaires, the most common heuristic mentioned was indeed the profit-oriented heuristic used here: increase investment, if market 2 yields a

higher token return than market 1, and decrease investment otherwise (Ostrom et al., 1994, p. 121). Again, no ex post validation has been done.

CONCLUSION

In this chapter a modeling approach is presented, which bases agent behavior on experimental data in order to capture micro behavior properly. Two models of economic experiments are presented in which the aggregate results are reproduced, and the agent behavior follows behavioral regularities extracted directly from individual data. The models are presented here not in detail and serve the purpose of illustrating our general modeling approach. The models are part of a framework which allows for extensions and variations of model framings in order to capture institutional changes, like punishment and communication.

There are two main reasons for developing models like the ones presented in this chapter. The first is a technical reason. Object-oriented programming combined with agent-based modeling can help us develop models faster and with a greater variety. Differences in experimental settings can be implemented very quickly. When modeling something new, we can focus our attention on the essential aspects. So far, however, the framework exists only for models based on economic experiments. It is currently extended to include case studies with similar problems, but with a deeper social context. The modeling framework allows testing the plausibility of the assumptions derived from experimental settings for understanding real-world situations in field studies. A concrete model of a specific experiment or field study has to be carefully validated both on the aggregate level and the individual level. However, a framework that facilitates agent-based modeling can help with group model building and designing role-playing games.

This leads to the second reason. In order to understand and adequately model human behavior in real social dilemmas, we need to understand their motives for cooperative, reciprocal, or punishing behavior. Under what circumstances can mutual cooperation be sustained? When do people trust one another enough to initiate cooperation? What institutions (communication, sanctioning mechanisms) can replace the role of or increase mutual trust? These aspects are addressed by experimental economics and need to be incorporated in agent-based models. On the other hand, this can also lead to further experiments especially designed to survey data needed to improve the models.

This in turn leads to models that are more valid with respect to micro behavior. It is essential for management tools to capture human behavior in a realistic way, because management strategies developed for theoretically sound, but unrealistic human beings tend to fail. Our modeling approach is a step towards implementing realistic behavior or procedural rationality. It is, admittedly, a step at a very basic modeling level, the agent level. However, the modular nature of our model toolbox makes integration into larger models feasible.

REFERENCES

Andreoni, J. (1989). Giving with impure altruism: Applications to charity and Ricardian equivalence. *Journal of Political Economy, 97*(6), 1447-1458.

Burnham, T. C. (2003). Engineering altruism: A theoretical and experimental investigation of anonymity and gift giving. *Journal of Economic Behavior & Organization, 50,* 133-144.

Cook, K. S., & Cooper, R. M. (2004). Experimental studies of cooperation, trust, and social exchange. In E. Ostrom & J. Walker (Eds.), *Trust and reciprocity. Interdisciplinary lessons from experimental research* (pp. 209-244). New York: Russel Sage Foundation.

Cox, J. C. (2004). How to identify trust and reciprocity. *Games and Economic Behavior, 46,* 260-281.

Davidsson, P. (2002). Agent-based social simulation: A computer science view. *Journal of Artificial Societies and Social Simulation, 5*(1). Retrieved June 10, 2006, from http://jasss.soc.surrey.ac.uk/5/1/7.html/

Fehr, E., & Gächter, S. (2000). Cooperation and punishment in public goods experiments. *The American Economic Review, 90*(4), 980-994.

Fehr, E., & Gächter, S. (2002). Altruistic punishment in humans. *Nature, 415*(January), 137-140.

Fehr, E., & Schmidt, K. M. (1999). A theory of fairness, competition, and cooperation. *The Quarterly Journal of Economics,* (August), 817-868.

Gigerenzer, G. (2001). The adaptive toolbox. In G. Gigerenzer & R. Selten (Eds.), *Bounded rationality. The adaptive toolbox* (pp. 37-50). Cambridge, MA: The MIT Press.

Gigerenzer, G., & Selten, R. (2001). Rethinking rationality. In G. Gigerenzer & R. Selten (Eds.), *Bounded rationality. The adaptive toolbox* (pp. 1-12). Cambridge, MA: The MIT Press.

Gigerenzer, G., & Todd, P. M. (Eds.). (1999). *Simple heuristics that make us smart.* New York/Oxford: Oxford University Press.

Hare, M, & Deadman, P. (2004). Further towards a taxonomy of agent-based simulation models in environmental management. *Mathematics and Computers in Simulation, 64,* 25-40.

Hoffman, E., McCabe, K., & Smith, V.L. (1996). Social distance and other-regarding behavior in dictator games. *American Economic Review, 86,* 653-660.

Holt, C. A. (1999, December 29). *Y2K bibliography of experimental economics and social science: Voluntary contributions mechanism.* Retrieved June 27, 2005, from http://www.people.virginia.edu/~cah2k/vcmy2k.htm

Isaac, R. M., Walker, J. M., & Thomas, S. H. (1984). Divergent evidence on free riding: An experimental examination of possible explanations. *Public Choice, 43,* 113-149.

Kahneman, D., & Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica, 47*(2), 263-291.

McCabe, K. A., & Smith, V.L. (2001). Goodwill accounting and the process of exchange. In G. Gigerenzer & R. Selten (Eds.), *Bounded rationality. The adaptive toolbox* (pp. 319-340). Cambridge, MA: The MIT Press.

McClintock, C. G. (1972). Social motivation: A set of propositions. *Behavioral Science, 17*(5), 438-454.

Moss, S., & Edmonds, B. (2005). Sociology and simulation: Statistical and qualitative cross-validation. *American Journal of Sociology, 110*(4), 1095-1131.

Nikiforakis, N. (2004). *Punishment and counter-punishment in public goods games: Can we still govern ourselves?* Discussion Paper Series 2004/05, Royal Holloway University of London, UK. Retrieved June 10, 2006, from http://www.rhul.ac.uk/Economics/Research/WorkingPapers/pdf/dpe0405.pdf/

Nooteboom, B., & Six, F. (Eds.). (2003). *The trust process in organizations.* Cheltenham, UK: Edward Elgar.

Ostrom, E. (2004). Toward a behavioral theory linking trust, reciprocity, and reputation. In E. Ostrom & J. Walker (Eds.), *Trust and reciprocity. Interdisciplinary lessons from experimental research* (pp. 19-79). New York: Russel Sage Foundation.

Ostrom, E., Gardner, R., & Walker J. (1994). *Rules, games, & common-pool resources*. Ann Arbor, MI: The University of Michigan Press.

Pahl-Wostl, C., & Ebenhöh, E. (2004). An adaptive toolbox model: A pluralistic modelling approach for human behaviour based on observation. *Journal of Artificial Societies and Social Simulation, 7*(1). Retrieved June 10, 2006, from http://jasss.soc.surrey.ac.uk/7/1/3.html*l*

Rubinstein, A. (1998). *Modeling bounded rationality.* Cambridge, MA: The MIT Press.

Selten, R. (1990). Bounded rationality. *Journal of Institutional and Theoretical Economics, 146,* 649-658.

Selten R. (1998). Aspiration adaptation theory. *Journal of Mathematical Psychology, 42*(2-3), 191-214.

Selten, R. (2001). What is bounded rationality? In G. Gigerenzer & R. Selten (Eds.), *Bounded rationality. The adaptive toolbox* (pp. 13-36). Cambridge, MA: The MIT Press.

Tesfatsion, L. (2002, January 7). *Agent-based computational economics: Growing economies from the bottom up.* ISU Economics Working Paper No. 1. Retrieved June 10, 2006, from http://www.econ.iastate.edu/tesfatsi/

KEY TERMS

Agent-Based Modeling: Modeling refers to the process of designing a software representation of a real-world system or a small part of it with the purpose of replicating or simulating specific features of the modeled system. In an agent-based model, the model behavior results from behavior of many small software entities called agents. This technique is used to model real-world systems comprised of many decision-making entities with inhomogeneous preferences, knowledge, and decision-making processes. An advantage of this method is that no assumptions need to be made about an aggregate or mean behavior. Instead, this aggregation is made by the model. See Davidsson (2002) for a topology of different modeling techniques including what he calls agent-based social simulation, and see Tesfatsion (2002) for a discussion of agent-based computational economics. Hare and Deadman (2004) discuss different uses of agent-based models in environmental management.

Aspiration Adaptation Theory: Part of bounded rationality theory, the idea is that humans have an aspiration level. If a choice promises to satisfy this aspiration level, it is made without an extensive search for an optimal strategy. Selten coined the term *satisficing* instead of optimizing. If, however, after some search, no satisficing alternative is found, the aspiration level can be adapted downwards. Then a choice can be made among the alternatives already found that satisfy the new aspiration level (see Selten, 1998, 2001).

Bounded Rationality: A decision theory that rests on the assumptions that human cognitive capabilities are limited and that these limitations are adaptive with respect to the decision environments humans frequently encounter. Decision are thought to be made usu-

ally without elaborate calculations, but instead by using fast and frugal heuristics. These heuristics certainly have the advantage of speed and simplicity, but if they are well matched to a decision environment, they can even outperform maximizing calculations with respect to accuracy. The reason for this is that many decision environments are characterized by incomplete information and noise. The information we do have is usually structured in a specific way that clever heuristics can exploit (see Gigerenzer and Selten, 2001).

Experimental Economics: In Experimental Economics, behavior of human subjects is researched in controlled experiments with monetary incentives. The settings include simple games in which the subjects play with or against each other. Their decisions directly influence the payoffs they receive. By using stylized games in controlled situations, economic experiments produce comparable and reproducible data. Varying specific aspects in these experiments can help to understand which aspects of a decision situation influence human behavior in what way (see Kagel & Roth, 1995).

Heuristics: Simple decision-making processes that can be characterized as fast and frugal. In bounded rationality theory these heuristics are assumed to be adapted to certain decision environments. By exploiting the informational structure of the environment, heuristics can be both fast and accurate. An paradigmatic example is the *recognition heuristic*. It is applicable in decision environments in which the information and lack of information are structured according to a characteristic of the entities in question. If we are asked, for example, which English soccer team will win a match, and we have heard of one of the teams and not of the other, we tend to chose the one we know. And we tend to be correct with this choice (see Todd & Gigerenzer, 1999).

Public Goods and Common Pool Resources: Goods that have in common that it is difficult or impossible to exclude potential consumers from them. The difference between those two categories is the different degree of subtractability. The utility derived from public goods is not or only slightly diminished by others using the same good. Examples include coded law and fresh air. Common pool resources, on the other hand, are characterized by subtractability. Examples include the fish population in a lake and groundwater. There are goods that lie in between, for example infrastructure like highways: as long as its use is well below its capacity, one more car does not hinder the other cars; in rush hour, however, cars compete for space. While the main problem with Public Goods is the provision and corresponding free-rider behavior, the main issue with common pool resources is appropriation or over-appropriation. These problems are addressed by different experimental decision environments: voluntary contribution mechanism refers to the provision of public goods; appropriation experiments deal with (over-)appropriation of common pool resources (see Ostrom et al., 1994).

ENDNOTES

[1] In dictator games, one player—the proposer—is assigned the task of dividing an amount of money between him or herself and the other player. The second player can only accept the first player's decision. Since the players are usually anonymous, theoretically nothing prevents the first player from keeping the total amount.

[2] In ultimatum games, one player—the proposer—is assigned the task of dividing an amount of money between him or herself and the other player—the responder—who has veto power. That is, the second player can accept or reject the decision. If he or she accepts, the money is paid as allocated by the first player. If the responder rejects, both players get nothing.

[3] There is a great variety of gift exchange games which have in common that one player decides to give an amount of money to the other player, who also has to decide on an amount of money to give to the first player. The exchange may be simultaneous or the second gift is viewed as a return for the first gift. Usually the experimenter increases the gifts or the first gift

[4] We thank Mr. Fehr and Mr. Gächter for providing us with the data.

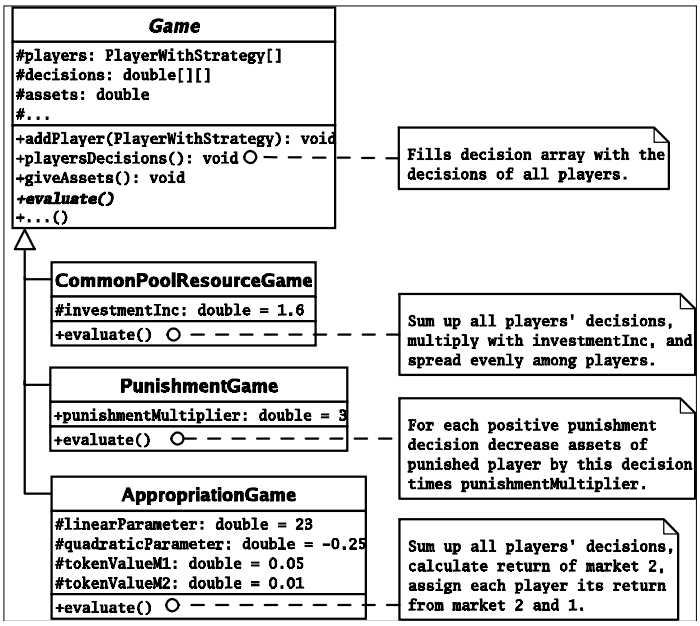
[5] We thank Mr. Walker for providing us with the data.

APPENDIX A

What Had to be Done to Come from Example 1 to Example 2?

First of all, the player agents are the same in both examples. The main changes from one model to another lies in the different games and consequently the different heuristics employed. Thus, the game has to be implemented. In the first example the game was first the voluntary contribution mechanism implemented as a common-pool resource game, and secondly the same common-pool resource game alternated with a punishment game. The game in the second example is an appropriation game. All three games are subclasses of an abstract superclass Game implementing different methods to evaluate the outcome (see Figure 10). Common pool resource game and appropriation game give assets to the player agents, punish-

Figure 10. UML diagram of Game class and subclasses described in the examples

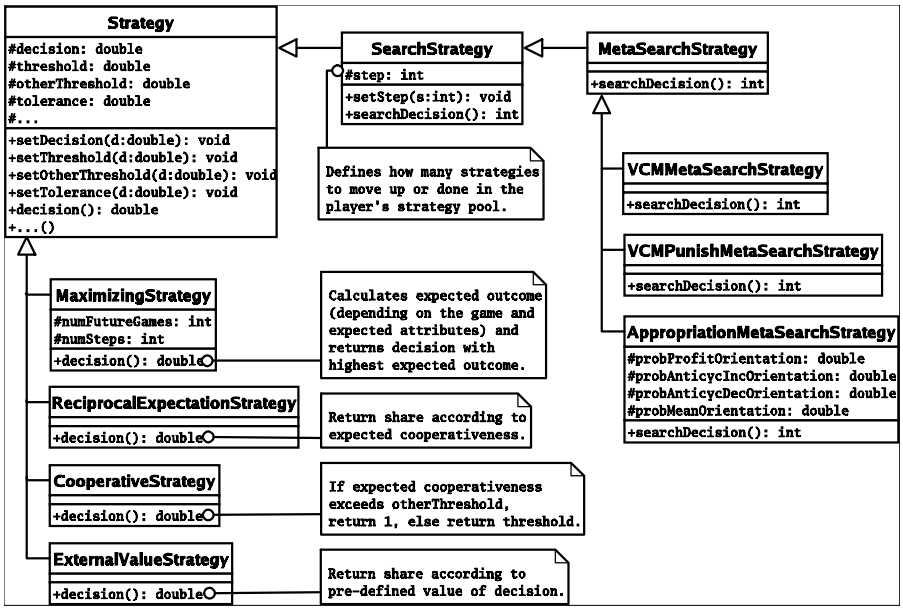


ment game does not. This can easily be achieved by setting the assets in punishment games to 0.

The strategies in the second example are conceptually different from those in the first example. However, all strategies (including the not discussed punishment strategy) are subclasses of a simple strategy, implementing different methods for the decision making (see Figure 11). While the strategies from the first example do actual calculations, the strategies in the second example return only a predefined number. Note that the search strategies in both examples are the same (increase/decrease in the strategies realm). The meta-search, however, depends strongly on the experiment.

Also, experimenters are different. While in the first example a standard experimenter is used, in the second example the experimenter also organizes the communication process. The last great difference is the builder. For many experiments with only slight variations, only the builder needs to be changed slightly. Many variations can even be modeled with different parameter settings, which can be changed without changing code. For example, strangers' and partners' treatments in the voluntary contribution mechanism can be modeled by setting the shuffle parameter to true (strangers) or false (partners). There may, of course, be other necessary changes, for example in the heuristics in order to reproduce reputation formation. In our case, the builders are again subclasses of a common abstract superclass ExperimentBuilder. All model parameters are set in the builders, as well as which and how many games to create, and which strategies to give to the player agents.

Figure 11. UML diagram of the strategy classes used in the examples



Chapter XVIII
Heterogeneous Learning Using Genetic Algorithms

Thomas Vallée
LEN, Nantes University, France

ABSTRACT

The goal of this chapter is twofold. First, assuming that all agents belong to a genetic population, the evolution of inflation learning will be studied using a heterogeneous genetic learning process. Second, by using real-floating-point coding and different genetic operators, the quality of the learning tools and their possible impact on the learning process will be examined.

INTRODUCTION

A quick analysis of the most recent literature in economics shows a tremendous increase in the number of articles using *genetic algorithms (GAs)*. This tendency can be observed in nearly all areas of economics, from econometrics to finance (see Vallée & Yildizoğ u, 2004, for a panorama of GAs' applications in economics). Nevertheless, although GAs are always learning algorithms, the motivations for using GAs in economics may be divided into two categories. First, GAs may be used as a *simple* numerical learning tool. That is, GAs will be used in order to find numerical values for nonlinear models of growth (Dorsey & Mayer, 1995; Beaumont & Bradshaw, 1995), optimal parameters values of some nonlinear regressions (Pan, Chen, Khang, & Zhang, 1995), best functions of regression (Szpiro, 1997), and best stock exchange estimators (Pereira, 2000). Second, and related to this chapter, GAs may be used as a metaphor of individual/social learning process in order to study, for example, the equilibrium convergence in some dynamic macroeconomics models (Arifovic, 1996, 2001; Dawid, 1999), how cooperation may emerge in prisoner dilemma game (Axelrod, 1987; Yao & Darwen, 2000), or the individual R&D strategies of firms (Yýldýzoğ lu, 2002).

This chapter belongs to the second use of GAs. It will show how one can use a genetic algorithm in order to simulate and understand a learning process in a particular economic model. In this chapter, the economic model we use is a standard Keneysian monetary game as described in many economics textbooks (e.g., Mankiw, 2004). We chose this particular model for two reasons. First, its simplicity. Second, it generated a huge literature dealing with time inconsistency (Kydland & Prescott, 1977; Vallée, Deissenberg, & Başar, 1999), credibility (Cukierman & Meltzer, 1986; Drazen & Masson, 1993), reputation (Barro & Gordon, 1983; Backus & Driffill, 1985), and central bank independence (McCallum, 1997; Blinder, 1998; Driffil & Rotondi, (2005).

This extensive literature on credibility suggests that an optimal response to cheating behavior requires punishment. Here cheating is understood as a discrepancy between an agent's announced and implemented strategies. The analysis is often applied to understand, for example, the unemployment-inflation tradeoff dilemma.

Obviously, if an economic agent wants to punish another one, there is a need to find an efficient punishment strategy. This is characterized by minimizing both the likelihood that future cheating behavior will occur and the eventual costs to the punisher. In many scenarios such an optimal punishment strategy may initially be unknown. As a consequence, the punisher will have to invest resources in order to learn it.

In this chapter, such a learning process will be modeled using genetic algorithms (GAs). In the proposed framework, each agent is understood to be a member of the genetic population. Although the learning process in GAs is an adaptive one, each member may not learn in the same way. Indeed, some agents may implement strategies that are incompatible with the optimal learning process. Thus, as indicated by Dawid (1996), GAs appear to be a suitable tool to model such a heterogeneous learning process. Nonetheless, effectively using such a GA requires a full understanding of its mechanism.

The behavior of GAs is strongly determined by the balance between exploiting what already works best and exploring possibilities that might eventually evolve into something even better (Herrera, 2000). Riechmann (1999, 2001) showed that three different learning schemes are contained in GAs: these correspond to learning by imitation, by communication, and by experiment. Whereas the former takes place as a follow-the-leader process, learning by communication happens when economic agents meet and exchange information, while the ability of at least one agent to innovate is crucial for learning by experiment.

In this chapter it is contended that the foregoing interpretations of learning are not sufficient. More specifically, genetic learning should be analyzed as an information treatment process built on two components. First, there is a need to generate and transmit a given quantity of information. Such a process can be realized by experiment, imitation, or improving the quality of communication. Second, this information must be appropriately used. In other words, the agents must have the capacity to evaluate such information in order to properly respond by an appropriate change of strategy. Of course, there are key issues concerning the quality of the foregoing processes, as well as to the optimal quantity of information to be exchanged. Thus, a poor assimilation of available information can hamper a learning process. Similarly, unsuitable, insufficient, or excessive information may interfere with the quality of a learning process.

The goal of this chapter is twofold. First, assuming that all agents belong to a genetic population, the evolution of inflation learning

will be studied using a heterogeneous genetic learning process. Second, by using real-floating-point coding and different genetic operators, the quality of the learning tools and their possible impact on the learning process will be examined.

The outline of the rest of this chapter is as follows. In the next section, we present a simple unemployment-inflation game. An overview of a real coding genetic algorithm is then provided, and a specific genetic algorithm is used to simulate a heterogeneous inflation learning process. Subsequently, the associated simulations are analyzed, and finally, an indicator for measuring the amount of information that is exchanged during the learning process is proposed.

AN INFLATION-UNEMPLOYMENT GAME WITH HETEROGENEOUS AGENTS

The Model with Homogeneous Agents

In a standard inflation-unemployment game, such as the linear-quadratic framework proposed by Barro and Gordon (1983a, 1983b), there are two players: the government and the private agents. They are, respectively, leader and followers. The government's instruments are both the announced and the real level of inflation, respectively π^a and π, while the private agents' instrument are their levels of expected inflation. In this version of the model, the latter are understood to have an identical value of π^e. The instruments π and π^e critically determine the level of unemployment u. Assuming a standard Phillips curve relation:

$$u = u_n - c(\pi - \pi^e) \tag{1}$$

Here, u_n is the natural rate of unemployment, while c is a positive constant. In such a framework, it is possible for the government to drive the economy below the natural rate of unemployment, provided it can fool the private sector's expectations regarding inflation. One way to do so is to manipulate the announced inflation target π^a, which in turn can impact the private agents' expectations. Consequently, the divergences between π and π^a corresponds to the willingness of the government to cheat. In a repeated game, if the government cheats in a given period $\pi \neq \pi^a$, the private agents may subsequently not believe such an announcement so that there is an associated loss of credibility. Yet, this implies that the anticipations of the private agents must be defined by a reactive rule, which takes into consideration the discrepancies between announced and actual rate of inflation π. If the private agents cannot surmise the government's targets, which are defined by a loss function, then their expectations may never converge to the true rate of inflation . Accordingly, the process of learning about the government's loss function can itself modify the evolution of the inflation path.

Let us assume that the government's loss function is given by a standard linear quadratic form:

$$J^L = \frac{1}{2}\left\{(u-\bar{u})^2 + (\pi - \bar{\pi})^2\right\} \tag{2}$$

where $\bar{u}$ and $\bar{\pi}$ are, respectively, the targeted levels of unemployment and inflation, while u is determined by the Phillips curve relation in (1), once the government has set the inflation rate.

It will be assumed that the private agents seek to minimize the errors in their expectations, as specified by the following loss function:

$$J^F = \frac{1}{2}\left(\pi^e - \pi\right)^2 \tag{3}$$

If the government is credible, then the actual rate of inflation is supposed to be identical to the announced one such that $\pi = \pi^a$.

In light of equations (1) and (2), the reaction function of the government is defined by minimizing its loss function with respect to its instrument, the inflation rate, such that:

$$\pi = T^L(\pi^e) \equiv \arg\min_{\pi} J^L(\pi, \pi^e) = \frac{c(u_n - \bar{u}) + c^2\pi^e + \bar{\pi}}{1 + c^2} \tag{4}$$

Analogously, the reaction function corresponding to the private agents is defined by minimizing the loss function (3) with respect to their expected inflation rate, so that:

$$\pi^e = T^F(\pi^a) \equiv \arg\min_{\pi^e} J^F(\pi^a, \pi^e) = \pi^a \tag{5}$$

A one-period game can then be simulated for specific parameters and target values under alternative formulations for the government's conceivable strategies. More specifically, four different strategies are highlighted by the results in Table 1 (see Vallée et al., 1999, for a description of such behavior).

Here the government alternatively makes a clear commitment such that the announced and actual rates of inflation are the same (case 1), whereby the government experiences a loss, but not the private agents. In cases 2 and 3, the government cheats $\pi \neq \pi^a$, so that it eliminates its own losses, but this has the side effect of generating high losses for the private sector. The final case corresponds to a Nash equilibrium which is defined by:

$$\pi^{Nash} = \bar{\pi} + c(u_n - \bar{u}), \text{ with } \pi^{Nash} = T^L(T^F(\pi^{Nash})) \tag{6}$$

In this outcome, there is no cheating and hence no losses for the private sector.

Table 1. Inflation-unemployment game— c = 1, $\bar{\pi}$ = 2, $\bar{u}$ = 5, and u_n = 7

Strategies	π^a	π^e	π	J^L	J^F
(1) Commitment	2	2	2	2	0
(2) No commitment (cheating)	2	2	3	1	0.5
(3) Optimal cheating	0	0	2	0	2
(4) Nash	4	4	4	4	0

Obviously, if this game is repeated, cheating behavior will lead the private agents not to trust the government any longer. Consequently, they will have to learn what the evolution of the government's inflation target is. An optimal learning process should converge to the Nash inflation level, since at this point the expected and actual rates of inflation are convergent.

Heterogeneous Agents

If it is assumed that there exist N heterogeneous private agents,[1] the anticipated level of inflation becomes an arithmetic average of the individuals' different expectations, which is understood to be known by the government:

$$\pi^e = \frac{1}{N}\sum_{i=1}^{N} \pi_i^e \tag{7}$$

It is again assumed that each private agent seeks to minimize the errors in its expectations, such that:

$$J_i^F(\pi, \pi_i^e) = \frac{1}{2}(\pi - \pi_i^e)^2 \tag{8}$$

As in the previous one-period example, assuming that the government wants to cheat optimally the private agents, then the government will announce in the first period a zero rate of inflation $\pi_1^a = 0$. Given such an announcement, each agent will choose a level of

expected inflation $\pi^e_{i,1}$, with $i = 1.., N$. This will depend on its assessment of the government's credibility. The average anticipated value of inflation $\pi^e_1 = \frac{1}{N}\sum_{i=1}^{N} \pi^e_{i,1}$ will be influenced by the heterogeneity of the agents' expectations, where $\pi^e_{i,1} \neq \pi^e_{j,1}$ for some agents i and j. In light of this expected inflation rate π^e, the government is free to choose what will be the actual rate π, where the reaction function in (4) again applies.

For those private agents that are initially fooled, the credibility of the government's subsequent announcement in period is damaged. As a consequence, the individual private agents will each seek to learn what the government's actual inflation target is. This leads to a dynamic inflation game, where the private agents' loss is minimized by the Nash inflation target value. That is, each private agent, by choosing its expected rate, will impact the average level of expectations to which the government will, in turn, respond by setting an optimal inflation rate in the same period. The only way for the private agents to achieve a zero value for their loss function is for all of them to anticipate the Nash equilibrium. It is of particular significance that such a solution requires a convergence of expectations across an initially heterogeneous population.[2] In summary, the following decision sequence holds:

- Agent i chooses $\pi^e_{i,t}$, which then results in an average expected inflation level π^e_t.
- The government reacts to the expectations by choosing the actual inflation rate $\pi_t = T^L(\pi^e_t)$.

HETEROGENEOUS LEARNING USING GENETIC ALGORITHMS

An Overview of Genetic Algorithms

Genetic algorithms are heuristic search methods that are based on biological evolution.[3] We briefly recall that the structures of GAs are based on the following steps:

1. **Initialization:** An initial population[4] of N chromosomes is created (e.g., first actions/strategies are chosen randomly).
2. **Evaluation:** For each chromosome, its fitness is evaluated by an appropriate function.
3. **Selection:** A new population of N chromosomes is created by using a selection method.
4. **Reproduction:** Possible crossovers and mutations occur for this new population.
5. **Evolution:** The process starts again at step 2.

The behavior of GAs is strongly determined by the balance between exploiting what already works best and exploring possibilities that might eventually evolve into something even better (Herrera, 2000). In other words, there is an optional balance between learning by imitation and by communication, as opposed to learning by experiment (Riechmann, 1999).[5]

Each chromosome is made up of a sequence of genes from a given alphabet. Since the concern here is with real-valued actions, floating-point numbers are used directly, rather than binary digits. Hence, it is assumed that more natural representations have a greater efficiency and produce better solutions.[6] Furthermore, real-point-coding offers a larger set of crossover and mutation operators. In this chapter two kinds of both crossover and mutation operators are used: the arithmetic and heuristic crossover operators, and the uniform and boundary mutation ones.[7] For each of these operators, an appropriate frequency is set, which corresponds to the discrete number of times the operator is called at every generation.

Crossover Operators

Let X and Y be two parents that are members of the population. The arithmetic crossover creates two complimentary linear combinations of these parents. That is, after generating a random number ($\alpha = U(0,1)$), the new parents are set according to:

$$\tilde{X} = \alpha X + (1-\alpha)Y \tag{9}$$

$$\tilde{Y} = (1-\alpha)X + \alpha Y \tag{10}$$

Heuristic crossover produces a linear extrapolation of the parents by using the fitness values of the two parent chromosomes to determine the direction of the search. It is possible that the solution generated will not be feasible. If a feasible chromosome is not produced, the parents are returned as children. So, a new child $\tilde{X}$ is created by the following process: given that the fitness of X is better than the one of Y,[8] and assuming that the solution is possible (feasibility equals to one), we have:

$$\begin{aligned} \tilde{X} &= X + r(X - Y), \\ \tilde{Y} &= X. \end{aligned} \quad \text{with}$$

$$\text{Feasibility} = \begin{cases} 1, \text{ if } b_1 < X < b_2 \\ 0, \text{ otherwise} \end{cases}$$

Here, $r = U(0,1)$, while b_1 and b_2 are the authorized bounds of X, defining the allowing search space to which each admissible solution must belong to, such that $X \in [b_1, b_2]$. If $\tilde{X} \notin [b_1, b_2]$, a new random number r is generated, and the process is repeated until a feasible solution is reached or until a given number of attempts have been attained. However, if X and Y have the same fitness, X and Y are reproduced without change. As discussed by Michalewicz (1992, p. 129), this operator is a unique crossover, since it: (1) uses values of the objective function in determining the direction of the search, (2) produces only one offspring, and/or (3) may not produce offspring at all.

The crossover operator is a method for sharing information between chromosomes.

Mutation Operators

Uniform mutation randomly selects one variable and sets it equal to a random number from a uniform distribution on the bounds interval . Boundary mutation sets the value of to or , with probability .

The mutations operator is a suitable method to create innovations in the population.

Other Operators

Finally, the roulette wheel is used as a selection operator,[9] such that the two parents are selected with a probability directly proportional to their relative fitness. Hence, letting $f(X^i)$ denote the fitness value of chromosome X^i, the probability that this chromosome will be selected for the reproduction phase is:

$$\frac{f(X^i))}{\sum_{j=1}^{N} f(X^j))}.$$

We also allow for an elitism procedure in some simulations, whereby a given proportion of the best members of the previous population is retained.

These two operators allow imitations between chromosomes.

The Pseudo Code

Each member of the genetic population is characterized by its anticipated level of inflation.

This level will change according to the genetic operators. In order to avoid excessive changes, the values of anticipations must belong within a given dynamic interval. That is, each agent's anticipation will have to belong to the range $[\pi^e_{min,t}, \pi^e_{max,t}]$, with at least $\pi^e_{min,t} \geq 0, \forall t$. In summary, the economic genetic algorithm used can be described by the following steps:

1. Initial announcement of inflation π^a_1.
2. Calculation of the first range of authorized anticipations: $[\pi^e_{min,1}, \pi^e_{max,1}]$.
3. Creation of the initial population of agents characterized by their strategies $\pi^e_{i,1}$.
4. Calculation of the average anticipated value of inflation $\pi^e_1 = \frac{1}{N}\sum_{i=1}^{N} \pi^e_{i,1}$.
5. Calculation of the realized value of inflation $\pi_1 = T^L(\pi^e_1)$.
6. Evaluation of each agent's fitness $f_{i,1}(\pi^e_{i,1}, \pi_1)$.
7. For j = 2 to T, there is the
   a. Calculation of the new range $[\pi^e_{min,j}, \pi^e_{max,j}]$.
   b. Creation of a new population (reproduction, crossover, and mutation).
   c. Calculation of π^e_j.
   d. Calculation of π_j.
   e. Evaluation of each agent's fitness $f_{i,j}(\pi^e_{i,j}, \pi_j)$.
8. End.

SIMULATIONS

The following initial values were used in all of the simulations: $c = 1$, $\bar{\pi} = 2$, $\bar{u} = 5$, $u_n = 7$, and $k = 0.5$. The fitness function to be maximized is specified by: $f(\pi^e_{i,t}, \pi_t) = 400 - 20(\pi_t - \pi^e_{i,t})^2$, while the authorized range of changes in the anticipated level of inflation is defined by the following rule:

$\pi^e_{min,t} = (1-k)\pi_{t-1}$, $\pi^e_{max,t} = (1+k)\pi_{t-1}$, with $k > 0$

If $\pi^e_{min,t} < 0$, $\forall t \neq 1$, its value is changed to 0. Furthermore, each simulation starts with $\pi^a_1 = 0$. The anticipation error $\varepsilon_t = \pi^e_t - \pi_t$ is then calculated at every $t \in [1, T]$.

Simulation 1: No Mutations and Arithmetic Crossovers

When the population size is set to 50, and the crossover frequency to 35%, there are 18 arithmetic crossovers. The initial anticipation's range is [0,1]. Results are illustrated in Figure 1. It is apparent that, on average, the private agents, as a whole, do not find the optimal Nash inflation level ($\pi^N = 4$).

How should such a result be interpreted? Reproduction and crossover operators allow the agents to use imitation and communication for their learning goals. Of course, if nobody is initially well informed, no one may expect that a state of perfect information (e.g., strategy)

Figure 1. Evolution of π^e_t and π_t, N = 50, $[\pi^e_{min,1}, \pi^e_{max,1}] = [0,1]$, 30 runs

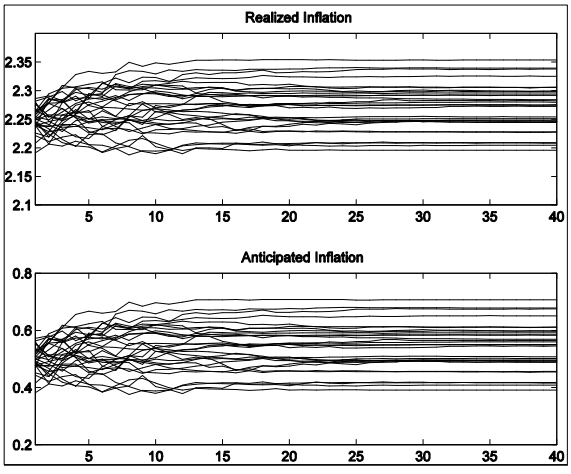
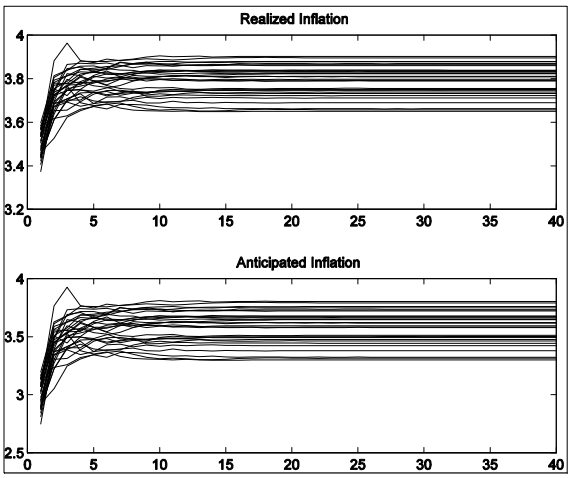


Figure 2. Evolution of π_t^e *and* π_t, $N = 150$, $[\pi^e_{min,1}, \pi^e_{max,1}] = [0,6]$, *30 runs*



will be reached. Such a non-convergence could mean that the quality of learning based on imitation and on simple communication is not good enough. However, it may also mean that the initial stock of available information was not sufficiently efficient. This question can be elucidated by including the Nash value in the initial anticipation range, which in this case means to set $[\pi^e_{min,1}, \pi^e_{max,1}] = [0,6]$. Furthermore, even when the population is increased to 150 agents, the learning process still is not stabilized around the Nash value, as shown in Figure 2. Thus, even if, at some future time *t*, some agents are perfectly informed $\pi^e_{i,t} = 4$, the learning process is not able to exploit such information later on.

In conclusion, heterogeneous genetic learning, based on the use of imitation and simple communication, is not able to achieve a perfect learning outcome. Such a result can be explained by the insufficient creation of new useful information. As pointed out by Riechmann (1999, p. 237): "If nobody knows about a certain detail, which could make a good strategy a perfect one, then nobody can acquire this detail by communication." Thus, learning by communication is able to find better strategies than learning by imitation, because the best details of alternative strategies can be combined. In addition, still the learning may not be able to exploit good information, if the quality of communication is poor. Therefore, to increase the quality of the heterogeneous genetic learning, we have either to allow new experiments (mutations) or to introduce a better quality of communication between agents (heuristic crossovers).

Simulation 2: Arithmetic Crossovers and Mutations

As in Simulation 1, the population size and the crossover frequency are set to, respectively, 50% and 35%, while the initial range for the anticipations is defined over [0,1]. However, a 4% frequency of mutations is assumed, which means that two agents are authorized to experiment unknown strategies at each period. Nonetheless, when experiments provide new information to be exploited, they also decrease the stability of the learning process (see Goldberg, 1989). As illustrated in Figures 3 and 4, adding a uniform mutation makes possible the convergence to the Nash average level of inflation. Nevertheless, this learning convergence is slow. More interesting, Figure 5 shows that the learning process does not involve systematic errors. In four simulations, with $t \in [400,1000]$, the average error of anticipations $\bar{\varepsilon} = \frac{1}{601}\sum_{t=400}^{1000} \varepsilon_t$ is, respectively, 0.0149, –0.0703, 0.0236, and 0.0198, while the overall average is . With 16 runs (Figure 6), and with , the overall average is –0.0030. With 16 runs (Figure 6), and with $t \in [500,2000]$, the overall average is 0.0049.

It should be stressed that all the mutation operators do not generate the same properties. Hence, using boundary mutation rather than a

Figure 3. Evolution of π_t^e and π_t, N = 50, 30 runs, uniform mutations

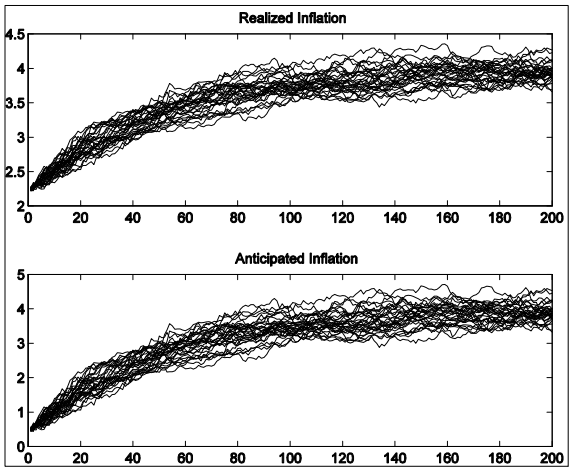


Figure 4. Evolution of π_t^e and π_t, 4 runs, N = 50, T = 1000 and uniform mutations

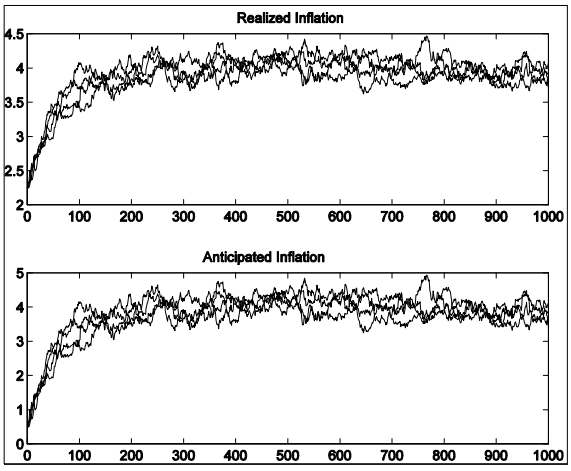


Figure 5. Evolution of ε_t, uniform mutations

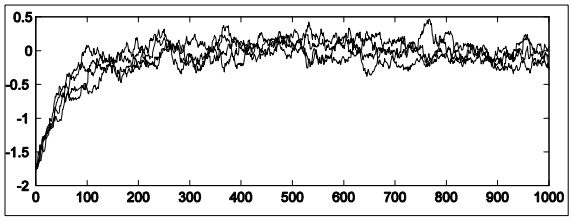
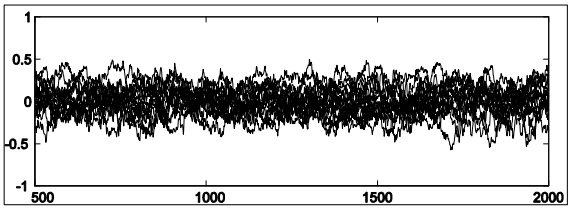


Figure 6. Evolution of ε_t, uniform mutations

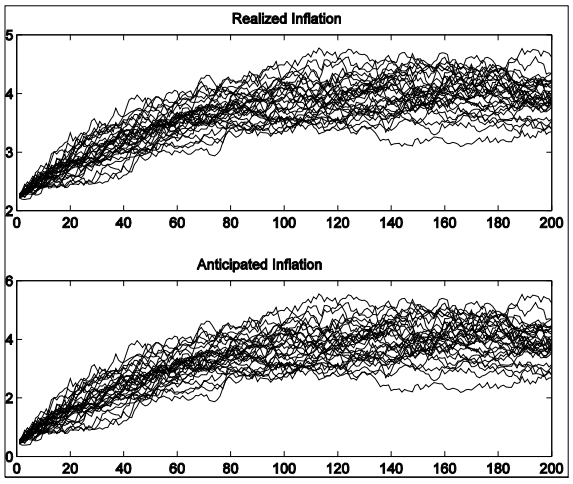


uniform mutation changes the results as follows (see Figure 7). First, the learning convergence is faster. This is due to the fact that boundary mutation quickly increases the range of authorized anticipations $[\pi^e_{min,j}, \pi^e_{max,j}]$. Nevertheless, this operator creates an increased quantity of useless information. Hence, by allowing inadequate experiments later on, useless information (noise) is generated. As shown in Figure 8, the variances of the ε_t are higher than the ones using uniform mutation. In the four runs, and with $t \in [500, 2000]$, the corresponding values of $\bar{\varepsilon}$ are, respectively, –0.674, 0.0478, 0.0414, and 0.0449. The overall average is 0.0166.

Simulation 3: No Mutations and Heuristic Crossovers

The quality of the communication can be enhanced by using a more efficient tool as the heuristic crossover operator. As in the case of the arithmetic crossover, this operator allows two private agents to exchange information. However, with the heuristic crossover, the agents exchange signals (chat) much more. As already indicated, the heuristic crossover is the unique genetic operator that directly uses information from the fitness function.[10] Therefore, the communication contains more relevant in-

Figure 7. Evolution of π_t and π_t^e, 30 runs, boundary mutations



formation. The question asked by each agent is no longer only what was done last period, but also how beneficial were such strategies.

Figure 9 illustrates 30 runs based on a population of 50 agents, and with the heuristic crossover frequency set to 5%. It is easily established that learning is both optimal and fast.

Figure 9. Evolution of π_t^e and π_t, 30 runs, N = 50, heuristic crossovers

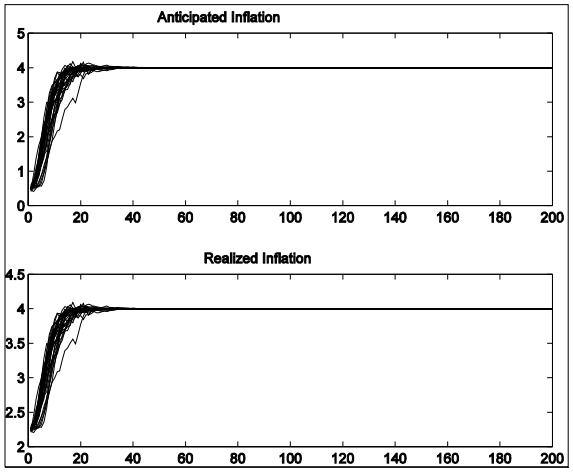
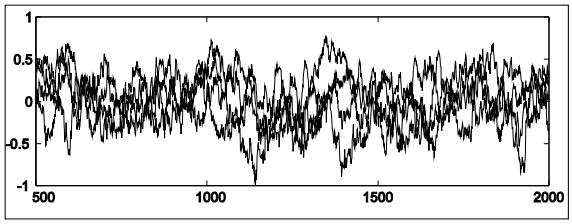


Figure 8. Evolution of ε_t, boundary mutations



Second, the learning is perfect ($\varepsilon = 0$). Nonetheless, even if a higher quality of communication is provided by the heuristic crossover operator, quality by itself does not suffice without the quantity. Such a quantity of relevant information depends on the population size. As shown by Figures 10 to 12, when the population size is too low, some runs do not converge to the Nash value.

Simulation 4: Elitism Procedure

During the learning process, a potentially good individual strategy can disappear. This is because of a bad experiment (mutation) or be-

Figure 10. Evolution of π_t^e and π_t, 30 runs, N = 5, heuristic crossovers

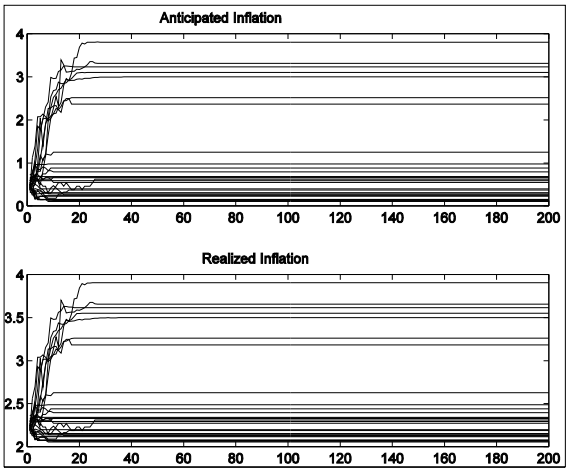


Figure 11. Evolution of π_t^e and π_t, 30 runs, $N = 10$, heuristic crossovers

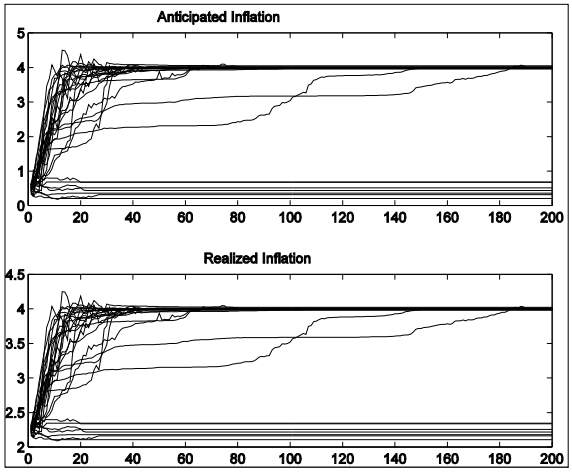
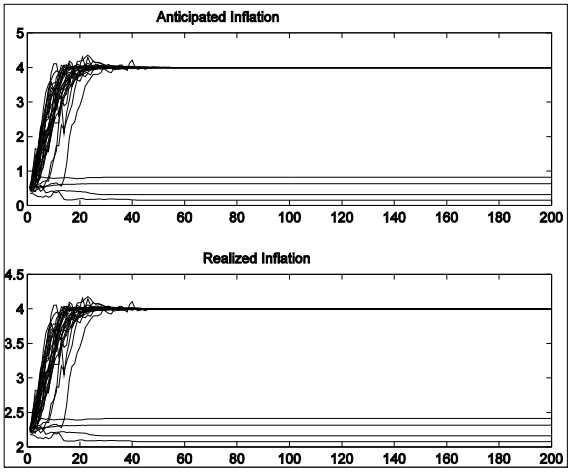


Figure 12. Evolution of π_t^e and π_t, 30 runs, $N = 20$, heuristic crossovers



cause of a bad imitation (even if $\pi_{i,t}^e = 4$ for some *i*, still it is possible that $\pi_t^e \neq 4$). Thus, social learning may involve individual imitation to choose the wrong direction. In order to avoid it, a well-known solution in GAs is to add an elitism procedure, whereby some agents will keep their best previous strategies.

A set of simulations, illustrating such an elitism procedure, were undertaken by setting the frequencies of the arithmetic crossover and uniform mutation to, respectively, 35% and 4%, and adding five elitist agents out of an overall population of 50 agents. As illustrated in Figures 13 and 14, the following expected results were verified. Not only was the learning fast and optimal, but also stabilization of the heterogeneous learning process was achieved.[11]

Simulation 5: Perfect Expectations

Let us now assume that some private agents do have perfect expectations, so π_t is known at every time *t*. Introducing just a few such agents in the population should increase the quality of the learning process, given the perfect quality of their information. As a result, imitation and simple communication may be sufficient to ensure optimal learning, so long as badly informed agents meet well-informed ones. Two kinds of simulations were run. Initially, it was assumed that 10% of the population held perfect expectations. Subsequently, this proportion was increased to 50%. All the simulations were based on a 35% frequency of arithmetic crossovers. As expected, learning is optimal and faster when the number of well-informed agents increases (Figure 15).

A GENETIC ENTROPY MEASURE

A central issue raised in this chapter relates to Riechmann's question[12] of whether heterogeneous genetic algorithms learning leads to an optimal stable learning, and if so, how? The simulation analysis suggests four conceivable answers.

First, as already noticed by Riechmann (1999, 2001), simple communication and imitation, or, alternatively, arithmetic crossovers and repro-

Figure 13. Evolution of π_t^e and π_t, 5 elitist agents, N = 50, 30 runs

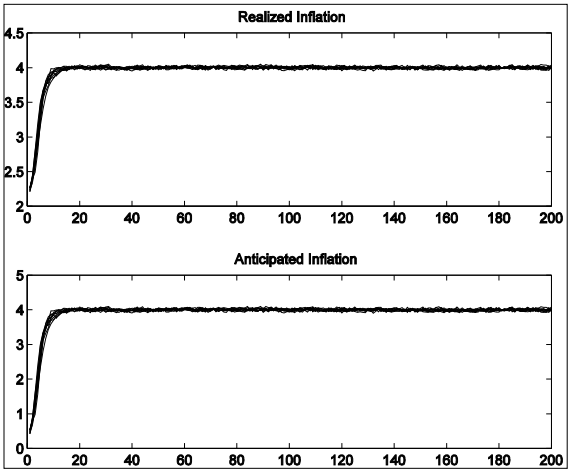
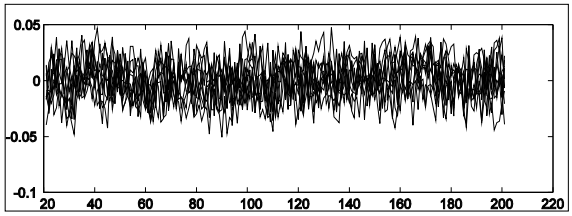


Figure 14. Evolution of ε_t, 10 runs, 5 elitist agents



ductions, are generally not enough to get a stable and optimal learning process. As pointed out, an exception occurs when perfectly informed agents are present in the population. Second, adding experiments (mutations) may compensate the lack of communication between agents. Nonetheless, even if such individual learning may initially be helpful, this selfish behavior may interfere with the stabilization of the learning process. Third, in contrast to experiments, optimal and stable heterogeneous learning may be reached by improving the quality of the communication, namely of the exchanged information. Fourth, it has been demonstrated that increasing the quality of the communication is a necessary condition, but not a sufficient one, since the population must be able to exchange a sufficient quantity of information.

In order to develop further this last point, the following measure, reflecting the number of agents' strategies changes, is defined:

$$NC = \sum_{t=2}^{T} \sum_{i=1}^{N} (\pi_{i,t}^e - \pi_{i,t-1}^e).$$

The value of the *NC* value indicates the number of changes realized during the learning process, which can be interpreted as measuring the amount of exchanged information. If the learning process never stabilizes, the value will never stop increasing. As in standard entropy theory, such an outcome may be interpreted as

Figure 15. Evolution of π_t^e and π_t, 10 (left) and 50 (right) well-informed agents, N = 100

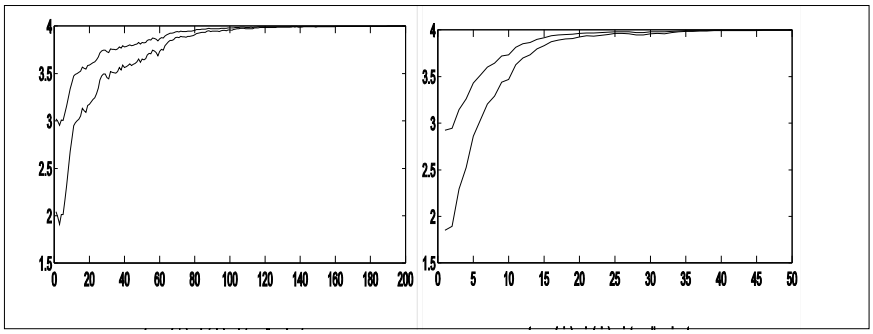


Table 2. Values of NC for alternative simulations

Run	AC	HC	AC-UM	AC-BM	AC-UM-EL
1	139.16	863.9	4837.3	7919.5	1948.3
2	136.24	792.1	4850.4	8754.4	1975.6
3	99.03	976.7	4823.9	8554.7	1982.3
4	147.89	811.8	4751.1	7996.0	2030.3
5	127.38	851.5	4791.9	7547.7	2105.2
6	107.99	950.7	4517.0	7077.3	1899.5
7	100.66	811.1	4538.7	9197.4	2044.4
8	102.61	760.6	5180.1	8479.2	1932.1
9	166.75	988.6	4965.8	9540.6	2018.8
10	85.95	986.5	4775.6	8308.0	2075.3
Total	1213.71	9393.5	48032	83375	20025

a high level of noise in the system. In comparison, a low constant value of *NC* may mean that the system is stabilized. Nonetheless, it does not imply that the learning has been optimal. Indeed, it will now be shown that there exists an optimal value for this measure.

Study of the Value

The different values for *NC* will first be compared for the previous simulations. These results are summarized in Table 2 on the basis of simulations using a population of 50 agents, 35% frequencies for the arithmetic crossover (AC) and heuristic crossover (HC), along with a 4% frequency for uniform (UM) and boundary (BM) mutations. Finally, elitism (EL) was set to a value of 10%.

An analysis of the reported measures for *NC* in Table 2 leads to the following conclusions. First, as expected, introducing elitist agents stabilizes the learning process by decreasing the quantity of exchanged information. Specifically, the total value of *NC* decreases from 48032 to 20025. Second, a minimal value of is required to achieve optimal learning. In this regard, the *NC* value from the AC run (around 100) is too low in comparison with the *NC* value from the HC run (around 900). Third, allowing experiments (corresponding to the AC-UM and AC-MB runs) improves the convergence of the learning process, but it also increases the amount of useless noise. In particular, the total values for *NC* are, respectively, 48032 and 83375, as opposed to 9393.5 for the HC run.

Impact of Changing Population Size and Frequencies

As is well known, a low population size does not permit efficient learning, regardless of the quality of the learning process. The results, presented in Table 3, corroborate this. With a population of only 10 agents, HC runs with a low value of never approach the Nash value. As is readily verified, results reported in Table 3 are compatible with those in Table 2. For example, a value of *NC* of 180.63, when $N=10$, is equivalent to a value of *NC* of $180.63 \times 5 = 903.15$, when $N = 50$.

The impact of changing the frequency levels will now be examined by using AC and HC simulations. Table 4 summarizes three simulations using, respectively, 10%, 35%, and 80% frequency levels. It is straightforward to verify that an increase of the frequency does not have the same consequences when arithmetic crossovers are used rather than the heuristic ones. Hence, with simple communication (AC), decreasing frequency increases the quantity of

Table 3. Values of NC, 10 HC runs, N = 10

RUN	π_T^e	NC value	RUN	π_T^e	NC value
1	0.9350	6.9330	6	4.0008	174.54
2	3.6352	157.02	7	4.0004	76.133
3	4.0000	180.63	8	0.7465	9.6079
4	0.8286	6.0002	9	0.6215	10.844
5	0.5387	7.3766	10	4.0000	137.88

Table 4. Average values of and on 30 runs,

Frequency	AC	HC
10%		
NC	246	1331.8
π_T^e	0.63861	4.00016
35%		
NC	128.31	1004.38
π_T^e	0.52375	4.00000
80%		
NC	78.2	95558.6
π_T^e	0.53036	3.91924

exchanged information (78.2 to 246). As a result, the average anticipated value of inflation is also slightly improved from 0.53 to 0.63. Such a result can be readily explained since simple communication tends to create a homogeneous population. With a lower frequency, this homogenization process slows down. Thus, the amount of exchanged information may increase. However, this interpretation does not hold with heuristic crossover. In such a case where this operator provides a higher quality of communication, an excessively low frequency means a lower use of this useful communication tool. Consequently, the possibility of exchanging some reliable information decreases. Nevertheless, an excessive use of this communication tool creates useless noise. Analogous to Greshman's Law, too much information kills information. This result confirms that both an optimal value of and an optimal frequency level should exist. In conclusion, whatever the quality of the communication tools, they need to be used in a suitable manner.

CONCLUSION

While it has been demonstrated that a bad learning process may be improved, either by allowing new experiments or by improving the communication between agents, the latter approach appears preferable. It has also been remarked that, whatever the learning tools, they must be suitably used. Specifically with a heterogeneous learning process, agents should not be allowed to communicate too much when the communication is not good, as with arithmetic crossovers, or when the communication provides too much information, as with heuristic crossovers. Nonetheless, to provide a better interpretation of such results, the "entropy" measure of the genetic learning process needs to be more finely specified. This issue remains for future research.

ACKNOWLEDGMENT

The author thanks Robert F. Owen and C. Collin for an extensive number of crucial suggestions.

REFERENCES

Arifovic, J. (1996). The behavior of the exchange rate in the genetic algorithm and experimental economies. *Journal of Political Economy, 104*(3), 510-541.

Arifovic, J. (2001). Evolutionary dynamics of currency substitution. *Journal of Economic dynamics and Controls, 25*(3-4), 395-417.

Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. In L. D.

Davis (Ed.), *Genetic algorithms and simulated annealing*. San Francisco: Morgan Kaufmann.

Backus, D., & Drifill, J. (1985). Inflation and reputation. *American Economic Review, 75,* 530-538.

Barro, R., & Gordon, D. (1983a). A positive theory of monetary policy in a natural rate model. *Journal of Political Economy, 91*(4), 589-610.

Barro, R., & Gordon, D. (1983b). Rules, discretion and reputation in a model of monetary policy. *Journal of Monetary Economics, 12*(1), 101-121.

Beaumont, P., & Bradshaw, P. (1995). A distributed parallel genetic algorithm for solving optimal growth models. *Computational Economics, 8*(3), 159-180.

Blinder, A. S. (1998). *Central banking in theory and practice*. Cambridge, MA: The MIT Press.

Bonini, P. (2001). *New Keynesian economics and credibility analysis*. PhD Thesis, University of Birmingham, USA.

Cukierman, A., & Meltzer, A. H. (1986). A theory of ambiguity, credibility and inflation under discretion and asymmetric information. *Econometrica, 54,* 1099-1128.

Dawid, H. (1996). Adaptive learning by genetic algorithm. *Lecture Notes in Economics and Mathematical Systems,* (441). Berlin: Springer-Verlag.

Dawid, H. (1999). *Adaptive learning by genetic algorithm. Analytical results and applications to economic models*. Berlin: Springer-Verlag.

Dorsey, R., & Mayer, W. (1995). Genetic algorithms for estimation problems with multiple optima, nondifferentiability, and other irregular features. *Journal of Business and Economic Statistics, 13*(1), 53-66.

Drazen, A., & Masson, P. (1993). *Credibility of policies versus credibility of policymakers*. NBER Working Paper.

Driffill J., & Rotondi, Z. (2005). *Credibility of optimal monetary delegation: A comment*. Working Paper.

Faust, J., & Svensson, L. (2001). Transparency and credibility: Monetary policy with unobservable goals. *International Economic Review, 42,* 369-397.

Ginsburgh, V., & Michel, P. (1998). Optimal policy business cycles. *Journal of Economic Dynamics and Control, 22*(4), 503-518.

Herrera, F., & Lozano, M. (2000). Gradual distributed real-coded genetic algorithms. *IEEE Transactions on Evolutionary Computation, 4*(1), 43-63.

Jensen, H. (1997). Credibility of optimal monetary delegation. *American Economic Review, 87,* 911-920.

Krishnakumar, K., & Goldberg, D. (1992). Control system optimization using genetic algorithm. *Journal of Guidance, Control, and Dynamics, 15*(3), 735-740.

Kydland, F., & Prescott, E. (1977). Rules rather than discretion: The inconsistency of optimal plans. *Journal of Political Economy, 85*(3), 473-492.

Lozano, M., Herrera, F., Krasnogor, N., & Molina, D. (2004). Real-coded memetic algorithms with crossover hill-climbing. *Evolutionary Computation, 12*(3), 273-302.

Mankiw, G. (2004). *Macroeconomics* (5th ed.). New York: Worth Publishers.

McCallum, B. (1997). Crucial issues concerning central bank independence. *Journal of Monetary Economics, 39,* 99-112.

Mertz, P., & Freisleben, B. (2000). Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE Transactions on Evolutionary Computation, 4*(4), 337-352.

Michalewicz, Z. (1992). *Genetic algorithms + data structures = evolution programs.* Berlin: Springer-Verlag.

Michalewicz, Z. (1994). Evolutionary computation techniques for nonlinear programming problems. *International Transactions in Operational Research, 1*(2), 223-240.

Michalewicz, Z., & Attia, N. (1994). Evolutionary optimization of constrained problems. In A.V. Sebald & L.J. Fogel (Eds.), *Proceedings of the 3rd Annual Conference on Evolutionary Programming* (pp. 98-108). River Edge, NJ: World Scientific.

Michalewicz, Z., Janikow, C., & Krawczyk, J. (1992). A modified genetic algorithm for optimal control problems. *Computers and Mathematics with Applications, 23*(12), 83-94.

Özyildirim, S. (1997). Computing open-loop non-cooperative solution in discrete dynamic games. *Evolutionary Economics, 7*(1), 23-40.

Pan, Z., Chen, Y., Khang, L., & Zhang, Y. (1995). Parameter estimation by genetic algorithms for nonlinear regression. *Proceedings of the International Conference on Optimization Techniques and Applications* (pp. 946-953). Singapore: World Scientific.

Pereira, R. (2000). *Genetic algorithm optimization for finance and investment.* Technical Report, Latrobe University, USA.

Riechmann, T. (1999). Learning and behavioral stability—An economic interpretation of genetic algorithms. *Journal of Evolutionary Economics, 9*(2), 225-242.

Riechmann, T. (2001). Genetic algorithm learning and evolutionary games. *Journal of Economic Dynamics and Control, 25*(5-7), 1019-1037.

Riechmann, T. (2002). Genetic algorithm learning and economic evolution. In S.-H. Chen (Ed.), *Evolutionary computation in economic and finance* (pp. 45-60). Berlin: Springer-Verlag.

Szpiro, G. (1997). Forecasting chaotic time series with genetic algorithms. *Physical Review E, 55,* 2557-2568.

Vallée, T., Deissenberg, C., & Başar, T. (1999). Optimal open loop cheating in dynamic reversed linear quadratic Stackelberg games. *Annals of Operations Research, 88*(1), 217-232.

Vallée, T., & Yıldızoğlu, M. (2004). Présentation des algorithmes génétiques et de leurs applications en économie. *Revue d'Économie Politique, 114*(6), 711-745.

Yao, X., & Darwen, P. (2000). Genetic algorithms and evolutionary games. In *Commerce, complexity and evolution* (pp. 325-147). Cambridge, UK: Cambridge University Press.

Yıldızoğlu, M. (2002). Competing R&D strategies in an evolutionary industry model. *Computational Economics, 19*(1), 51-65.

ENDNOTES

[1] Ginsburgh and Michel (1998) have proposed a somewhat similar model where there is a mixed population consisting of both some agents with naive adaptive expectations, and others with rational expectations. Nonetheless, unlike the for-

mulation here, since all the adaptive expectations agents use the same learning rule, there is no real heterogeneous learning in their model.

2 In our example, it is only when $\pi_t^e = 4$ and $\pi_{i,t}^e = 4$, $\forall i$, that $\pi_t = 4$ and $J_i^F = 0$, $\forall i$. That is, the optimal learning process should lead the private agents, as a whole, to set the average anticipated level of inflation to the Nash value. However, perfect stability of the learning process will be reached only if $\pi_{i,t}^e = 4$, $\forall i$. Otherwise, some agents will have an incentive to change their anticipations.

3 Goldberg (1989) offers a good general introduction to the subject, while Riechmann (2001, 2002) has analyzed the links between GAs and evolutionary game theory. Vallée and Yıldızoğlu (2004) provide a panorama of GAs' applications in economics.

4 As noticed by Riechmann (2001): "It is important to clarify the following point: a genetic individual is not interpreted as an economic agent, but as an economic strategy *used by* an economic agent. This interpretation allows for several agents employing the same strategy."

5 GAs are known to be powerful in searching all regions of the state space, since mutation and crossing over allow GAs to search in some promising areas not included inside the initial population.

6 See Michaliwicz (1992) for a comparison between the binary and the real-point codings.

7 Again, see Michalewicz (1992) for a more precise presentation of these operators.

8 Otherwise, one has to change X and Y in the equations.

9 In fact, Riechmann (1999) demonstrated that "Lyapunov stability" of the genetic learning may be expected.

10 Indeed, this operator could change the standard genetic algorithm to a memetic one. This is also called a genetic local search algorithm or an hybrid evolutionary algorithm. Such an algorithm combines the efficiency of a genetic algorithm and the use of a local search method. See Merz and Freisleben (2000) and Lozano, Herrera, Krasnogor, and Molina (2004) for related discussions.

11 The specific values of $\bar{\varepsilon}$, with $t \in [20, 200]$, are, respectively, 0.0029, 0.0023, 0.00003, 0.0025, –0.0054, 0.0009, –0.0003, 0.0006, –0.0047, and 0.0017. The overall average is 0.000038369.

12 "Does genetic algorithm learning lead to behavioral stability, and if so, how?" (Riechmann, 1999, p. 226).

Chapter XIX
Modeling the Firm as an Artificial Neural Network

Jason Barr
Rutgers University, USA

Francesco Saraceno
Observatoire Français des Conjonctures Économiques, France

ABSTRACT

The purpose of this chapter is to make the case that first a standard artificial neural network can be used as a general model of the information processing activities of the firm; second, to present a synthesis of Barr and Saraceno (2002, 2004, 2005), who offer various models of the firm as an artificial neural network. An important motivation of this work is the desire to bridge the gap between economists, who are mainly interested in market outcomes, and management scholars, who focus on firm organization. The first model has the firm in a price-taking situation. We show that increasing environmental complexity is associated with larger firm size and lower profits. In the second and third models, neural networks compete in a Cournot game. We demonstrate that they can learn to converge to the Cournot-Nash equilibrium and that optimal network sizes increase with complexity. In addition, we investigate the conditions that are necessary for two networks to learn to collude over time.

INTRODUCTION

The purpose of this chapter is two-fold: (1) to make the case that a standard backward propagation artificial neural network can be used as a general model of the information processing activities of the firm, and (2) to present a synthesis of Barr and Saraceno (BS) (2002, 2004, 2005), who offer various models of the firm as an artificial neural network.

An important motivation of this work is the desire to bridge the gap between economists, who are mainly interested in market outcomes, and management scholars, who focus on the

internal components of the firm. The topic of optimal firm organization in relation to its environment has been extensively studied in the management literature (Lawrence & Lorsch, 1986; Burns & Stalker, 1994; Gailbraith, 1973; Chandler, 1980), but remains largely unexplored in industrial organization and in economics in general.

In spite of the path-breaking work of Coase (1937) and Williamson (1985), the internal nature of the firm continues to be seen by most economists as a black box production function. The few exceptions are within two streams of research: the area initiated by Nelson and Winter (1982), which studies the firm's activities from an evolutionary point of view, and more recently, the *agent-based* literature, developed to better understand the information processing behavior of the firm (see Chang & Harrington (2006) for a review of agent-based models of the firm). These papers define an *organization* as a group of boundedly rational individuals that pursues a common objective, transcending the individuals' objectives, in order to process information and make decisions. These models study the costs and benefits of different organizations.

Our models of the firm fit within this agent-based literature by modeling the firm as a type of artificial neural network (ANN). Typical uses of ANNs in economics include non-linear econometrics (Kuan & White, 1992) and game theory applications, where simple neural networks (perceptrons) are used, for example, to distinguish which strategies should be played given the history of strategies (Cho, 1994). Our approach is different and unique in that the neural network is used as a model of the firm itself. Sgroi (2005) discusses the use of neural networks for modeling boundedly rational agents in economics.

ARTIFICIAL NEURAL NETWORKS AND THE FIRM

In the 1950s, ANNs were developed as simple models of the brain.[1] The brain can be characterized as a massively parallel and decentralized information processing machine, which controls the operations of the body and sees to its normal, routine functioning. In a similar sense, the mangers of a firm serve as "the brains of the operation," and without management a firm cannot exist. In its most basic description, a neural network is a type of non-linear function that takes in a set of inputs, and applies a set of parameters (weights) and a transformation (squashing) function to produce an output. The network can be *trained* to *learn* a particular data set; training is an iterative process where the weights are adjusted via a learning algorithm to improve the performance (reduce the error) of the network over time. After the network has learned to map inputs to outputs, it is used to *generalize*—that is, to recognize patterns never seen before on the basis of previous experience.

Management scholars have documented several activities of the firm that bear close resemblance to the activities of a brain (see Simon, 1997; Cyret & March, 1963; Lawrence & Lorsch, 1986; Gailbraith, 1973):

- **Decentralized Parallel Processing:** The firm is a coordinated network of agents (nodes) which processes information in a parallel and serial manner in order to make decisions.
- **Embedded Decision Making:** Organizations have some form of a hierarchical structure. Decisions are made within decisions.
- **Learning by Experience:** Agents become proficient at activities by gaining

experience, often by a trial-and-error process.

- **Organizational Adaptation to the Environment:** Since firms face different environments, organizational adaptation implies that there is no one best way to organize.
- **Pattern Recognition and Generalizability:** As agents process information, they recognize patterns and hence the firm is able to process information that resembles patterns learned earlier.
- **Firms Contain More Knowledge than Any One Agent:** The firm is not simply a collection of agents, but rather a coordinated collection of specialized agents. Thus the knowledge of the firm is greater than the sum of the knowledge of its agents.

Our focus is on the internal structure of the network, as we study the relationship between network structure and performance. For simplicity, however, we focus on only one particular type of neural network, a backward propagation network with one hidden layer. Holding constant the type of network allows us to make changes in the *economic features* of the model to see how economic outcomes change as the size of the network changes.

Network Error and Learning

The ability of the firm-as-network-of-managers to learn the relationship between the environment and some relevant variable is a function of the number of managers.[2] The error the managers make when learning from the environment can be separated in into two types of (theoretical) errors: *approximation error* and *estimation error*.[3] If there are too few managers estimating the environmental data, then it may be that regardless of how many data points (environmental states) the managers process, the minimum error obtainable is still relatively large. This type of error is referred to as the *approximation error*. Generally speaking, as we increase the number of managers (and the number of estimation parameters—the weights), we increase the computational power of the firm and hence its ability to learn the underlying mapping between the environment and some economic variable; therefore the approximation error is decreasing as we increase the number of managers for a given dataset.

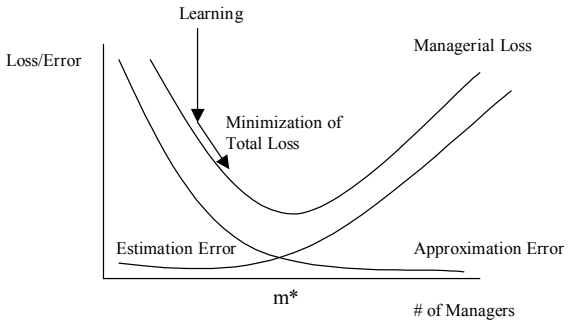
The other type of error, the *estimation error,* is due to the limited time a network has to learn a dataset of a given size. A small network has fewer weights to adjust and therefore can more quickly (i.e., in fewer iterations) find the values that minimize the error. As the network size grows, it needs more time (iterations) to search the weight space. Therefore, the estimation error is increasing in network size. In standard regression analysis, this error is analogous to the loss of degrees of freedom that we have as we increase the number of regressors.

The tension between these two errors generates a tradeoff: fewer managers will imply a larger approximation error, but also a smaller estimation error for a given number of iterations. On the other hand, reducing the approximation error by increasing the number of managers increases the number of data points needed to learn the environment, which means that for the same number of periods, the estimation error will be larger, all else equal.

The tradeoff implies that there is an "optimal" network size, m^*, defined as the dimension that makes the bound to the "managerial loss" (or error) a minimum. Figure 1 graphically shows this tradeoff.

For a very small network, the gain in approximation obtained by increasing m outweighs the errors in estimation. After a certain point, instead, the gain in approximation is not enough

Figure 1. The tradeoff between estimation error vs. approximation error



to compensate for the loss in "degrees of freedom"; as a result total loss will stop decreasing.

THE NEURAL NETWORK

The artificial neural network comprises a series of nodes (agents) that process and transmit information. We can think of each agent as a type of information processor: the agent takes in information (signals), generates a weighted sum, transforms the data via a squashing function, and then outputs a signal.[4] In our case, the squashing function is the standard sigmoid function $G(a)=1/(1+e^{-a})$, where $G(a)\in(0,1), a\in\mathbb{R}$.

The firm-as-network comprises three "layers": an input (environmental data) layer, which is information about the external environment; a hidden (management) layer, which processes information from the input layer; and an output (CEO) layer, which processes information from the hidden layer to produce a final output. Specifically, the actions of the firm each period are given by the function

$$\hat{y}(\mathbf{x}) = G\left(\sum_{j=1}^{m} w_j^o G\left(\mathbf{x}\mathbf{w}_j^h\right)\right) \equiv NN(\mathbf{x}),$$

where $\mathbf{x}\in\{0,1\}^N$ is the current state of the environment, discussed in more detail below. $\hat{y}$ is the forecasted output for the firm. $\mathbf{w}_j^h \in \Re^N$ is a vector of weights for managers $j=1,\ldots,m$, and $w_j^o \in \Re$, $j=1,\ldots,m$ are the weights for the CEO.

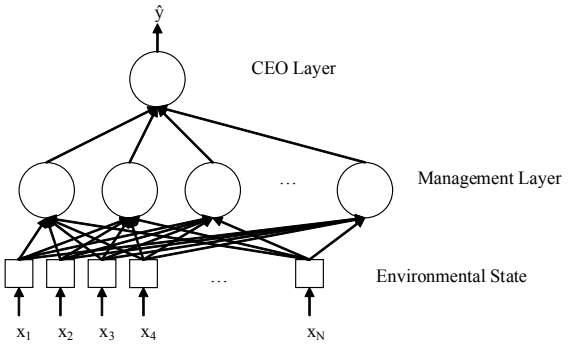
Each manager takes the dot product $\mathbf{x}\mathbf{w}_j^h$ and then applies the squashing function to produce an output, which is transmitted up to the CEO. The CEO takes a weighted sum of the managerial outputs

$$\sum_{j=1}^{m} w_j^o G\left(\mathbf{x}\mathbf{w}_j^h\right)$$

and applies the squashing function to produce a final output $\hat{y}$. In the models presented below, $\hat{y}$ can be, for example, the market price of a good, the demand intercept, or a rival's output decision.

Figure 2 gives a representation of the network. Note that we implicitly make a number of important assumptions: first the organization has a specific hierarchical structure in that managers only report to a single CEO; second, they do not communicate with each other; and finally, all managers see all environmental information.

Figure 2. An artificial neural network



The Learning Algorithm

Each period the firm compares its output forecast to an actual value to obtain the error $\varepsilon_t = (y_t - \hat{y}_t)^2$. Below, we give an economic motivation for the error term. The firm then uses a type of gradient descent algorithm to adjust the weights to minimize the error over time. Here we give a brief description of the algorithm; for more details see Barr and Saraceno (2002).

The gradient of ε with respect to the output-layer weights is

$$\frac{\partial \varepsilon}{\partial w_j^{\,o}} = (y - \hat{y})\hat{y}(1 - \hat{y}) z_j^{\,h},$$

where $w_j^{\,o}\, j=1,\ldots,m$ are the weights between the hidden layer and the output, and $z_j^{\,h}$ is the output of hidden layer agent j. Notice that we make use of a property of the sigmoid function: $G'(a) = a(1-a)$. By a similar analysis we can find the gradient of the error surface with respect to the hidden-layer weights.

The gradients serve as a basis for adjusting the weights in the opposite (negative) direction of each of the gradients:

$$w^o{}_j(t+1) = w^o{}_j(t) - \eta \frac{\partial \varepsilon}{\partial w_j^{\,o}},$$

where η is a smoothing parameter. A similar procedure is used for the hidden (management) layer. When the updating of weights is finished, the firm processes the next input pattern and repeats the weight-update process. Over successive iterations, the firm improves its performance, and the errors are reduced.

The Environment

An important consideration that has often been overlooked in models of the firm is the inclusion of the external environment facing the firm. Firms adapt to, and learn about, the events occurring around them (such as changing input prices, consumer tastes, and technology); and a firm's ability to learn directly affects its performance.

We model the environment as a set of "features" $X = \{0,1\}^N$ that may be present (one) or absent (zero). Each period, the firm is faced with a vector $\mathbf{x}_t \in X$, which represents the current state of the environment. Below, we assume the environment affects some important variable (e.g., the demand intercept) according to the following function:

$$y_t = f(\mathbf{x}_t) = \frac{1}{(2^N - 1)} \sum_{k=1}^{N} x_{kt} 2^{N-k}, \qquad (1)$$

where y_t is the variable to be learned. The functional form assumes that these features are ordered according to the size of their influence on the particular variable of interest. Equation 1 converts a vector of binary digits to its decimal equivalent, normalized to be in the $[0,1]$ interval.

Environmental Complexity

Our measure of environmental complexity is given by the value of N, the size of the environmental vector. Each period, a randomly selected vector is chosen with probability ($1/2^N$). Thus, the parameter N embodies two types of complexity: both the quantity and the frequency of information processed by the firm. If N is

small, over several iterations, the firm will, on average, view a particular vector more times than if N is large. In the simulations below, N ranges from *5* to *40* in *5* unit increments.[5]

A MODEL OF THE FIRM AS PRICE TAKER

In the sections below, we present three models of the firm as an ANN. Here we discuss the case of the firm in a perfectly competitive market. In sections 5 and 6 we expand the model to include Cournot competition.

To begin, assume we have a price-taking firm with a profit function at time t of

$$\pi_t = p(\mathbf{x}_t)q_t - \frac{c}{2}q_t^2 - \varphi(m),$$

where $p_t=p(\mathbf{x}_t)$ is the price of the good, which changes each period as the environment changes. q_t is the quantity choice, c is a variable cost parameter (normalized to one), and $\varphi(m)$ is the "wage" cost of carrying the network, where m is the number of managers in the management layer. Time subscripts are dropped for notational convenience.

Because our focus is on the effect of environmental complexity on learning (and below on strategic interaction), we set $\varphi(m)=0$ for the remainder of this chapter. This can be justified in one of two ways: either we can assume that $\varphi(m)$ is constant over the range of firm sizes we consider and therefore can be normalized to zero, or we can assume a fixed marginal cost per manager. This would have the effect of changing the quantitative results but not the qualitative results of the simulations presented below. Thus without loss of generality, we do not consider the cost to the firm of carrying a network of particular size.

If the firm knew the price of the good, it would each period choose an output level q^* to maximize its profit. In this case, optimal output is given by $q^*=p$, and optimal profit is given by[6]

$$\pi^* = \frac{1}{2}[p(\mathbf{x})]^2.$$

We assume, however, that the firm does not know the price, but rather makes an estimate based on observed environmental states. Thus each period, the firm chooses an output $q = \hat{p}$, where $\hat{p}$ is the estimated price. The firm then observes the true price and realizes a profit of

$$\pi = p\hat{p} - \frac{1}{2}\hat{p}^2. \qquad (2)$$

The firm then compares its actual profit to the optimal profit to determine its *loss*—that is, its deviation from optimal profit:

$$L = (\pi^* - \pi) = \frac{1}{2}p^2 - \left[p\hat{p} - \frac{1}{2}\hat{p}^2\right] = \left(\frac{1}{2}\right)(p - \hat{p})^2 \equiv \frac{1}{2}\varepsilon^2.$$

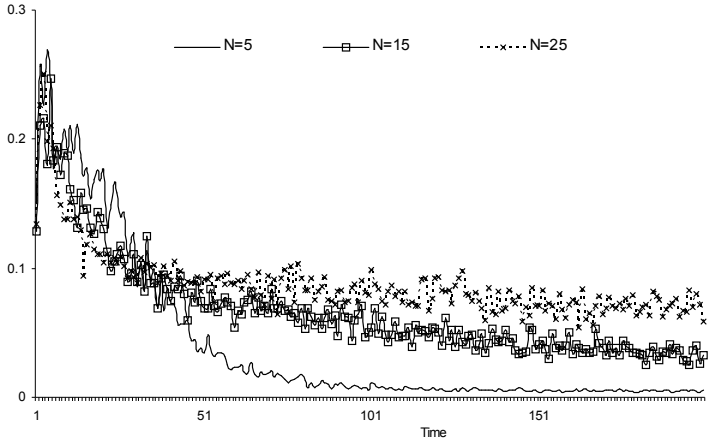
Notice that actual profit can be written in terms of deviation from maximum profit: $\pi = \pi^* - L = \pi^* - \frac{1}{2}\varepsilon^2$. Since $\hat{p}$ is the estimate made by the firm, we have per period loss:

$$L = \frac{1}{2}\varepsilon^2 = \frac{1}{2}\left(p - G\left(\sum_{j=1}^{m} w_j^o G\left(\mathbf{x}\mathbf{w}_j^h\right)\right)\right)^2. \qquad (3)$$

The firm tries to minimize its loss over time by adjusting its weights according to the learning algorithm discussed in above. Define

$$\Pi = \frac{1}{T}\sum_{t=1}^{T}\pi_t = \left(\frac{1}{T}\sum_{t=1}^{T}(\pi_t^* - L_t)\right)$$

*Figure 3. Average error vs. time for different complexity levels (*m=8*)*



as the average profit achieved over T periods. The objective of the firm then is to maximize Π given its network size.

A Simulation Experiment

In this section we present the results of a simulation experiment that investigates the relationship between the optimal size of the firm and the complexity of the environment. To perform the experiment, we do the following. For each level of environmental complexity *N=5, 10,...,40* and network size *m=2,...,20*, each period, the firm estimates a price, given that it observes the current state of the environment $\mathbf{x}_t$—that is, $\hat{p}_t = NN(\mathbf{x}_t)$.

Then it determines its profit and error by comparing its profit to the maximum profit. It uses the error to update its weights in order to improve future profits. The firm performs these steps for 200 periods. We then compare the average profit obtained for each network size.[7]

Results

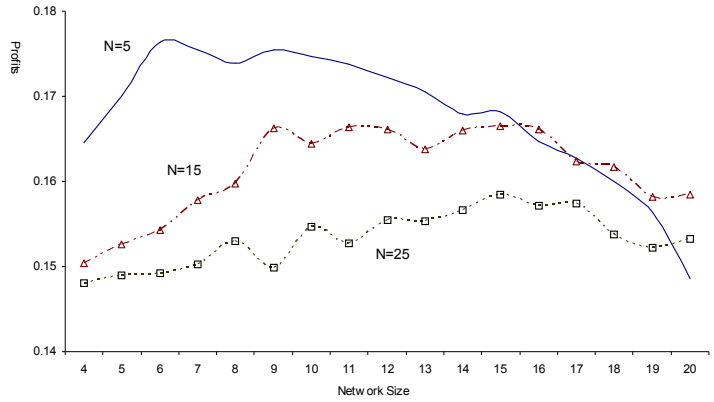
Figure 3, which is a result for a typical run, shows how the firm learns over time. We set the network size to *m=8*, and look at the error (learning) over time. This graph demonstrates organizational learning curves. As the error diminishes over time, profits will increase, all else equal, since the firm is learning to forecast closer to the correct price.

Notice also that as the environment becomes more complex, average learning ability decreases, since there is more information to be learned and the problem is more difficult to solve, holding the network size fixed.

Next in Figure 4, we show the relationship between average profits and the size of the network for three typical runs. In general we see an arc-shaped relationship: profit initially increases with network size, and then decreases after some point. This captures the tradeoff discussed earlier. The optimal balance between approximation error and estimation error is generally increasing in complexity. For a simple environment *(N=5),* the maximum profit is reached with a relatively small network. As we increase the complexity level, the maximum profit is achieved with a relatively larger network.

Figure 5 shows the relationships between optimal firm size (the network size that maxi-

Figure 4. Profits vs. network size for different complexities

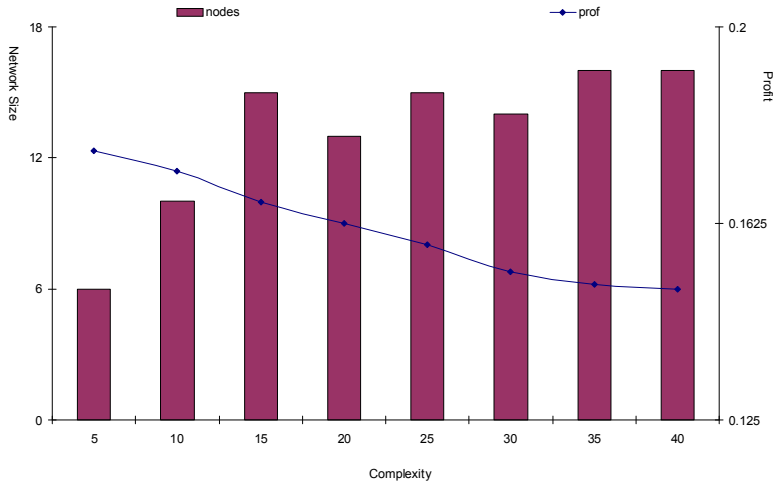


mizes average profits over 200 iterations) and profits vs. complexity. As we can see, in general, optimal firm size is increasing with complexity and profits are decreasing.

THE COURNOT MODEL

Cournot (1838) was the first to analyze strategic interaction between firms in an oligopolistic setting. For this reason his model had an enormous impact when game theory developed the tools to analyze this type of interaction.[8] In short, the model consists of two (or more) firms who set their quantities as "best responses" to their competitors' output choice; the equilibrium then consists of a set of quantities for which each oligopolist is maximizing its profit given the quantity of the others.

Figure 5. Optimal firm size and profit vs. complexity



Here, as an expansion of the perfectly competitive model, we investigate whether two neural networks competing in a Cournot game can learn to achieve the equilibrium for the game, and further how environmental complexity and firm size affect the game. We assume that the firm faces a linear, downward sloping demand:

$$p_t = \alpha(\mathbf{x}_t) - \frac{\beta}{2}(q_{1t} + q_{2t}),$$

where $\alpha(\mathbf{x}_t)$ is the intercept, which is a function of the state of the environment. Each period the two firms make an output decision, where q_i, $i=1,2,$ is the firm's output choice. (Again, time subscripts are dropped for ease of exposition.)

We assume the slope parameter β and the variable cost parameter c are constant and normalized to one. This gives a profit function of

$$\pi_i = \left(\alpha - \frac{1}{2}(q_1 + q_2)\right)q_i - \frac{1}{2}q_i^2, \; i=1,2. \qquad (4)$$

Profit maximization for the firm, given an output choice of its rival, generates a best response function:

$$q_i^{br} = \frac{1}{2}\left(\alpha - \frac{1}{2}q_{-i}\right), \qquad (5)$$

where $-i$ is the output quantity choice of firm i's rival. If we plug (5) into (4), we can derive the firm's profit function as a function of its rival's output choice

$$\pi_i^{br} = \left(q_i^{br}\right)^2. \qquad (6)$$

If both firms play the Cournot-Nash equilibrium (i.e., set each firm's best response function equal to the other), then optimal output, price, and profit, respectively, are given by:

$$q^{Cournot} = \frac{2}{5}\alpha, \quad p^{Cournot} = \frac{3}{5}\alpha, \quad \pi^{Cournot} = \frac{4}{25}\alpha^2.$$

Further, we assume that α is a function of the external environment:

$$\alpha(x) = \frac{1}{(2^N - 1)}\sum_{k=1}^{N} x_k 2^{N-k}.$$

That is to say, the intercept—the sum of those things that shift demand, such as tastes and income—is determined by the presence or absence of environmental features.

Each period, the firm observes $\mathbf{x}$ and produces an output according to $\hat{q}_i = NN_i(\mathbf{x})$, which generates a profit of

$$\pi_i = (\alpha - (\hat{q}_i + \hat{q}_{-i}))\hat{q}_i - \frac{1}{2}\hat{q}_i^2, \quad i = 1,2. \qquad (7)$$

Notice that the firm estimates an output and not the intercept; this is a convenient simplification. The knowledge of the network is contained in the weights; the firm implicitly learns the relationship between the external environment and the intercept while trying to forecast its optimal output.

Also notice that the lack of direct strategic interaction in terms of firm size does not imply that the two firms do not influence each other. In fact, the competitor's choice enters into the best response quantity of the firm and consequently affects the weight updating process. In this framework, as can be seen in equation 7, firm $-i$*'s* actions affects firm i*'s* payoffs, rather than firm i*'s* actions.

After the firm chooses an output quantity, it observes its rival's choice and then the market clearing price. Next it learns the value of α via

$$\alpha = p + \frac{1}{2}(\hat{q}_1 + \hat{q}_2).$$

Given all this information, it determines the best response that should have been played given its rival's choice of output:

$$q_i^{br} = \frac{1}{2}\left(\alpha - \frac{1}{2}\hat{q}_{-i}\right).$$

The firm then compares its actual profits (equation 7) to the best response profits from equation 6 to determine its loss:

$$L_i \equiv \left(\pi_i^{br} - \pi_i\right) = \left(q_i^{br} - \hat{q}_i\right)^2 = \varepsilon_i^{Cournot}.$$

Over repeated estimations the firms use their errors to adjust their weights to improve their performance over time via the learning algorithm discussed above.

Network Size Equilibria

In addition to strategic interaction due to demand, the two firms face strategic interaction in regards to choice of network size. A rival's choice of particular network size influences its performance, which in turn affects the performance of its rival. This gives rise to the concept of a network size equilibrium (NSE), which is a choice of network size for each firm such that, given the rival's size choice, it has no incentive to switch the number of agents.

Formally we define an NSE as the pair $\{m_1^*, m_2^*\}$ such that

$$\Pi_i\left(m_i^*, m_{-i}^*\right) \geq \Pi\left(m_i, m_{-i}^*\right), \quad i = 1,2,$$

where

$$\Pi_i = \frac{1}{T}\sum_{t=1}^{T}\pi_{it}\left(m_i, m_{-i}\right).$$

Here we ask the questions: Does at least one NSE exist for each complexity value? And what is the relationship between complexity and the network size equilibrium? We focus on the equilibria that exist after T periods, and do not have endogenous dynamics with the number of managers—that is, we do not examine firms changing the number of managers during the learning process. Rather we conduct a kind of comparative statics exercise, whereby we look at the NSEs that exist for given environmental conditions.

In this experiment, networks of different sizes, $m_1, m_2 \in \{2,\ldots,20\}$, compete against each other for T=200 iterations.[9] This competition is run for each $N = 5,10,\ldots,40$. Then, for each N, we see if one (or more) NSE exists.

Results

Figure 6 shows an example of how the firm learns to achieve the equilibrium outcome in the Cournot game. The top line is the ratio of the actual market price to the Nash-equilibrium price. The bottom line is the absolute value of the difference between the firm's actual output choice and the Cournot best response. With both time series, the firm converges to the Cournot-Nash equilibrium value.

Figure 7 shows an example of the strategic interaction effect in terms of network size. The figure shows an example of the profit obtained by each firm, holding firm two's network constant (m_2=8) and varying firm one's network size (fixing N=10). In general, as firm one increases its size, it improves its performance until some point, after which performance decreases (for such a low level of complexity, the best performance is obtained for quite a small size, m =5). Firm 2's performance is also affected by firm 1's size. As we increase the latter, firm 2's performance steadily improves and surpasses firm 1 (profit will obviously be equal when the two firms have equal size). This is due to the fact that an improved performance

Figure 6. Abs(q_1-q_1^{br}) and price/price vs. time (m_1=m_2=4, N=5)*

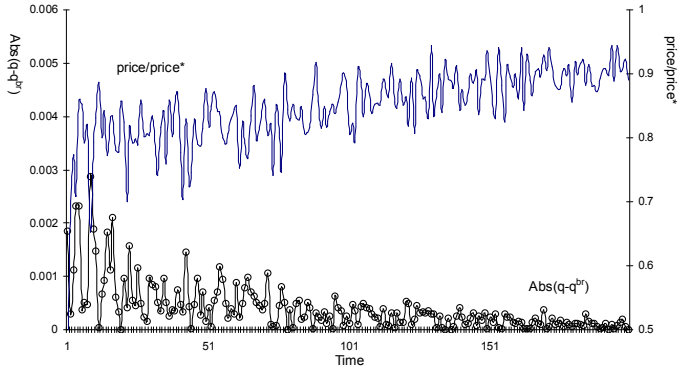
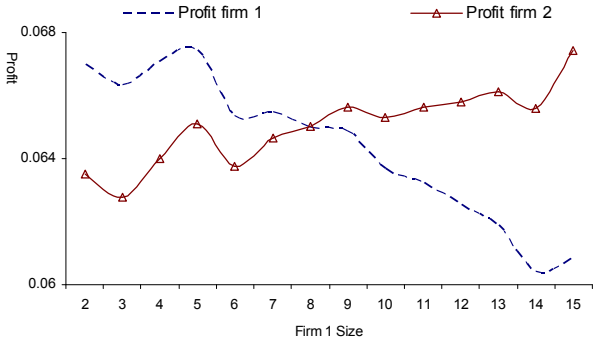


Figure 7. Average profit vs. firm 1's network size (m_2=8, N=10)



for firm 1 improves performance for firm 2, but then at some point, firm 1 becomes "too large" and firm 2 develops a relative advantage.

Figure 8 shows the average network sizes for the firms at their network size equilibria. For all values of *N*, there is always at least one NSE, and in several cases there are more. In fact, the average number of NSEs per *N* is 2.9, with a standard deviation of 1.1. There does not appear to be any relationship between complexity and number of equilibria. As we can see in Figure 8, similar to Figure 5, average network size is increasing in complexity and average profit is decreasing.

COOPERATION

To what extent can two competing networks learn to collude in their output choices? Here we fix the type of strategy used by each firm as a type of tit-for-tat strategy and investigate under what conditions collusion is possible. Since the Cournot game fits within a repeated prisoner's dilemma (RPD), we use the RPD framework to investigate collusion between two neural networks.

To begin, assume the two firms were to collude and act like a monopolist. Then, demand is

Figure 8. Average "industry" size equilibria and profit vs. complexity

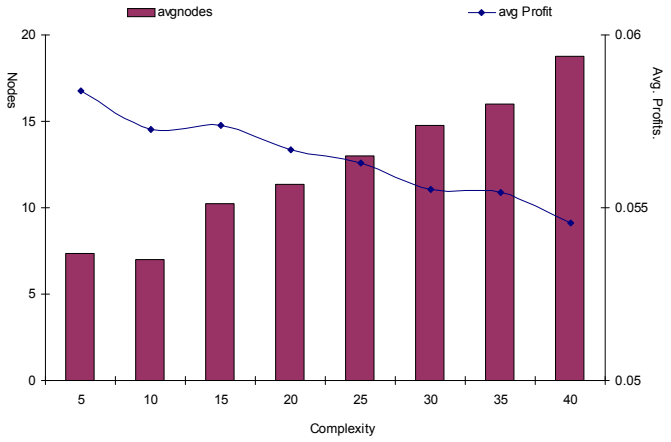


Table 1. Cournot-prisoner's dilemma game

		Player 1	
		Cooperate (C)	**Defect (D)**
Player 2	**Cooperate (C)**	$0.125\alpha^2, 0.125\alpha^2$	$0.141\alpha^2, 0.191\alpha^2$
	Defect (D)	$0.191\alpha^2, 0.141\alpha^2$	$0.167\alpha^2, 0.167\alpha^2$

$$p = \left(\alpha - \frac{1}{2}Q\right),$$

where Q is the joint monopoly output level. The corresponding profit function is

$$\pi = \left(\alpha - \frac{1}{2}Q\right)Q - \frac{1}{2}Q^2 = \alpha Q - Q^2.$$

Solving for the optimal quantity then yields

$$Q^{monop} = \frac{\alpha}{2}, \quad p^{monop} = \left(\frac{3\alpha}{4}\right), \quad \pi^{monop} = \frac{\alpha^2}{4}.$$

Assume the two firms decide to split the market and each produce

$$q_i^{monop} = \frac{\alpha}{4}, i = 1,2,$$

and get a profit of

$$\pi_i^{monop} = \frac{\alpha^2}{8}$$

On the other hand, each firm could "cheat" by producing more than the shared monopoly output, on the assumption that its rival will hold to the agreement. This would yield a higher profit for the firm.

The choice of whether to collude or not generates a simple one-shot *2x2* game, a Prisoner's Dilemma (see Table 1). We assume that the cooperative strategy for both firms is the shared monopoly output, and the defection

strategy is to play a best response to the rival's output choice. In this case, if the rival holds to the agreement, the firm will play a Cournot best response to the rival's shared monopoly output choice.

In this game the defect-defect outcome is the only equilibrium. However, it has been shown that in a repeated framework, when the game is not likely to end soon, there are an infinite number of equilibria (Fudenberg & Tirole, 1991). As Axelrod (1984) discusses, a simple yet powerful strategy is tit-for-tat (TFT), which says a player should start cooperating and then play the same strategy that its rival played last period.

In this chapter, we have firms employ a slight variation of the TFT strategy to see if two networks can learn to cooperate over time. More specifically the firm chooses an output each period based on the following rule:

$$q_i = \left\{ \begin{array}{c} \frac{1}{2}\left(\hat{\alpha}_i - \frac{1}{2}\hat{q}^i_{-i}\right) \text{ if } \left(\hat{q}^i_{-i} - \frac{\hat{\alpha}_i}{4}\right) > \rho_i \\ \frac{\hat{\alpha}_i}{4} \text{ otherwise} \end{array} \right\} (i=1,2), \tag{8}$$

where q_i is firm *i's* output, $\hat{q}^i_{-i}$ is firm *i's* estimate of its rival's output, and $\hat{\alpha}_i$ is firm *i's* estimate of α. Equation 8 says that if the firm estimates its rival to be a cheater—that the rival is expected to deviate from forecast monopoly profit—then it plays the optimal forecasted Cournot output; that is, it defects as well.

Unlike earlier sections, here firms estimate both the demand intercept and their rival's output quantity each period. That is, the neural network has two outputs; it uses this information to decide whether to defect or not.[10] Similar to earlier sections, the external environment shifts the demand intercept.

The threshold value $\rho_i \geq 0$ represents the firm's "willingness to be nice." For relatively small values, such as $\rho_i = 0$, firm *i* will play defect relatively more often; for values $\rho_i \geq \ddot{\rho}$, the firm will be so nice that it will never defect. Notice that in making its decision whether to defect or not, the firm has two possible sources of error: the first is the environment, and the second is the opponent's quantity; this is why it will allow a deviation ρ_i from the monopoly output before reverting to the non-cooperative quantity.

The niceness parameter is important for several reasons. Axelrod (1984), for example, shows that the "personality" of a player's strategy is an important determinant of the sustainability of cooperation. In addition, Henrich et al. (2001) show that players' willingness to cooperate and reciprocate varies widely across societies and is deeply embedded in the cultures of the players.

The error terms are given by:

$$\varepsilon_{1i} = \left(\hat{\alpha}_i - \alpha\right)^2$$

$$\varepsilon_{2i} = \left\{ \begin{array}{c} \left(\hat{q}^i_{-i} - q_{-i}\right)^2 \text{ if} \left(q_{-i} - \frac{\alpha}{4} > \rho_i\right) \\ \left(\hat{q}^i_{-i} - \frac{\alpha}{4}\right)^2, \quad \text{otherwise} \end{array} \right\},$$

where q_{-i} is the rival firm's output choice. This gives us a loss of $L_i = \varepsilon_{1i} + \varepsilon_{2i}$. The firm uses this loss to adjust its weights each period according to the learning algorithm described above.

We measure cooperation as follows. Let

$$c_{it} = \left\{ \begin{array}{l} 1 \text{ if firm } i \text{ plays } C \text{ in period } t \\ 0 \text{ otherwise} \end{array} \right\}$$

Figure 9. Firm 1's cooperation rates over time (m_1=m_2=8, N=10)

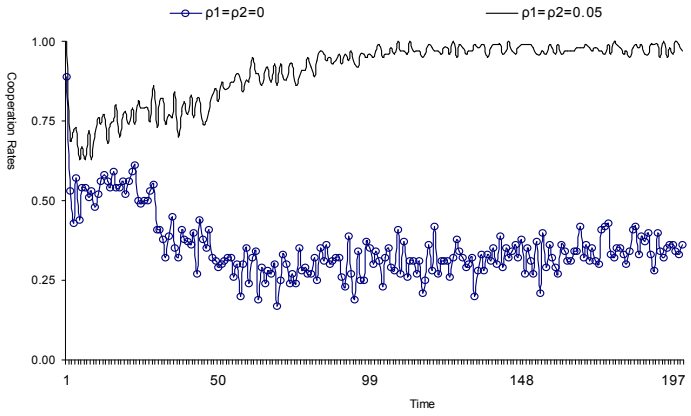
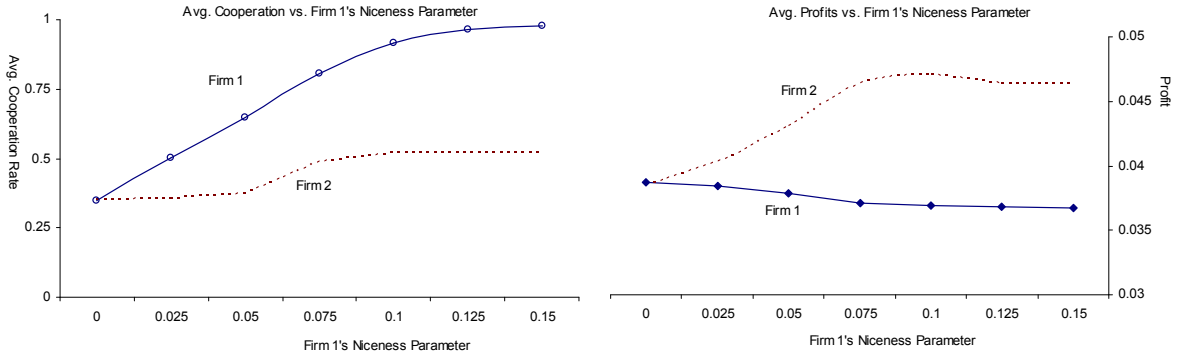


Figure 10, Cooperation and profits vs. firm 1's niceness parameter



Then over R runs we take the *average cooperation rate* for each period as

$$\bar{c}_{it} = \frac{1}{R}\sum_{r=1}^{R} c_{itr} \ .$$

Results

Figure 9 gives a demonstration of how niceness affects cooperation. The figure plots firm 1's cooperation rates over time, when both firms have a niceness level of 0.0 and 0.05. When both firms have $\rho_1 = \rho_2 = 0$, cooperation rates steadily decline and then reach a minimum level of about 0.35. On the other hand, if the two firms are sufficiently nice, $(\rho_1 = \rho_2 = 0.05)$, then cooperation rates steadily increase over time.

Next in Figure 10, we show how niceness affects performance, when firm 2 holds its niceness parameter constant and firm 1 changes its parameter. Notice how we see a slight increase in firm 2's cooperation rate, however firm 1's profits fall slightly while firm 2's increase.

Table 2. Regression results. Dep. var.: 100 avg. cooperation of firm 1. Robust standard errors given. All coefficients are statistically significant at greater than 99% level.*

Variable	Coefficient	Std. Error	Variable	Coefficient	Std. Error
ρ_1	972.1	6.8	$m_1*\rho_1$	0.685	0.116
ρ_2	432.5	6.6	$m_2*\rho_1$	-3.99	0.12
$\rho_1*\rho_2$	-1012.0	14.7	$N*\rho_1$	0.807	0.056
ρ_1^2	-6219.6	97.8	$N*\rho_2$	-0.767	0.056
ρ_2^2	-2940.6	99.6	*Constant*	31.1	0.3
ρ_1^3	16364.5	428.6	m_1	-0.156	0.025
ρ_2^3	8292.4	434.9	m_2	1.516	0.025
N	-0.250	0.013	m_1^2	0.004	0.001
N^2	0.001	0.000	m_2^2	-0.044	0.001
# Obs.	10,000				
R^2	0.962				
Adj. R^2	0.962				

To investigate the relationship between cooperation and complexity and firm size, we generated for each complexity level $N=\{5,10,\ldots,40\}$, 1,250 random draws of niceness values, $\rho_1, \rho_2 \in [0,0.15]$, and firm sizes, $m_1, m_2 \in \{2,\ldots,20\}$. We let the firms compete for 200 iterations and recorded whether firm 1 cooperated or not each period. For each draw of variables, we then took for firm 1 the average cooperation rates of 30 runs as our dependant variable.

Table 2 shows the coefficients of a regression of firm one's average cooperation rate as a function of niceness values, firm sizes, and environmental complexity. The regression shows that there is, in general, a non-linear relationship between the exogenous variables and cooperation rates. In particular, increasing complexity is associated with lower cooperation rates, all else equal (for all values of ρ_1 and ρ_2). Roughly speaking, an increase in N by 5 is associated with about a 0.9 decrease in cooperation rates, plus or minus 0.01.[11]

CONCLUSION

This chapter has presented three models of the firm as an artificial neural network. The first model has the firm in a price-taking situation. The objective of the network is to learn the mapping between environmental signals and the price. We demonstrate how the network learns over time, and that, on average, optimal firm size is increasing and profits decreasing as environmental complexity increases. The optimal firm size is one that best balances the tradeoff between approximation error (error due to less than infinite network size) and estimation error (error due to limited time to learn).

The following model has two firms competing in a Cournot game. We demonstrate how two firms can learn to converge to the Cournot-Nash equilibrium. In addition, we show the existence of strategic interaction with firm size: a firm's choice of network size affects its rival's performance. We show that the average network size equilibria are increasing with environmental complexity.

Our last model builds on the Cournot model by exploring cooperation between two networks. Firms play a repeated Prisoner's dilemma type game with a variant of a tit-for-tat strategy. We show that firms can learn to collude in their output choices, and near-perfect cooperation is feasible if firms are sufficiently "nice." In addition we show via regression analysis how cooperation rates are determined by not only the niceness of the players,

but also the firm sizes and the complexity of the environment.

Though space precludes a more detailed treatment, we would like to briefly mention the relationship between our models and competition policy, which is also an area for further research. The traditional task of competition policy has generally been to prevent price raising/quantity restricting policies, and to promote technological innovation that might be suboptimally provided without sufficient government incentives (Pepall et al., 2004).

Our models depict a more complex picture: they show that firms may learn to restrict their output simply by indirect signals of their willingness to do so, rather than by any overt collusion. But, the ability to learn to restrict output is directly tied to the difficulty of the learning problem.

Thus, on one hand, ease of learning can improve efficiency, as prices charged are closer to profit maximizing prices; but learning can also reduce global welfare when it leads to increased collusion and relatively low output. So the nature of the environment can present a double-edged sword: simple environments increase efficiency but also the likelihood of output restrictions and collusion. Government policy, for example, has not so much to control mergers, but rather needs to act in a more comprehensive way to "break" such a tradeoff, keeping in mind that environmental complexity plays a crucial role in defining the tradeoff and offering possible solutions to it.

Our models also demonstrate that when learning and environmental interaction are considered, firm size *per se* may not be indicative of monopoly power. In fact, even in a perfectly competitive market, large firms may emerge simply due to optimal adaptation to the environment, and *vice versa*, oligopolistic structures may be associated with low firm size.

Our framework attempts to blend the traditional concerns of industrial organization—price and quantity decisions—with a new approach to investigate the internal features of the firm that might affect these decisions. Our focus has been on two things in particular: (1) the capability of a network to improve its performance in regards to a particular set of economic decisions and environmental characteristics, and (2) how the size of the networks influences and is influenced by the economic environment in general. The models presented here are but one attempt to build on general economic principles. Though concrete modeling choices for future work will have to depend on the features of the problem studied, it is our belief that these choices should take into account the complex relationship between firm structure and the environment, a view generally overlooked by economic theory.

REFERENCES

Axelrod, R. (1984). *The evolution of cooperation.* New York: Basic Books.

Barr, J., & Saraceno F. (2002). A computational theory of the firm. *Journal of Economic Behavior and Organization, 49,* 345-361.

Barr, J., & Saraceno F. (2004). *Organization, learning and cooperation.* Newark Working Paper #2004-001, Rutgers University, USA.

Barr, J., & Saraceno F. (2005). Cournot competition, organization and learning. *Journal of Economic Dynamics and Control, 29,* 277-295.

Burns, T., & Stalker, G. M. (1994). *The management of innovation* (revised ed.). New York: Oxford University Press.

Chandler, A. D. (1980). *The visible hand: The managerial revolution in American business.* Cambridge: Belknap Press of Harvard University Press.

Chang, M.-H., & Harrington, J. (2006). Agent-based models of organizations. *Handbook of computational economics, volume 2* (pp. 273-1337). New York: Elsevier.

Cho, I.-K. (1994). Bounded rationality, neural networks and folk theorem in repeated games with discounting. *Economic Theory, 4,* 935-957.

Coase, R. H. (1937). The nature of the firm. *Economica N. S., 4,* 386-405.

Cournot, A. A. (1838). *Researches into the mathematical principles of wealth* (translation: N. T. Bacon, 1960). New York: A.M. Kelly.

Croall, I. F., & Mason, J. P. (1992). *Industrial applications of neural networks: Project ANNIE handbook.* New York: Springer-Verlag.

Cyert, R. M., & March, J. G. (1963). *A behavioral theory of the firm.* Englewood Cliffs, NJ: Prentice-Hall.

Freeman, J. A. (1994). *Simulating neural networks with mathematica.* New York: Addison-Wesley.

Fudenberg, D., & Tirole, J. (1991). *Game theory.* Cambridge, MA: The MIT Press.

Galbraith, J. (1973). *Designing complex organizations.* Reading, MA: Addison-Wesley.

Henrich, J., Boyd, R., Bowles, S., Camerer, C., Fehr, E., Gintis, H., & McElreath, R. (2001). In search of homo economicus: Behavioral experiments in 15 small-scale societies. *American Economic Review, 91*(2), 73-78.

Kuan, C.-M., & White, H. (1994). Artificial neural networks: An econometric perspective. *Econometric Reviews, 13,* 1-91.

Lawrence, P. R., & Lorsch, J.W. (1986). *Organization and environment* (revised ed.). Boston: Harvard Business School Press.

Nelson, R. R., & Winter, S. G. (1982). *An evolutionary theory of economic change.* Cambridge: Belknap Press of Harvard University Press.

Niyogi, P. (1998). *The informational complexity of learning.* Boston: Kluwer Academic.

Peppal, L., Richards, D. J., & Norman, G. (2004). *Industrial organization: Contemporary theory and practice* (3rd ed.). New York: South-Western College Publishers.

Sgroi, D. (2005). Using neural networks to model bounded rationality in interactive decision making. In D. J. Zizzo (Ed.), *Transfer of knowledge in economic decision making.* New York: Palgrave Macmillan.

Simon, H. A. (1997). *Administrative behavior: A study of decision-making processes in administrative organizations* (4th ed.). New York: The Free Press.

Skapura, D. M. (1996). *Building neural networks.* New York: Addison-Wesley.

Vidyasagar, M. (1997). *A theory of learning and generalization.* New York: Springer.

Williamson, O. E. (1985). *The economic institutions of capitalism.* New York: The Free Press.

ENDNOTES

1 In general, ANNs have been used for a wide array of information processing applications. Two examples include pattern recognition and forecasting. For a more detailed treatment of neural networks, see Croall and Mason (1992), Freeman (1994), and Skapura (1996).

2 Clearly, the quality of management is at least as crucial as the quantity; however,

we do not tackle this issue in this chapter.

[3] This section discusses the concepts of Computational Learning Theory as discussed in Niyogi (1998) and Vidyasagar (1997).

[4] We can also think of each agent, loosely speaking, as a type of voter: the agent weighs the various input signals and then outputs a "vote," which is a value between 0 and 1.

[5] Notice our measure of environmental complexity, in effect, merges the "training" and "generalization" process into one step.

[6] In the standard perfect competition model, entry would bid the price down to zero since there are no fixed costs. Our simulations have the expected value of price equal to 0.5. We can justify this on the grounds that either entry is limited over the time frame under consideration or long-run average total cost is 0.5, but we ignore the fixed-cost component since it does not materially affect our simulation results.

[7] All simulations were done in Mathematica 3.0. Initial weights are generated from a Uniform[-1,1] distribution. In addition, for each $\{N, m\}$ pair, we repeat each simulation run *100* times and take averages to smooth out random fluctuations. Finally, we set the adjustment parameter $\eta=10$ for all simulations in the chapter.

[8] For a description of the literature on learning in Cournot models, see Barr and Saraceno (2005).

[9] Similar as above, for each triple of $\{m_1, m_2, N\}$, we generate 100 runs and take averages to smooth out fluctuations in each run.

[10] Having two outputs entails only a slight variation in the structure of the network and the learning rule. For more information, see Barr and Saraceno (2004).

[11] Barr and Saraceno (2004) explore equilibria in network size and niceness values. As expected, network size is increasing in complexity, but there appears to be no relationship between niceness and complexity in equilibria. For the sake of brevity, we do not present the results here.

Chapter XX
Evolutionary Modeling and Industrial Structure Emergence

Halina Kwasnicka
Wroclaw University of Technology, Poland

Witold Kwasnicki
University of Wroclaw, Poland

ABSTRACT

In the first part of the chapter, an outline of the evolutionary model of industrial dynamics is presented. The second part deals with a simulation study of the model focused on identification of necessary conditions for emergence of different industrial strictures. Textbooks of traditional economics distinguish four typical industry structures and study them under the names of pure competition, pure monopoly, oligopoly, and monopolistic competition. Variations in behavior modes of differently concentrated industries ought to be an outcome of the cooperation of well-understood evolutionary mechanisms, and not the result of juggling differently placed curves representing supply, demand, marginal revenue, marginal cost, average costs, and so forth. Textbook analysis of industrial structures usually omits influence of innovation on market behavior. Evolutionary approach and simulation allow for such analysis and through that allow enriching the industrial development study. One of the important conclusions from this chapter is that evolutionary analysis may be considered as a very useful and complementary tool to teach economics.

INTRODUCTION

Almost all evolutionary economics (on EE foundations, see Dopfer, 2005) models worked out in past decades are dynamical ones and are focused on far-from-equilibrium analysis. There is no place to review and to characterize evolutionary models in economics in detail.[1] In a

nutshell the other main features of evolutionary models may be summarized as follows:

- development seen in historical perspective; macro-characteristics flow from aggregation of micro-behaviors of economic agents;
- population perspective;
- diversity and heterogeneity of behavior;
- search for novelties (innovation), hereditary information;
- selection which leads to differential growth; and
- spontaneity of development.

Some of those features seem to be crucial to call a model an evolutionary one, in our opinion to those crucial features belong: diversity and heterogeneity of economic agents (firms) and their behavior, search for innovation based on a concept of hereditary information (knowledge), and selection process which leads to diversified rate of growth and spontaneity of development. Heterogeneity and variety can therefore be considered as an important characteristic of evolutionary approaches to technological change (Nelson, 1995; Saviotti, 1996). An interesting question in relation to economic evolutionary models is presence of decision-making procedures. In many models that procedure is not present; in many others it has a more or less complicated form.

In the remaining part of this chapter, we outline an evolutionary model[2] and present a selection of current simulation results of that model. The main aim of this chapter is to show that evolutionary modeling can be used not only as an efficient research tool in economic analysis, but also as supporting tool in the economic education.

THE EVOLUTIONARY MODEL OF INDUSTRIAL DYNAMICS

The model describes the behavior of a number of competing firms producing functionally equivalent products. The decisions of a firm relating to investment, price, profit, and so forth are based on the firm's evaluation of behavior of other, competing firms, and the expected response of the market. The firm's knowledge of the market and knowledge of the future behavior of competitors is limited and uncertain. Firms' decisions can thus only be suboptimal. All firms make the decisions simultaneously and independently at the beginning of each period (e.g., once a year or quarter). After the decisions are made, the firms undertake production and put the products on the market. The quantities of different firms' products sold in the market depend on the relative prices, the relative value of products' characteristics, and the level of saturation of the market. In the long run, a preference for better products, namely those with a lower price and better characteristics, prevails.

Each firm tries to improve its position in the industry and in the market by introducing innovations in order to minimize the unit costs of production, maximize the productivity of capital, and maximize the competitiveness of its products on the market.

Simulation of industry development is done in discrete time in four steps:

1. search for innovation (i.e., search for new sets of routines which potentially may replace the old set currently employed by a firm);
2. firms' decision-making process (calculation and comparison of investment, production, net income, profit, and some other

characteristics of development which may be attained by employing the old and the new sets of routines; decisions of each firm on: (a) continuation of production by employing old routines or modernizing production, and (b) opening (or not) of new units);

3. entry of new firms; and
4. selling process (market evaluation of the offered pool of products; calculation of firms' characteristics: production sold, shares in global production and global sales, total profits, profit rates, research funds, etc).

The Search for Innovation

The creative process is evolutionary by nature, and as such its description should be based on a proper understanding of the hereditary information (see Kwasnicki, 1996, Chapter 2). According to the tradition established by Nelson and Winter (1982), we use the term *routine* to name the basic unit of the hereditary information of a firm. The set of routines applied by the firm is one of the basic characteristics describing it. In order to improve its position in the industry and in the market, each firm searches for new routines and new combinations of routines to reduce the unit costs of production, increase the productivity of capital, and improve the competitiveness of its products in the market. Nelson and Winter (1982, p. 14) define routines as "regular and predictable behavioral patterns of firms" and include in this term such characteristics as "technical routines for producing things … procedures of hiring and firing, ordering new inventory, stepping up production of items in high demand, policies regarding investment, research and development, advertising, business strategies about product diversification and overseas investment." A large part of research activity is also governed by routines. "Routines govern choices as well as describe methods, and reflect the facts of management practice and organizational sociology as well as those of technology" (Winter, 1984).

Productivity of capital, unit costs of production, and characteristics of products manufactured by a firm depend on the routines employed by the firm (examples of the product characteristics are reliability, convenience, lifetime, safety of use, cost of use, quality and aesthetic value).

We assume that at time *t,* a firm is characterized by a set of routines actually employed by the firm. There are two types of routines: *active* routines are employed by this firm in its everyday practice, and *latent* routines are stored by a firm but are not actually applied. Latent routines may be included in the active set of routines at a future time. The set of routines employed by a firm may evolve. There are four basic mechanisms for generating new sets of routines, namely: *mutation*, *recombination*, *transition,* and *transposition.*

The probability of discovering a new routine (mutation) depends on the research funds allocated by the firm for autonomous research, that is, in-house development. It is assumed that routines mutate independently of each other. The scope of mutation also depends on funds allocated for in-house development. The firm

Figure 1. Routines transition

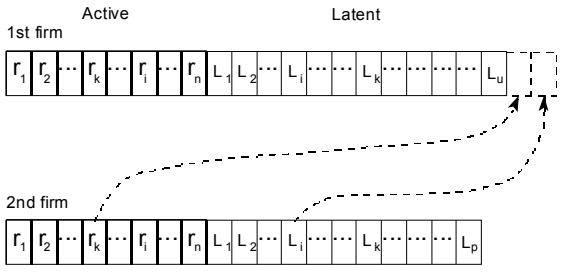
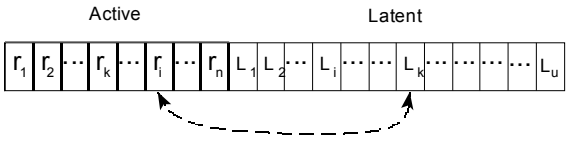


Figure 2. Routines transposition



may also allocate some funds for gaining knowledge from other competing firms and try to imitate some routines employed by competitors (recombination). A single routine may be transmitted (*transition*; see Figure 1) with some probability from firm to firm. It is assumed that after transition, a routine belongs to the subset of latent routines. At any time a random *transposition* of a latent routine to the subset of active routines may occur (see Figure 2). It is assumed that the probabilities of transition of a routine from one firm to another and the probabilities of transposition of a routine (from a latent to an active routine) are independent of R&D funds, and have the same constant value for all routines.

In general, the probability of transposition of a routine for any firm is rather small. But randomly, from time to time, the value of this probability may abruptly increase, and very active processes of search for a new combination of routines are observed. This phenomenon is called recrudescence. *Recrudescence* is viewed as an intrinsic ability of a firm's research staff to search for original, radical innovations by employing daring, sometimes apparently insane ideas. This ability is connected mainly with the personalities of the researchers, and random factors play an essential role in the search for innovations by recrudescence, so the probability of recrudescence is not related to R&D funds allocated by a firm to 'normal' research.

As a rule, mutation, recombination, and transposition on a normal level (that is, with low probabilities in long periods) are responsible for small improvements and, during the short periods of recrudescence, for the emergence of radical innovations.

Firm's Decisions

It seems that one of the crucial problems of contemporary economics is to understand the process of decision making. Herbert Simon states that "the dynamics of the economic system depends critically on just how economic agents go about making their decisions, and no way has been found for discovering how they do this that avoids direct inquiry and observations of the process" (Simon, 1986, p. 38).

The background of the decision-making procedure adopted in the model is presented in detail in Kwasnicki (1996). It is assumed that each firm predicts future development of the market (in terms of future average price and future average product competitiveness); and on the basis of its expectations on future market development and expected decisions of its competitors, each firm decides on price of its products, investment, and quantity of production that it expects to sell on the market. Each firm also considers current investment capability and the possibility of borrowing.

Price, production, and investment are set by a firm in such a way that some objective function is maximized. Contrary to the neoclassical assumption, it is not a maximization in the strict sense. The estimation of values of the objective function is not perfect and is made for the next year only. In other words, it is not a global, once and for all, optimization, but rather an iterative process with different adjustments taking place from year to year.

We assume that firms apply the following objective function:

$$O_i(t+1) = (1-F_i)\frac{\Gamma_i(t+1)}{\Gamma(t)} + F_i\frac{Q_i^s(t+1)}{QS(t)},$$

$$F_i = a_1 \exp(-a_2 \frac{Q_i^s(t+1)}{QS(t)}) \qquad (1)$$

where F_i is the magnitude coefficient (with values between 0 and 1), Q_i the supply of firm i, Γ_i the expected income of firm i at $t+1$ (defined by equation 2), QS is the global production of the industry in year t,and Γ the global net income of all firms in year t. $\Gamma(t)$ and $QS(t)$ play the role of constants in equation and ensure that the values of both terms in this equation are of the same order; a_1 and a_2 are parameters.

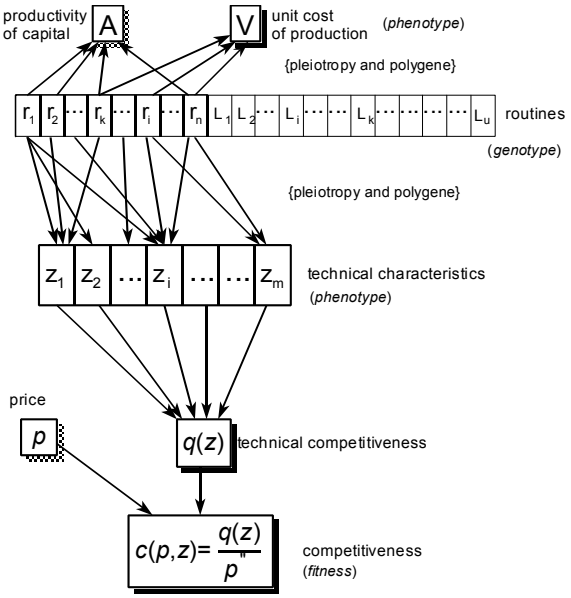
The expected income of firm i ($э_i$) is defined as:

$$\Gamma_i = Q_i^s(t)(p_i(t) - V_i v(Q_i^s(t)) - \eta), \qquad (2)$$

where V is unit production costs, $v(Q)$ is the factor of unit production cost as a function of the scale of production (economies of scale), η is the econstant production cost.

The function O_i expresses short- and long-term thinking of firms during the decision-making process (the first and second terms in equation 1, respectively). Plausible values for the parameters are $a_1 = 1$ and $a_2 = 5$, implying that the long run is much more important for survival and that firms apply a flexible strategy, namely the relative importance of short- and long-term components change in the course of a firm's development (the long-term one is much more important for small firms than for the large ones).

Figure 3. From routines to competitiveness, productivity of capital and unit cost of production—from genotype to phenotype



Products Competitiveness on the Market

The productivity of capital, variable costs of production, and product characteristics are the functions of routines employed by a firm (see Figure 3). Each routine has multiple, pleiotropic effects—that is, may affect many characteristics of products, as well as productivity, and the variable costs of production. Similarly, the productivity of capital, unit costs of production, and each characteristic of the product can be a function of a number of routines (polygeneity). We assume that the transformation of the set of routines into the set of product characteristics is described by m functions F_d,

$$z_d = F_d(r), \qquad d = 1,2,3,..,m, \tag{3}$$

where z_d is the value of characteristic d, m the number of product characteristics, and r the set of routines. It is assumed that the productivity of capital $A(r)$ and the unit cost of production $V(r)$ are also functions of the firm's routines, where these functions are not firm specific and have the same form for all firms.

An attractiveness (competitiveness) of the product on the market depends on the values of the product characteristics and its price. The competitiveness of products with characteristics z and price p is equal to:

$$c(p,z) = \frac{q(z)}{p^{\alpha}}, \qquad z = (z_1, z_2, ..., z_m), \tag{4}$$

where $q(z)$ is the technical competitiveness, z a vector of product characteristics, and α price elasticity.

In the presence of innovation, technical competitiveness varies according to the modification of routines made by each firm, or because of introducing essentially new routines. Technical competitiveness is an explicit function of product characteristics. As explained above, each routine does not influence the product's performance directly, but only indirectly through the influence on its characteristics. We assume the existence of a function q enabling calculation of technical competitiveness of products manufactured by different firms. We say that q describes the adaptive landscape in the space of product characteristics. In general, this function depends also on some external factors, varies in time, and is the result of co-evolution of many related industries.

All products manufactured by the entrants and incumbents are put on the market, and all other decisions are left to buyers; these decisions primarily depend on the relative values of competitiveness of all products offered, but quantities of products of each firm offered for sale are also taken into account.

The dynamics of industry development depend also on so called replicator (selection) equation, imposing that the share of firm i in global output increases if the competitiveness of its products is higher than the average of all products present on the market, and decreases if the competitiveness is lower than the average. The rate of change is proportional to the difference between the competitiveness of products of firm i and the average competitiveness.

SIMULATION OF INDUSTRY DEVELOPMENT

Textbooks of traditional economics distinguish four typical industry structures and study them under the name of pure (perfect) competition, monopoly, oligopoly, and monopolistic competition.[3] To explain how prices and profits are formed in the typical industries, traditional economics uses such notions as: demand and supply functions, marginal cost, average total cost, average variable cost, average fixed cost, marginal revenue, total revenue, and so on. Usually, each typical situation is considered separately in different chapters. Reading these chapters and looking at diagrams supporting the reasoning, one may get the impression that different mechanisms are responsible for the development of industries with different concentrations. It seems that the study of industry behavior at different concentrations ought to be based on an understanding of the development mechanisms that are essentially invariable and do not depend on current industry conditions, particularly on the actual number of competitors. Variations in behavior modes of differently concentrated industries ought to be an outcome of the cooperation of well-understood mechanisms of development, and not the result of

juggling differently placed curves representing supply, demand, marginal revenue, marginal cost, average total cost, average variable cost, average fixed cost, and many other variables. We do not claim that the findings of traditional economics flowing from the analysis of *curves placement* are wrong; quite the contrary, they are in accord with real phenomena, but does such analysis explain anything?

To prove that the long-run profit is equal to zero for a perfect competition market, the traditional economic theories assume an infinite number of competitors in the market. In reality, as in our simulation, the number of competitors may be only finite, but we may expect that for a reasonably large number of competitors, the results will be very close to the theoretical predictions. How many firms may be treated, from a practical point of view, as *the infinite number of competitors*? Some characteristics of the industry at the equilibrium state obtained in a series of experiments with a different number of competitors, under additional assumptions that the initial size of all firms is the same (that is, equi-partition of the market is assumed) and that the size of the market is constant (that is, $\gamma = 0$), are presented in Table 1.

The controlling variable in the series of experiments is the number of competitors. The results presented in Table 1 are the outcome of the co-working of the same mechanisms of development embedded in the model described in the previous section. The results are grouped into two parts: for the normal rate of return, ρ equal to zero, and for the rate ρ equal to 5%. Our normal rate of return corresponds, in some way, to the normal profit embedded in the neoclassical supply function. The value of the normal rate of return may be considered as an effect of the development of the whole economy, and for any single industry may be treated as exogenous. In any real processes the normal rate of return is greater than zero, but the results of a simulation for equal to zero are presented as an example of some extreme, theoretical case, just to compare the role played by the normal rate of return for industry development. The values of profit under $\rho = 0$ may be considered as a *natural* normal rate of return. In both series of experiments, close similarity of the model's behavior to real industrial processes is observed, and in this sense the results correspond to the findings of traditional economics. As in real processes of industry development, the greater the concentration of the industry, the larger the profit of the existing firms, but with the difference that, in contrast to the assumption of profit maximization of traditional economics, the objective of the firms in our model (the O_i—equation 1) is a combination of the short term (firm's income) and long term (firm's production, or expected firm's share).[4] The one extreme is monopoly (with profit in excess of 150% in our simulations); the other is perfect competition between an infinite number of firms with profit equal to zero. The profit drops very quickly with an increasing number of competitors. In our simulations, industries with the Herfindahl firms' number equivalent[5] n_H greater then 12 competitors may be considered as very close to the ideal situation of perfect competition (profit-to-capital ratio for these industries is smaller than 10^{-7}). The dynamics of change strongly depend on industry concentration. Starting from the same initial conditions, the more concentrated industries reach an equilibrium state much quicker. For fewer than eight competitors, the equilibrium state is reached within 20-40 years, but for a greater number of competitors the dynamics is significantly slower, and for industry very close to perfect competition (over 15 competitors), equilibrium is reached within 80-120 years. Many other simulation experiments suggest that for plausible values of parameters, the competition process may be considered as per-

Table 1. Industry concentration; global characteristics at the equilibrium state

n $n_H(0)$	Π/K [%]	Π/S [%]	p/V
normal rate of return $\rho = 0$			
1	151.907	71.685	4.2382
2	52.692	46.757	2.2539
4	22.096	26.915	1.6419
6	11.450	16.026	1.4290
8	6.050	9.160	1.3210
10	2.804	4.464	1.2561
12	0.643	1.060	1.2128
13	0.000	0.000	1.2000
16	0.000	0.000	1.2000
32	0.000	0.000	1.2000
normal rate of return $\rho = 0{,}05$			
1	146.908	69.326	4.2382
2	47.692	42.321	2.2539
4	17.096	20.824	1.6419
6	6.450	9.028	1.4290
8	1.050	1.590	1.3210
10	0.000	0.000	1.3000
12	0.000	0.000	1.3000
16	0.000	0.000	1.3000
32	0.000	0.000	1.3000

Where: Π - profit; K – capital; S – sales; p – price; V – unit cost of production

fect for the industries with the Herfindahl firms' number equivalent greater than 12. We observe a trade-off between the profit rate and the normal rate of return, for example, for highly concentrated industry if the normal rate of return increases from 0 to 5%, as in Table 1, the profit rate decreases also by 5%, and the price is kept on the same level. But the trade-off acts up to the moment when a positive profit for the same price of products is maintained. If the profit for the same price becomes a loss, then firms decide to increase the price to keep a zero profit and are satisfied with the normal rate of return. In our simulation, for $\rho = 5\%$, the trade-off is observed for industry with fewer than nine competitors; for a greater number of firms, the *natural* normal rate of return is lower than 5%, and the firms increase the price to keep profit equal to zero (compare relevant values in Table 1). The positive normal rate of return also causes the profit-to-sales ratio to diminish, but there is no full trade-off as between the normal rate of return and the profit/capital ratio. Reduction of the profit/sales ratio is always smaller than the increase in the normal rate of return.

Changes of the values of the capital physical depreciation have a similar effect on the characteristics of industry development as changes in the normal rate of return; for example, we observe a similar trade-off between the capital physical depreciation and the profit as we observe in experiments with a positive normal rate of return; reduction of the capital physical depreciation (amortization) in highly concentrated industry by 5% leads to an increase of the profit/capital ratio, also by 5%. So it may be expected that for highly concentrated industries, the rising of amortization or rising of the normal rate of return will not significantly affect the products' price, but for less concentrated industries we may expect higher prices to cover the higher opportunity costs.

Table 2. Concentration of the market; non-uniform firms' size distribution

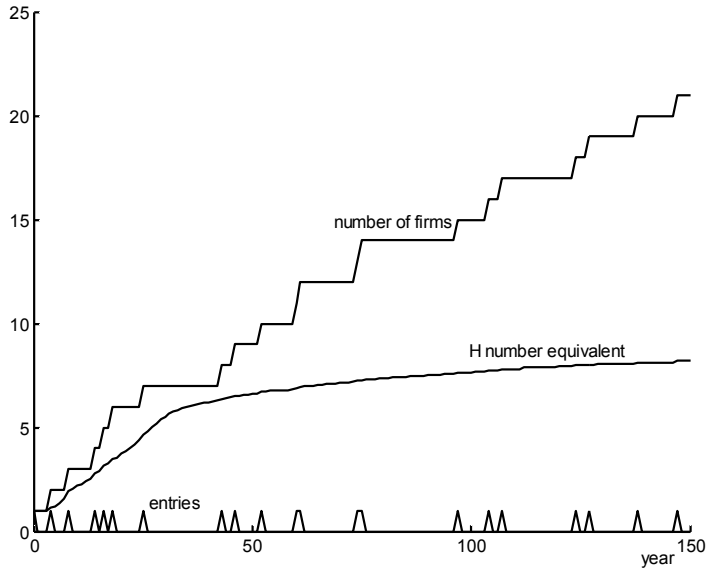
n	$n_H(0)$	$n_H(100)$	$n_H(200)$	T_e [year]	$\Pi/K(100)$ [%]	$\Pi/K(100)$ [%]	$p/V(200)$
2	1.02	2.00	2.00	14	47.692	47.692	2.2539
4	2.61	4.00	4.00	22	17.096	17.096	1.6419
6	4.18	6.00	6.00	47	6.450	6.450	1.4290
8	5.75	7.30	7.68	–	2.932	2.282	1.3456
12	8.93	9.76	9.81	–	0.216	0.033	1.3007
16	12.12	12.15	12.16	–	0.026	0.001	1.3000
32	25.52	25.59	25.59	–	0. 022	0.001	1.3000

Note: T_e is a year in which the H index is equal to the number of firms, i.e. $n_H = n$. The years of measurement of relevant characteristics are given in parentheses.

The dynamics of change also depend on the initial structure of industry. To investigate to what extent the initial firms' size distribution influences the dynamics of the process, the following series of experiments were made. Starting from highly diversified firms' size, we measure the values of basic characteristics of industry over the course of time and observe the tendency towards uniform distribution for different concentrations of the industry. The initial Herfindahl firms' number equivalent and some general characteristics of development of the model for a different number of competitors for t = 100 and 200 are presented in Table 2. For relatively high concentration of the market (that is, for the number of firms smaller than eight), there are no significant differences in the dynamics of change between industries with uniform and non-uniform firms' size distribution. This is due to a very strong tendency towards uniform distribution (caused by intensive price competition) for the highly concentrated industries. The more concentrated the industry is, the quicker the uniform firms' size distribution is reached—compare values of T_e in Table 2 for highly concentrated industries. For a small concentration of the industry, the dynamics of reaching the equilibrium state are significantly lower and also there is no such strong tendency towards the uniform firm's size distribution; quite the contrary, some conservative tendency to stabilize the size distribution is observed. For industries very near to perfect competition, the distribution of the firms' size is almost the same as at the beginning of simulation (seen in relevant values of n_H for years 100 and 200, when the number of firms is greater than 12). As we will see in the next section, the only way for small firms to pursue the big firms and to gain higher market shares is to introduce innovation.

In the following series of experiments, an investigation of the ability of free entrants to penetrate the industries of different concentrations has been made (no economies of scale present). It was assumed that for a given number of equal-sized firms, at some moment, a small firm with an assumed small capital enters the market. From the moment of entrance, we observe the evolution of the structure of industry, and particularly we observe the market share of the entrant. What interests us is "Does the entrant's market share grow to reach the same size as that of the initial firms?" Or in other words, "Is the firms' size distribution at equilibrium uniform?" As a measure of convergence, we use time T_e which spans from the moment of entrance to the moment of the uniform firms' size distribution (let us call this time the penetration time). As it turns out, the invasion is quite easy for a highly concentrated industry; for example, for the monopoly industry the newcomer is able to increase its initial market share of 0.5% to the equilibrium 50% market fraction in nine years: for two, three, and four firms, the relevant values of the penetration time T_e are 16, 22, and 35 years, respectively. The penetration time grows hyperbolically with diminishing concentration of industry; for example, if the industry is dominated by six competitors, the newcomer needs 98 years to get the same fraction of the market as the initial firms, and for seven firms the relevant time becomes very long, namely 195 years. There is no possibility of penetrating the market if the number of firms is greater than seven. Because of much higher competitive conditions, the average profit within the industry is very small, and the newcomer is not able to collect enough capital to invest and to raise its market share.[6] The penetration time for n_{H} greater than seven is infinite; at the equilibrium state the newcomer's market share stabilizes at a very low level, which is lower than the smaller the industry concentration is; for example, for eight, nine, ten, and fifteen competi-

Figure 4. Number of firms, entries, and Hefindahl-Hirshman firms' number equivalent



tors, the newcomer's share at equilibrium is equal to 0.35%, 0.11%, 0.1%, and 0.09%, respectively.

In the basic model only the price competition is considered, and as we see, it is very difficult to enter the market under perfect competition. The prerequisite for successful invasion of the highly competitive market is concurrent introduction of the product's innovation, but this problem will be discussed in the next section, where the model which incorporates a search for innovation process will be presented. The orthodox economics states that in oligopolist industries, market shares are usually determined on the basis of non-price competition, such as advertising and product variations, so the real firms' size distribution deviates from the uniform one, and that oligopolists frequently have adequate financial resources to finance non-price competition. Basically it is true and we observe such type of industry behavior in the presence of incremental innovations (to some extent responsible for the *product variations*).

From Monopoly to Perfect Competition

Typical history of any industry starts from a single founder firm. For some time that firm is a monopolist on a small market. In a course of time, the market is growing and new competitors enter the market. Orthodox (textbook) economics assumes that with no entry barriers and no innovation, the industry will evolve toward perfect competition with a large number of equal, small firms. Let us create that situation in our simulation. We start from a single, small firm, the industry growth rate is equal to 3%, firms do not innovate, and there is the possibility of firms' entering into a market. Even in such simple economy, the process of industry development is far from that proposed in economics textbook. The results are pre-

Figure 5. Evolution of industry structure with entry and no innovation

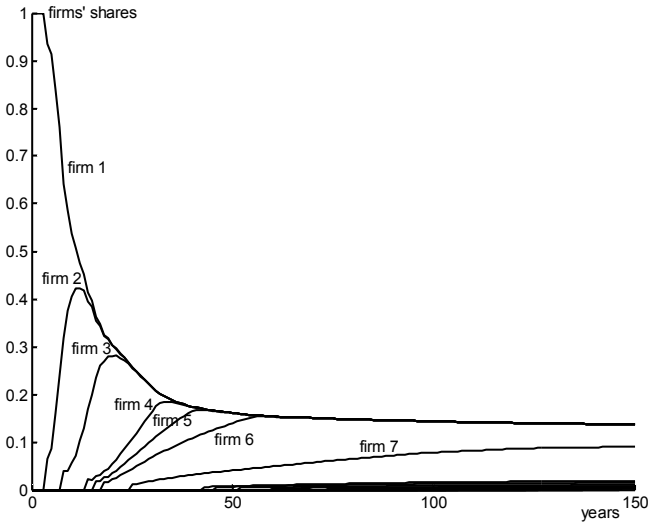
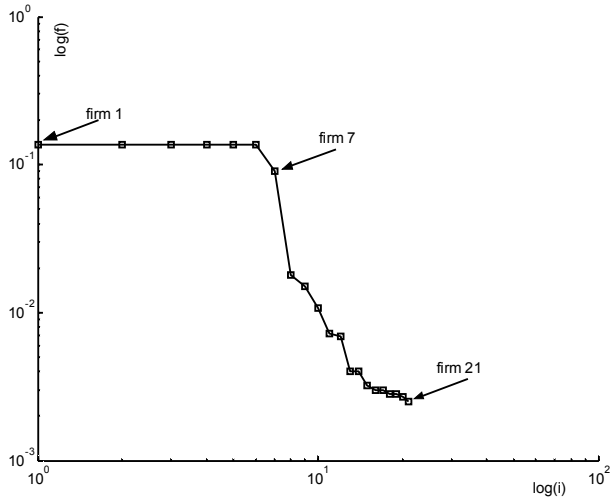
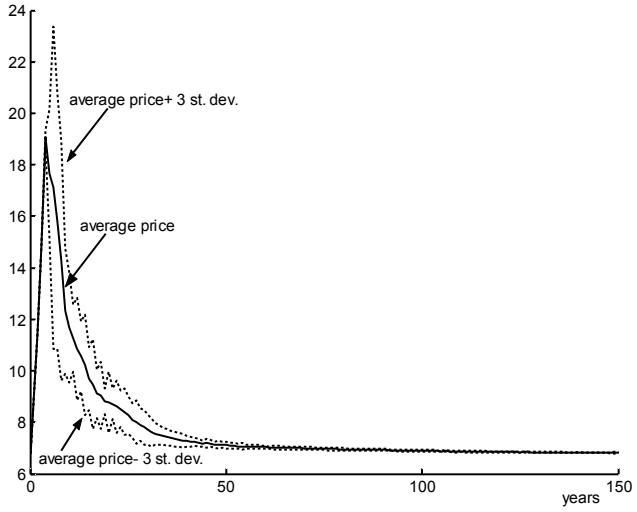


Figure 6. Firms' size distribution (log-log scale)



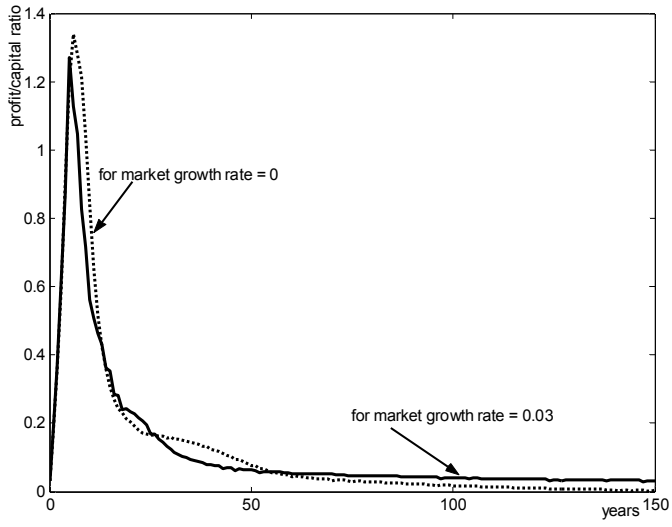
sented in Figures 4 to 7. Steady growth of a number of firms operating on the market are summarized in Figure 4. Due to relevant incumbents' price policy, we do not observe exits from the market. After 150 years of development, there are 21 firms, but contrary to the orthodox postulate, firms are not equal sized. In fact only the early entrants (in our experiment, the first five entrants) are able to compete efficiently with the founder firm and relatively

Figure 7. From monopoly to competitive price



quickly, all six firms have the same market shares equal to 16% (see Figure 5). This efficient competition was able due to *natural* price policy of the monopolist. The founder firm increased the price in the first years of industry development up to 19 units (Figure 7) and next was forced do reduce the price in a course of new competitors' emergence. Entering firms impose lower price, therefore their products are more competitive and their market shares increase. Notice that this process of price competition leads to diversity of prices on the market (in Figure 7, this high diversity of price is seen by comparison of two dotted lines related to the range 'average price ± three standard deviations'). Later entrants are not able to compete efficiently because the average price is significantly reduced and the price margin is very low. The good example of such late entrant is *firm 7* which entered the market it the twenty-seventh year. Although the price and its diversity at this time were relatively high, but during the expansion of that firm the price was significantly reduced to its equilibrium value therefore after over 100 years, in the end of simulation its market share was equal to 9%—much lower then the share of the first six firms with their market shares equal to 14%. To the end of simulation, another 14 firms entered the market and their equilibrium shares were lower the later the entering year. We can say that contrary to the orthodox postulate that under perfect competition we have a large number of equal-sized firms, our simulation results suggest that natural mechanisms of competition force emergence of different-sized firms. The equilibrium firms' size distribution is far from the orthodox uniform one; in fact it consists of two segments—the first segment relates to the relatively few early entrants and has uniform distribution, and the second one relates to later entrants and its distribution is highly skewed. Typical equilibrium distribution is presented in Figure 6: the first six firms reach equal shares and all other 15 entrants have much lower shares (the shares of last nine entrants is smaller then 0.5%).[7]

Figure 8. Profit-to-capital ratio for stable and expanding market



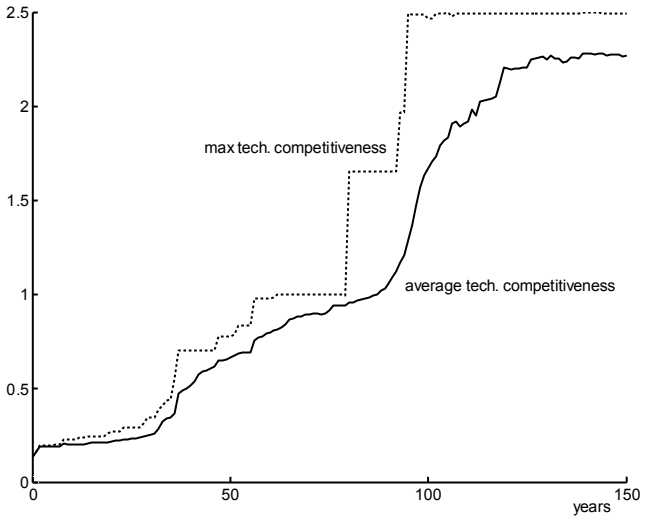
The path from monopoly to perfect competition structure is also seen in time changes of profit-to-capital ratio (Figure 8). In the first years of industry evolution, the concentration was very high (Herfindahl firms' number equivalent was smaller than two firms—see Figure 4, n_H, 'H number equivalent') and the profit ratio skyrocket to almost 130%, but it was quickly reduced in the next decades to less then 20%. In the equilibrium the profit ratio is very close to the growth rate of the market (i.e., to 3%). This is another flaw of the orthodox economics, which teaches that under perfect competition, the profit ratio is equal to zero. It is true only for the stable market. Just for comparison, changes of the profit ratio in the case of stable market (i.e., for growth rate equal to zero) is presented in Figure 8 (dotted line). It is seen that there are no significant differences between stable and growing markets in the early decades of evolution, but the differences are significant at the near equilibrium states. The profit is equal to zero for stable market, but is close to the market growth ratio for expanding market.

Innovation and Industry Evolution

The most important weakness of the orthodox economics seems to be lack of innovation in their models and concentration on equilibrium analysis. Innovation can be considered as the heart of modern economic evolution. Here we present only a small sample of the simulation results just to show how similarities of the proposed model's behavior relate to real industrial innovative processes.

The only difference with the conditions of simulation presented in the former section is possibility of searching for innovation—that is, firms are able to modify their routines just to find innovations leading to improving technical competitiveness of their products, diminishing unit cost of production and to increasing the productivity of capital. Just to show how diversified behavior of the firms is, the changes of technical competitiveness, variable cost of production, and productivity of capital are presented in Figures 9, 10, and 11. Besides the average values of relevant characteristics, the

Figure 9. Average technical competitiveness and maximum technical competitiveness



so-called frontiers of development are presented (namely maximum values of technical competitiveness, minimum values of unit cost of production, and maximum productivity of capital). The discrepancies between the average values and the frontiers give a hint about the existing diversity of firms' characteristics. It is important to underline that the mode of development perceived through the average values is a rather gradual one, but the mode of frontiers development is far from being gradual. It is clearly seen that the frontiers evolution is a punctuated one (i.e., the stasis periods are separated by jumps of the frontier values). The jumps in the frontiers are related to radical innovations' emergence. In the stasis phases innovations are also introduced but they are incremental ones. The most visible effect of introducing radical innovation relates to increase of market shares of successful firms; firm 8 introduced the first radical innovation (around the year 40) and firm 9 the second radical innovation around the year 70, and their shares increased significantly during next two or three decades after introducing the radical innovation (see Figure 12). Success terms of gaining significant market share can also be reached by introducing a series of relatively important incremental innovations. This is a case of firm 6 (Figure 12) which introduced such series in the third decade of industry development. Notice that besides the relatively small number of firms having significant shares of the market, there always exists a large number of small and very small firms (Figure 12).

This process of radical innovation emergence strongly influences the mode of changes of other characteristics of industry development. As an example we present changes of firms' number, price and, profit ratio (Figures 13, 14, and 15, respectively). We see that the emergence and dissemination of radical innovation causes an increased number of exits and a significant reduction in the number of firms, as well as an increase in market concentration (Figure 13—number of firms, exits, and *H number equivalent*). Emergence of radical innovation also causes an increase in price

Figure 10. Average unit cost of production and minimum unit cost of production

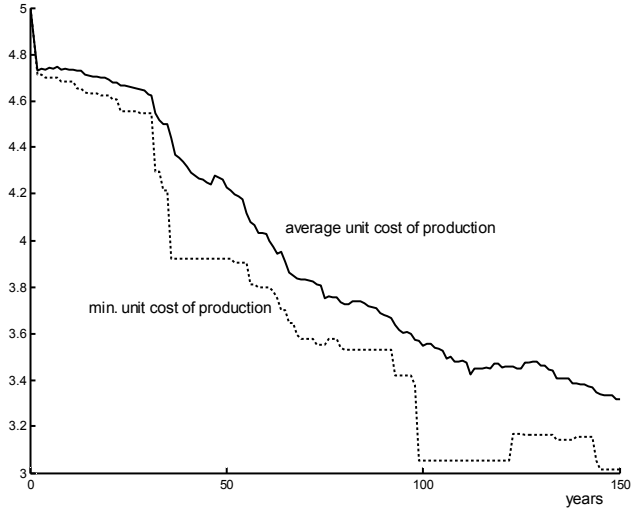
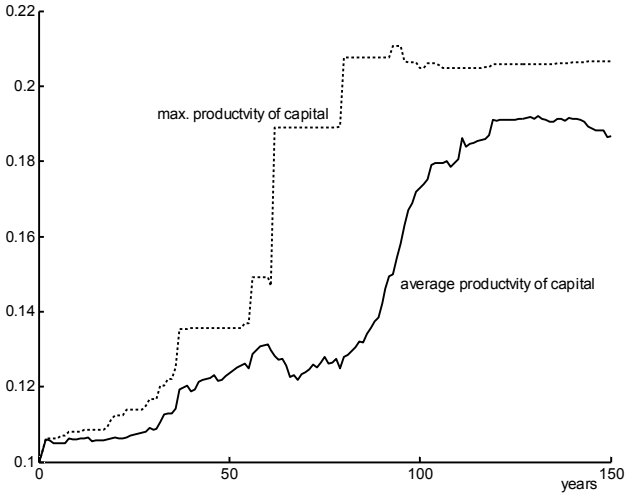


Figure 11. Average productivity of capital and maximum productivity of capital



diversity (Figure 14). This occurs because innovative firms tend to increase the price (to gain temporal monopoly rent and to cover the costs of R&D process), and the unsuccessful firms, having no possibility to imitate the innovation of successful firms, decrease the price just to made their technologically obsolete products more competitive.

The last few decades of industry development in our simulation run seem to be interesting. The orthodox economics suggests that high industry concentration is usually related to high

Figure 12. Evolution of industry structure with entry, exit, and innovation

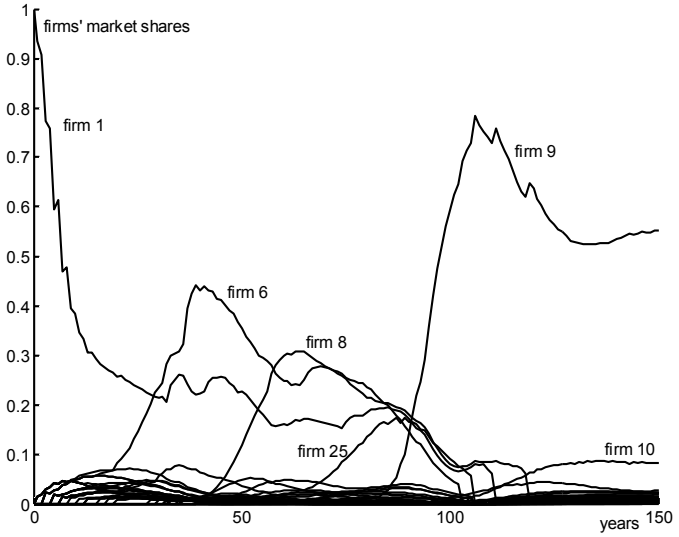


Figure 13. Number of firms, units, entries, and exits

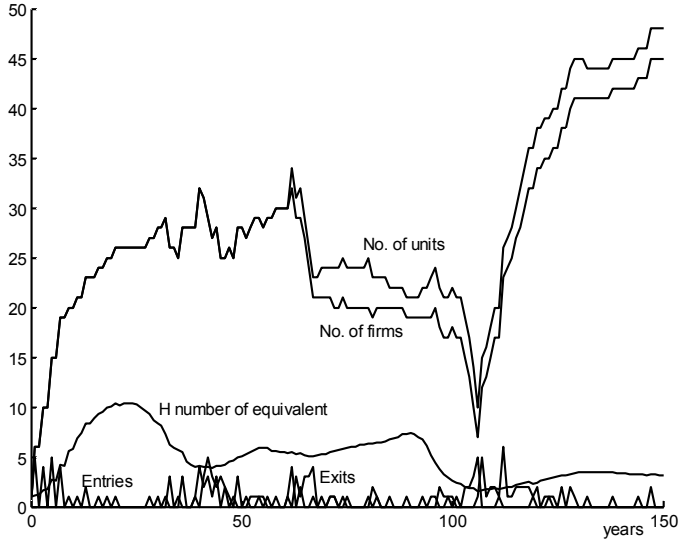
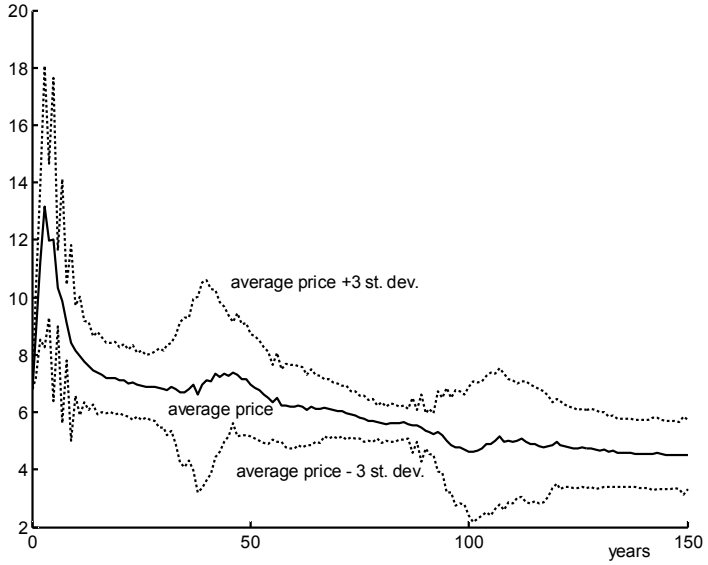


Figure 14. Average price and its diversity

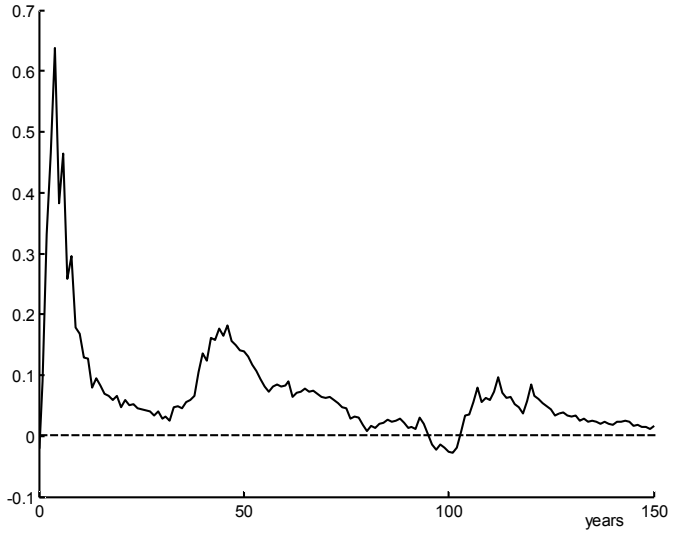


prices and large profits. Figures 13, 14, and 15 show that in that period we have a relatively large number of firms operating in the market (around 40 firms), but concentration of the market was rather high (H number of equivalent was equal to 3.2 firms). This high concentration is accompanied by low, although diversified, price (Figure 14) and very small profit ratio (1.5%). We can identify the development of the industry in the last decades of simulation to well-known, orthodox market structure of monopolistic competition.

CONCLUSION

Results of simulation experiments show that evolutionary modeling allows support of textbook conclusions related to industry development, but also reveals the weakness of the orthodox, textbook analysis. Repertoire of behavior of evolutionary models is much richer that those presented in the economics textbook. We can say that all phenomena related to textbook industrial analysis can be explained within the evolutionary paradigm, but the evolutionary analysis allows the explanation of a much wider spectrum of phenomena. Among them are questions related to the necessary number of firms operating in a market to call the market the perfect competitive one, problems of trade-off between profit rate and the normal rate of return, non-uniform firms size distribution for the perfect competition market, and the importance of innovation for industry behavior. It is shown that the closeness of evolutionary modeling to real processes is far reaching. One of the important conclusions from that paper and from the experience of teaching microeconomics is that evolutionary analysis may be considered as a very useful and complementary tool to teach economics.

Figure 15. Profit-to-capital ratio



REFERENCES

Dopfer, K. (Ed.). (2005). *The evolutionary foundations of economics*. Cambridge, UK: Cambridge University Press.

Frenken, K. (2005). *History, state and prospects of evolutionary models of technical change: A review with special emphasis on complexity theory*. Utrecht University, The Netherlands. Retrieved from http://www.complexityscience.org/NoE/Frenkencomplexityreview.pdf/

Kwasnicki, W., & Kwasnicka, H. (1992). Market, innovation, competition. An evolutionary model of industrial dynamics. *Journal of Economic Behavior and Organization, 19,* 343-368.

Kwasnicki, W. (1996). *Knowledge, innovation, and economy. An evolutionary exploration* (2nd ed.). Cheltenham; Brookfield: Edward Elgar.

Nelson, R. R. (1995). Recent evolutionary theorizing about economic change. *Journal of Economic Literature, 33,* 48-90.

Nelson, R. R., & Winter, S. G. (1982). *An evolutionary theory of economic change*. Cambridge, MA: Harvard University Press.

Saviotti, P. P. (1996). *Technological evolution, variety and the economy*. Cheltenham; Brookfield: Edward Elgar.

Silverberg, G. (1997). *Evolutionary modeling in economics: Recent history and immediate prospects*. Research Memoranda 008, Maastricht Economic Research Institute on Innovation and Technology, The Netherlands. Retrieved from http://ideas.repec.org/p/dgr/umamer/1997008.html

Silverberg, G., & Verspagen, B. (2003). Evolutionary theorizing on economic growth. In K. Dopfer (Ed.), *The evolutionary principles of*

economics (revised ed.). Cambridge: Cambridge University Press. Retrieved from http://ideas.repec.org/p/wop/iasawp/wp95078.html

Simon, H. A. (1986). On behavioral and rational foundations of economic dynamics. In R. H. Day & G. Eliasson (Eds.), *The dynamics of market economies.* Amsterdam: North-Holland.

Wallace, A. R. (1898). *The wonderful century. Its successes and failures*. New York.

Winter, S. (1984). Schumpeterian competition in alternative technological regimes. *Journal of Economic Behavior and Organization, 5,* 287-320.

KEY TERMS

Competitiveness: Ability of economic agents (firms) to compete in the market by offering technologically advanced or cheaper products.

Evolutionary Economics: In the broadest sense, a study of economic phenomena using analogies and metaphors of biological evolution. It represents an alternative approach to so-called "mainstream economics," where analyses is based on mechanical analogies and metaphors borrowed form classical physics.

Replicator Equation: A differential or difference equation that defines the selection dynamics of a population of competing agents (firms), considered within a frame evolutionary economics (also of evolutionary games).

Routine: *Regular and predictable behavioral patterns of firms*; in evolutionary economics, the concept of routine plays a similar role as concept of gene in evolutionary biology.

ENDNOTES

1 Good reviews of recent literature on evolutionary modeling can be found in Silverberg (1997), Silverberg and Verspagen (2003), and Frenken (2005). See also http://prawo.uni.wroc.pl/~kwasnicki/todownload/Schumpeterian modelling.pdf and http://prawo.uni wroc.pl/~kwasnicki/todownload/NW conference.pdf

2 Further reading on that model can be found at http://prawo.uni.wroc.pl/~kwasnicki/e-model.htm

3 What follows is only a short description of the essential features of these basic structures as understood by traditional (textbook) economics: *Pure (or perfect) competition* is a feature of industry which consists of a large number of independent firms producing a standardized product; no single firm can influence market price; the firm's demand curve is perfectly elastic, therefore price equals marginal revenue. *Monopoly* is where there is a sole producer of a commodity, and there are no straight substitutes for that commodity. *Oligopoly* is characterized by the presence within the industry of a few firms, each of which has a significant fraction of the market. Firms are interdependent; the behavior of any one firm directly affects, and is affected by, the actions of competitors. In *monopolistic competition,* there is a large enough number of firms; each firm has little control over price, interdependence is very weak or practically absent, so collusion is basically impossible; products are characterized by real and imaginary differences; a firm's entry is relatively easy.

[4] More detailed discussion on efficiency of different firm objectives are presented in Kwasnicki (1992, 1996).

[5] The Herfindahl–Hirschman index of concentration of the industry is equal to $H = \sum_i (f_i)^2$, where f_i is the market share of fim i. The Herfindahl firms' number equivalent is defined as $n_H = 1/H$ and is the number of equal-sized firms that would have the same H index as the actual size distribution of firms.

[6] The raising of the price above that imposed by the *old* firms to get higher profit is not possible because of diminishing competitiveness of the newcomer's products.

[7] In the end of simulation, when the industry was very close to the equilibrium state, there were 21 firms of different size, but the Herfindahl firms' number equivalent was equal to 8.2 (i.e., the market situation was similar to the perfect competition of roughly eight equal-sized firms of 12.5% market share each)—see Figure 4.

Chapter XXI
Population Symbiotic Evolution in a Model of Industrial Districts

Ugo Merlone
University of Torino, Italy

Pietro Terna
University of Torino, Italy

ABSTRACT

This chapter considers a model of industrial districts where different populations interact symbiotically. The approach consists of the parallel implementation of the model with jESOF and plain C++. We consider a district decomposition where two populations, workers and firms, cooperate while behaving independently. We can find interesting effects both in terms of worker localization consequences and of the dynamic complexity of the model, with policy resistance aspects.By using a multiple implementation strategy, we compare the advantages of the two modeling techniques and highlight the benefits arising when the same model is implemented on radically different simulation environments; furthermore we discuss and examine the results of our simulations in terms of policy-making effects.

INTRODUCTION

In this chapter we consider a model of industrial districts where different populations interact symbiotically. According to Becattini (2003), the industrial district's "first and fundamental decomposition is to be the one between the productive apparatus and the human community in which it is, so to speak, 'embedded'." In our approach, the two populations we consider allow for this decomposition: the first one represents the productive apparatus, while the second one can be considered the human community. Several aspects about industrial districts have been examined in the literature; for an introduction, the reader can refer to Garofoli (1981, 1991, 1992), Becattini et al. (1992), or Belussi and Gottardi (2000). Carbonara (2005) analyzes some common key features in the literature on geographical clusters and in particular on industrial districts.[1] She identifies, among the others, both "a dense network of

inter-firm relationships, in which the firms cooperate and compete at the same time" and "a dense network of social relationships, based mainly on face to face contact, which is strictly inter-connected with the system of economic relationships" (p. 217). The first aspect looks extremely interesting and is one of the aspects we consider in our approach. In fact, starting from the definition given by Squazzoni and Boero (2002), where "industrial districts can be conceived as complex systems characterized by a network of interactions amongst heterogeneous, localized, functionally integrated and complementary firms," we introduce and model the role of workers interacting with firms. The model of industrial district we obtain consists of two populations having different peculiarities and interacting in the same environment. In the literature, the representation of districts as communities of populations is not a new idea (e.g., Lazzeretti & Storai, 1999, 2003); nevertheless, to the best of our knowledge, studies devoted to the dynamical evolution of these populations are still limited.

Ecological models of population dynamics for different species can be found both in mathematical ecology and in computer science literature. Ecology of populations examines the dynamics of a number of organisms. In this field the use of mathematical models is quite common in explaining the growth and behavior of population; for a first introduction the reader may refer to Hastings (1997). The most famous model is probably the well-known Lotka-Volterra prey predator model (Lotka, 1925; Volterra, 1926); in this model, which is the simplest prey predator system, two species coexist with one preying on the other (for a concise mathematical discussion of the model, the reader may refer to Hofbauer and Sigmund, 1998). For more recent contributions about mathematical models of population, the reader may refer to Royama (1992). When considering different populations, cooperation has been another examined thoroughly in the literature, and specifically the evolution of cooperation (e.g., Axelrod, 1984). Most of the contributions stem from the well-known prisoner's dilemma game: for example, Flake (1998) discusses an ecological model where only a limited number of organisms can be supported and the population adopting a given of each strategy is some fractional part of the entire ecosystem; other approaches consider both cooperation and the geometry of the interaction network (see Gaylord & D'Andria, 1998, for some examples).

In the model of industrial districts we consider, cooperation is in some sense more implicit, since the structure of the model assumes that workers and firms cooperate. In fact, each of the two species (namely, the workers and the firms) is necessary to the other. In this sense our model exhibits a sort of necessary symbiotic evolution of the two species. In Frank (1997) three different models of symbiosis are considered: the first one is the interaction between kin selection and patterns of transmission; the second is the origin and the subsequent evolution of the interactions between two species; finally, the third considers symbiosis as asymmetrical interaction between species, in which one partner can dominate the other. Our model describes a symbiotic interaction that is similar to the second case, even if, when some parameters of the model are chosen appropriately, we may have a slight dominance of one species. Starting from this approach we consider the dynamics of the populations of firms and workers and their evolution. In particular, we are interested in shedding light on the emergence of industrial districts when the mentioned decomposition is considered, showing that this simple interaction is sufficient for firms to form clusters. While this cannot be an exhaustive explanation of districts' behavior, it is an interesting insight.

Since the system is highly complex, a valid mathematical model of it is itself complex, precluding any possibility of an analytical solution. As is common in these cases, the model must be studied by means of simulation; for further details on the role of simulation, the reader may refer to Law and Kelton (2000). The simulation approach for the analysis of districts is not new. For example, Zhang (2003) uses agent-based simulation to study the dynamics of high-tech industrial clusters, Squazzoni and Boero (2002) use computational techniques to focus on some evolutionary fundamentals of industrial districts modeling, and Brenner (2002) uses simulation to study some mechanisms of the evolution of industrial clusters.

While in the simulation of real systems several techniques are used in order to increase the model validity and credibility, not all of them can be used when implementing a theoretical model such as in our case. Software developers are well aware that computer programming is an intrinsically error-prone process, for example Becker (2005) claims: "As applications become more complex, their resource management requirements become more complex, and despite our best efforts, our designs often have holes in them, or we apply our designs incorrectly, or we make coding errors" (p. 37). This is well known in the simulation literature, for example in Law and Kelton (2000) several techniques are suggested for the model verification, including: "… determining whether the conceptual simulation model (model assumptions) has been correctly translated into a computer program" (p. 264). Nevertheless only recently the agent-based modeling literature seems to be aware of the potential pitfalls (e.g., Polhill et al., 2005). For these reasons we decided to use the same approach of Cerruti et al. (2005)—to consider and compare two different model implementations. To obtain two completely identical implementations, however, it is not straightforward given the complexity of the considered model. Nevertheless, we found that the whole process of comparing and discussing the different implementations is extremely beneficial. In this sense while results replication is to be considered preliminary—at the moment we cannot obtain exactly the same results with the two implementations—the process of discussing and comparing different implementations details and results seems to be extremely beneficial and promising in terms of model verification. Finally, this approach looks promising in dealing with some important issues such as those mentioned in Polhill et al. (2005).

The structure of the chapter is the following. First, we describe the model of industrial district we consider. Second, we give a theoretical formalization of the model and explain our simulation approach. In the following sections we describe the computational approach, illustrate and compare the two different computer implementations, and discuss the simulation results. Finally, the last section is devoted to conclusion.

THE MODEL

The model we consider simulates a small industrial district consisting of four different components:

- the orders or productive tasks to be completed—the tasks come from the external world and, when completed, disappear from the model scope;
- the skills or abilities that are necessary to perform the different production phases of each order;
- the firms receiving the orders either from the market or from other firms and processing them; and
- the workforce that, when hired by firms, allows them to process the orders.

Finally, in each time period new workers and firms are, according to some parameters, generated and randomly located.

Here we describe each of the four components and discuss their mutual relationships.

Each order contains a recipe, namely the description of the sequence of activities to be done by the firms in order to complete a specific product. The different activities or phases to be done belong to the skill set *S* but are not necessarily of the same kind; in this sense skills can be considered as technical abilities. This assumption is motivated by different studies of skill dynamics in manufacturing; for an empirical study on the Italian manufacturing firms, the reader may refer to Piva et al. (2003). To explain how skills are modeled, consider a district with three skills 0,1,2. A feasible example of order is '00120': this is a five-phase order where the first two phases need skill 0, the third needs skill 1, the fourth skill 2, and the last one skill 0. Each firm is specialized in a single skill and the same holds for workers. Obviously other approaches in modeling production processes using recipes are possible; for an example the modeling of production operations as recipes is adopted by Auerswald et al. (1998), or for a complexity and knowledge-based approach, the reader may refer to Sorenson et al. (2004).

In the current version of the model, specialization for firms and workers is randomly attributed and does not change. Firms can only process order phases denoted with their skill specialization, and workers can only be hired by firms with the same skill. The orders consisting of non-homogeneous phases need to be sent to different firms to be completed. The mechanisms workers are hired with and the orders firms pass on to each other rely on the social structure of the district: the environment is a (social) space with (metaphorical) distances representing trustiness and cooperation among production units (the social capital). While firms have a social visibility that increases according to their longevity, workers' visibility is fixed. Only mutually visible agents can cooperate—that is, firms may hire only skill-compatible workers that are in their social network and orders can be passed between mutually visible firms. This aspect refers to the other key feature that is mentioned by Carbonara (2004)—that of the network of social relationships. In our model, to keep the analysis simple, we do model the network in terms of distance. With these assumptions an important aspect of the model arises: cooperation is not optional; rather it is necessary for agents to survive.

While at each turn of the simulation both firms and workers bear some costs, both hired workers and producing firms receive some revenue. In the current version net profits and wages are modeled simply assuming marginal costs to be not greater than marginal revenues for firms and a positive salary for workers. As a consequence, if for a prolonged time either a worker is not hired or a firm has no worker, their balance may become negative. Workers and firms with negative balance disappear. Negative balance is the only cause of default for workers.

By contrast, other reasons for a firm default are either prolonged inactivity or the impossibility of sending concluded orders to other firms to perform the successive steps. While the interpretation of the first two default causes for firms is straightforward, the third one requires some explanation. First, we assume that a totally completed order is absorbed by the market; the rationale for this is that since the market generated this order, there is demand for it. Second, recalling that orders may consist of different phases, when a partially completed order cannot be sent to a firm with the needed skill, this means either that such a firm at the moment does not exist or that it is out of scope.

The latter may be interpreted as a lack of knowledge and trust—that is, gaps in the social capital. It is worth noting that all these aspects are consistent to the mentioned definition of industrial districts given by Squazzoni and Boero (2002). In all these cases the permanence of the agent on the market is economically unfeasible; for these reasons we summarize these situations with the broad term default.

All the firms and workers are located and operate on a two superimposed toroidal grids: one for the workers and one for the firms.[2]

The novelty of this model structure is the introduction of the interaction "within" a model layer (the one containing the productive structures), while the classical Lotka-Volterra structure exploits only the consequences of the interaction *between* two different layers. Firms, to produce, have to interact with other firms, with the constraint of considering mutually visible only the units sharing a portion of their visibility space as a synthetic way for representing trustiness. The *within* interaction influences the dimension of the productive clusters, while the *between* interactions have the role of determining the spatial localization of the clusters. With this abstract but not unrealistic tool, we can verify the emergence of well-known phenomena and, in a parallel way, of new ones, which appear to be not obvious, but plausible behaviors of the district structures.

THE COMPUTATIONAL APPROACH

Our model can be formalized as a discrete time dynamic system. Consider two sets of variables, the first one describing the worker $w_i \in W_i$ and the second one describing the firm $f_i \in F_i$. Since workers and firms are located on two $p \times q$ superimposed toroidal grids, we can consider a single grid where each cell can: (a) contain a worker and no firm, (b) contain a firm and no worker, (c) contain both a firm and a worker, or (d) be empty. In cases a, b, and c, the cell state consists of the informative variable of its content, while in case d all the informative variables are null. The state of cell at location i, j can be formalized as a vector $\mathbf{x}_{ij} = (\mathbf{w}, \mathbf{f}) \in W_1 \times \cdots \times W_m \times F_1 \times \cdots \times F_n$ with the convention that either **w** or **f** can be the null vector, when respectively either no worker or no firm are present. We define the time *t state of the system* as the vector of the states of cells at time t $\mathbf{x}^t := (\mathbf{x}^t{}_{11}, \mathbf{x}^t{}_{12}, \ldots, \mathbf{x}^t{}_{pq})$. Finally, consider the following stochastic processes:

- $\{\tilde{O}^t, t \in \mathrm{N}\}$ describing the orders generation
- $\{\tilde{W}^t, t \in \mathrm{N}\}$ describing new workers entry
- $\{\tilde{F}^t, t \in \mathrm{N}\}$ describing new firms entry

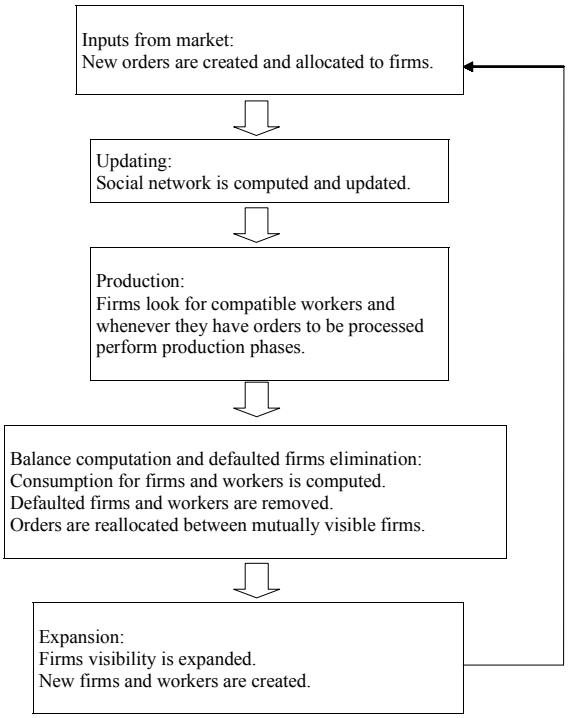
The evolution of the system can be formalized as follows:

$$\mathbf{x}^{t+1} = \varphi(\mathbf{x}^t, \tilde{O}^t, \tilde{W}^t, \tilde{F}^t)$$

The direct consequence of non-linearity and complexity is that the theoretical analysis of the formal model is not straightforward. This is well known in the literature and, according to many authors (e.g., Carley & Prietula, 1994), many models are too complex to be analyzed completely by conventional techniques that lead to closed-form solutions. In order to obtain some results, turning to simulation is natural and necessary.

Having described all the agents' interaction in our model of industrial district, to introduce the structure of the simulation is immediate. For each turn, first new orders are created and allocated to firms, then, after updating the social network induced by the mutual visibility of firms and workers, the productive process begins. Successively compatible and visible

Figure 1. Scheme of a simulation turn



workers are hired by firms, all the firms with orders perform productive phases, then costs and revenues are accounted for and balances are computed. Defaulted workers and firms are removed. Finally visibility for firms is updated, and new workers and firms are created. At this point the state of the system is displayed both in terms of graphical output and in terms of relevant data.

The activities of one turn are displayed in Figure 1.

THE COMPUTER IMPLEMENTATION

Different simulation tools are available to social scientists interested in agent-based simulations. Among the most widely used we recall Swarm, Repast, and NetLogo; yet other approaches are possible. For example it is also possible to implement models by custom simulation platforms using high-level languages. While the choice of the simulation platform should have no effects on the simulations results, this might not always be true (e.g., Polhill et al., 2005). Furthermore, in science, repetition is what allows results to be validated and accepted by the scientific community.

Furthermore, in science, repetition is what allows results to be validated and accepted by the scientific community. Considering computer simulations, repetition can be intended just as experiment replication by other scientists or, more dramatically, as model reimplementation. While experiment replication can be easily achieved, reimplementation is a more rigorous process in which the model is deeply examined again. Given the complexity of the models considered, and the fact that computer programming is an extremely error-prone process, reimplementation of the model may be extremely valuable in order to identify potential problems that may invalidate results.

We implement this model using both Swarm and a C++ custom-developed platform.

Our purpose is twofold. First, we compare the advantages of the two implementations and highlight the benefits arising when the same model is implemented on radically different platforms; second we discuss and examine the results of our simulations in terms of policymaking effects.

The Swarm Implementation

We use the original package jESOF (java Enterprise Simulation Open Foundation, described at http://web.econ.unito.it/terna/jes). The package is built using the Swarm library for agent-based models (described at http://www.swarm.org).

The purpose of the jESOF structure is to develop a two-side multi-layer world, considering both the actions to be done, in terms of orders to be accomplished (the "What to Do" side, WD), and the structures able to execute them, in terms of production units (the "which is Doing What" side, DW). WD and DW can be consistent or inconsistent and the presence of social capital—expressed by the necessity of firms' inter-visibility as a condition to exchange—introduces inconsistent situations reproducing real-world occurrences.

The jESOF dictionary follows:

1. **Unit:** a productive structure; a unit is able to perform one of the steps required to accomplish an order.
2. **Order:** the object representing a good to be produced; an order contains technical information (the recipe describing the production steps).
3. **Recipe:** a sequence of steps to be executed to produce a good.

The central tool in this simulation environment is a proprietary scripting language used to describe units acting in the simulated world and to define actions to be done within the simulation framework by the recipes contained in the orders. The recipes can call computational functions written in Java code to make complicated steps such as creating new firms or workers, to hire workers, to account for income and consumptions of the different units.

In our case the scripting language uses two different sets of recipes included in orders.

- In the firm stratum we have recipes related to production, with sequences of steps describing the good to be produced.
- In the workers' stratum, which is also the interaction place, recipes produce five kinds of effects: (1) new workers appear in the simulation context, either near to similar ones or randomly distributed; (2) firms hire workers and recipes modify workers and firms' private matrixes—this is done accounting for both the availability of the labor production factor (firm side) and household income (workers side); (3) firms make use of available labor production factors; (4) firms either short of orders to be processed, or lacking adequate workers on the market, or being unable to deliver produced goods, disappear from the economic scenario; (5) workers also disappear if unable to find a firm for a prolonged time.

Recipes are able to perform complex tasks, such as those described above, and are developed via computational steps. These steps can be interpreted as calls to code functions (methods of a Java class) invoked by a scripting language.

As an example, the sequence '1001 s 0 c 1220 2 0 0 1 s 0' is a recipe describing a task of type (2) above, going from a unit of type 1001 (a firm) to a unit of type 1 (a worker), and invoking a computational step with id code # 1220.

An example of the Java code is given in Figure 2; note that the method uses both internal parameters and matrix references.

The C++ Custom Implementation

The implementation we present here is written for C++, in particular we use Borland C++ Builder 5.0. Our approach consists of two phases: first, the model is coded as a set of classes; second, we decide what sorts of information are displayed to the user. The first phase is independent from C++ compiler used, while the second may rely more on the used compiler and will involve technical aspects not so relevant here. For these reasons we shall focus our attention on the first phase.

Figure 2. Java code implementation for a computational firm

```
Public void c1220(){

      if(pendingComputationalSpecificationSet.
        getNumberOfMemoryMatrixesToBeUsed()!=2)
         {
              System.out.println(« Code -1220 requires 2 matrixes ; «  +
                                  pendingComputationalSpecificationSet.
                                  getNumberOfMemoryMatrixesToBeUsed() +
                                  " found in order # " +
                                  pendingComputationalSpecificationSet.
                                  getOrderNumber());
              MyExit.exit(1);
         }

      rd=4;

      // displacements for the unit memory matrixes coordinates
      mm1=(MemoryMatrix) pendingComputationalSpecificationSet.
         getMemoryMatrixAddress(1);
      layer=pendingComputationalSpecificationSet.
         getOrderLayer();
      urd0=(int) mm1.getValue(layer, 0+rd, 2);
      ucd0=(int) mm1.getValue(layer, 0+rd, 3);
      urd1=(int) mm1.getValue(layer, 0+rd, 4);
      ucd1=(int) mm1.getValue(layer, 0+rd, 5);

      checkMatrixNumber=false;
      c1120();
      checkMatrixNumber=true;
      rd=0;
      urd0=0 ; ucd0=0 ; urd1=0 ; ucd1=0 ;

} // end c1220
```

Figure 3. C++ implementation code for a simulation turn

```
// create order
mydistrict->CreateOrder();
// allocate order or destroy it
mydistrict->AllocateOrder();
// permutate firms and workers
mydistrict->PermutateFirms();
mydistrict->PermutateWorkers();
//  compute intersections  firm/firm and firm/worker
mydistrict->FindFirmsIntersection();
mydistrict->FindFirmWorkerIntersection();
// firms work
mydistrict->HirePWorkers();
// firms consumption
mydistrict->FirmConsumption();
// worker consumption
mydistrict->WorkerConsumption();
// check alive firms and reallocate orders
mydistrict->ReallocateOrders();
// expand firms
mydistrict->ExpandFirmDomain();
// expand population
mydistrict->ExpandPopulation(CBProxWG->Checked);
// Display
mydistrict->FastPaintFlatDistrict(FlagView);
```

The implementation we chose is hierarchical: we modeled a container class called *district* which contains pointers to objects belonging to classes *order, worker,* and *firm*. The container class implements the methods performing each phase of a simulation turn, the graphical display methods, and the interface with the main unit. The code for a turn of simulation is reported in Figure 3.

For sake of brevity technical details are not discussed here; source codes are available from the authors upon request.

A Discussion on the Strengths and Weaknesses of the Different Implementations

While the first approach relies on well-tested libraries and most of the implementation details are hidden from the programmer, the second approach requires almost everything to be built from "ground zero." As a result the second approach is more time consuming and certainly more error prone. On the other side the definitive advantage of the second approach is flexibility both in terms of graphical output and of interactivity. In terms of speed the two versions are comparable; while this seems rather surprising it must be borne in mind that, in its current version, the C++ implementation is not speed optimized. The objective was rather to build a sufficiently reliable code even at the cost of some redundancies. Since the final goal of parallel implementation is the replication of the results, the whole process is certainly beneficial for the theoretical validation of the model. In fact the analysis and comparison of the implementation details resulted in the discussion of the assumptions of the whole model. Furthermore, while in general with parallel implementations some economies of scale in designing at least the "housekeeping" algorithms are

to be expected, this did not happen in our case. The reason is to be found in the fact that the C++ implementation could not benefit from the Swarm libraries. On the other side, however, this may avoid the implementation of ill-designed algorithms.

When comparing the two implementations, the results we obtain are qualitatively the same even if they are different in terms of exact replication. The reasons for these differences are several. Mainly they come from minor modeling differences in the two implementations; secondarily they come from more technical reasons such as the error accumulating when floating-point arithmetic operations are performed, and the different random number generator routines that are used in the two different codes.

As for the first point, while in our case the consequences of floating point arithmetic are not as serious as those described in Polhill et al. (2005), the fact that we perform several branching comparing floating point variables may be one of the reasons leading to different behaviors of our platforms. The other relevant aspect to be discussed is the one concerning random number generators. jESOF code uses well-tested routines internal to the Swarm library and devoted to integer or double precision number, quoting Swarm documentation "The generator has a period close to 2^{19937}, about 10^{6001}, so there is no danger of running a simulation long enough for the generator to repeat itself. At one microsecond per call, it would take about 3.2 10^{5987} years to exhaust this generator. For comparison, the age of the Universe is 'only' 2 10^{10} years! This generator can be asked either for a real number (a variable of type double) between [0,1) or for an integer, which will be uniformly distributed on the range [0,4294967295] = [0,2^{32}-1]". With the C++ implementation we felt we could not rely on the internal random number generator. The reasons for our choice can be found in Press et al. (2002), and we decided to use a Minimal Standard generator with a shuffling algorithm which is reported there as "ran1".

SIMULATION RESULTS

In order to quantify our observations, we introduced several performance measures, specifically number of firms, number of workers, average life for firms, and average life for workers. These simple measures allows for the observation of emergence of districts and their fragility or robustness. As a matter of fact, while the number of agents gives an idea of the expansion of the whole system, it gives little information about the stability of the emerging structures; conversely, average life gives information about stability.

In both platforms we could observe cycles that can be explained as the accumulation and exhaustion of resources. The orders coming from the market and their attribution to different firms determine cycles both in the workforce and the production. While in the relevant literature the frequency of cycles in the labor market is considered a challenging puzzle (see Hall, 2005; Shimer, 2005) in our model, it is possible both to give a temporal dimension to cycles length and to give a theoretical interpretation of their occurrence. If three months of inactivity for real firms are assumed to be sufficient for default, the inactivity parameter we use in the model allows us to find the temporal dimension of our simulations. For instance, when setting the inactivity default parameter to 15 turns, a cycle of 1,200 periods corresponds to about 20 years of actual time; in this span, we could observe about eight turns, and this is not too far from the empirical evidence (see Bruno & Otranto, 2004). Furthermore in order to give a theoretical motivation of cycles, it is possible to

Figure 4. Causal loop for firms without random perturbations

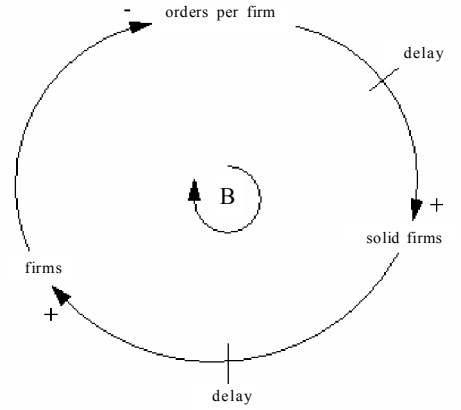


Table 1. Simulation parameters for the two experiments discussed

	Experiment 1	Experiment 2
World dimensions	100 x 100	100 x 100
Number of skills	3	1
Initial number of firms	7	7
Initial number of workers	50	10
New firm generation probability	0.2	0.1
New randomly located worker generation probability	0.8	0.8
New workers generated per turn	27	0
Firm consumption per turn	1	0.1
Firm revenue for delivered order	1	1
Firm initial endowment	5	5
Maximum number of inactivity turns	15	15
Maximum number of undelivered orders	10	10
Maximum firm visibility	unlimited	200
Worker consumption per turn	1	0.5
Worker wage per turn	1	1
Worker initial endowment	5	5

consider a system dynamics approach; if we temporarily do not consider the random effects of the order allocation process, a causal loop describing the firms evolution could be that of Figure 4.

As it is usual for causal loops, variables are related by *causal links* shown by arrows. Each arrow is assigned a polarity, either positive or negative, to indicate how the dependent variable changes when the independent variable changes.

The oscillations are caused by the negative feedback loop and the time delays. The latter determines the adjustment process to continue even after the state of the system reaches its equilibrium, forces the system to adjust too much, and triggers a new correction in the opposite direction; for further details see Sterman (2000).

When considering the stochastic processes $\left\{\tilde{O}^t, \tilde{F}^t, \tilde{W}^t, \quad t \in \mathrm{N}\right\}$, they can be accounted for the random perturbations we can observe. Finally these random perturbations bring us back to the similarities to natural population systems that our model exhibits (e.g., Kaitala et al., 1996).

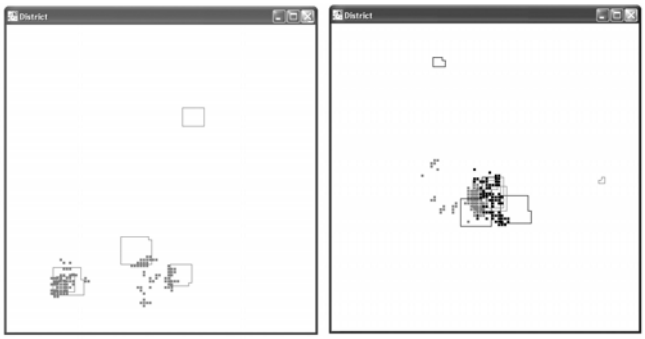
The structures we observed were generally rather fragile. Given the different reasons for default, firms must be able to receive orders to remain in the market, to find workforce, and to deliver (partially) processed orders.

Among the interesting effects we could observe, two deserve a particular discussion, namely localization effects and entry effect. The parameters we considered in our artificial experiments are summarized in Table 1.

A preliminary discussion to the first experiment is in order. While we consider entering firms to be randomly located in the social space, for workers we can consider two possibilities: clustered workers and uniformly located workers.

The rationale for clustered workers is to be found in the fact that workers with the same skills tend to be socially close. This closeness may be interpreted both as the result of social entities such as unions or guilds, or even as the indirect consequence of the process by which the workers learn their skill. In both cases, having clustered workers leads to the emergence of stable districts. This is not surprising since in our model firms and workers are complementary. Obviously when more than one skill is considered, since firms must be able to deliver

Figure 5. Districts emergence in a single (left) and double skill (right) economy with clustered workers



even partially processed orders, stable agglomerates consist of mutually visible firms with all the considered skills and the relative workforces. This is illustrated[3] in Figure 5.

The situation where both firms and workers created random location in the social space is much more interesting: in this case, we may observe the emergence of districts. As we have previously discussed, the evolution of the system is characterized by several short-living firms as a consequence of firms' and workers' fragility. When for some reasons two or more firms with all the skills remain in the world long enough and are mutually visible, they become stable and are able to employ a larger workforce. In this case the compatible workers that are in their scope find adequate support and are able to survive longer: a district emerges (see Figure 6).

We can also observe the consequences of workers' localizations in terms of system development (number of firms). When workers are clustered (by skill type), we observe a slow takeover of the economic system; by contrast when workers are randomly located, we can observe a fast initial development of the economic system with an immediate stability of larger districts. This effect can be easily observed in the Figures 7 and 8 where graphs of workers (stratum 1) and firms (stratum 4) are compared under the two situations.

Another interesting effect is the one occurring when varying the firm entry level. For example consider a simulation with one single skill and assume the entry probability for firms is low. When the simulation is run for a sufficient number of turns, the situation stabilizes as in Figure 9.

We can observe how the number of workers is rather stable and how they are located within firms. At this point, when increasing the entry probability for firms, one would expect both the

Figure 6. District emergence in a double skill economy with randomly located workers

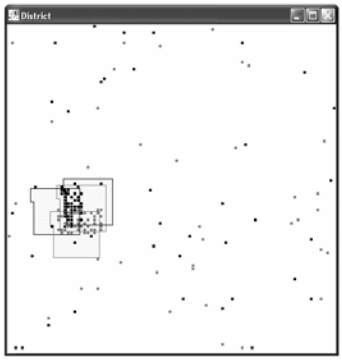


Figure 7. Workers' concentration by type of skill, with a relatively slow takeover of the economic system

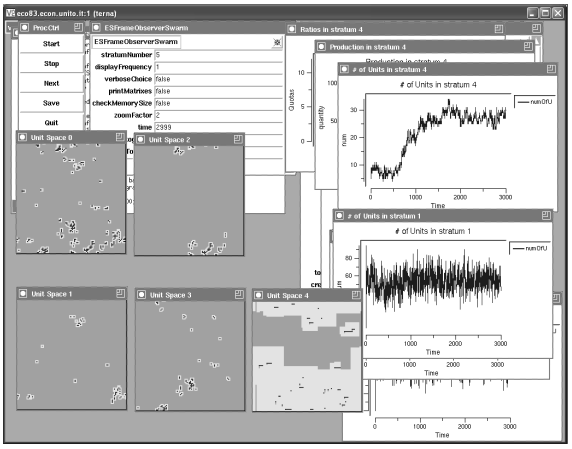


Figure 8. Random diffusion of different skill workers, with a rather fast initial development of the economic system

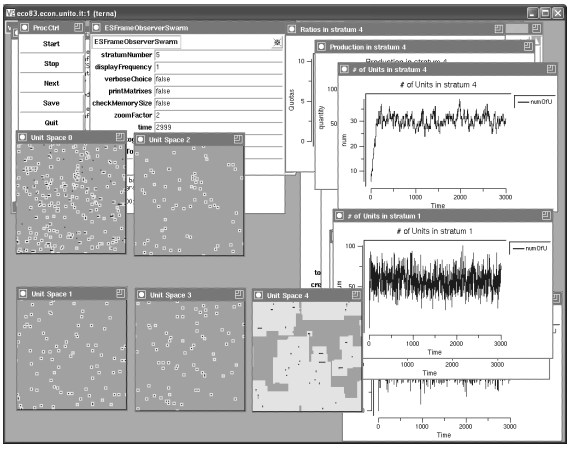
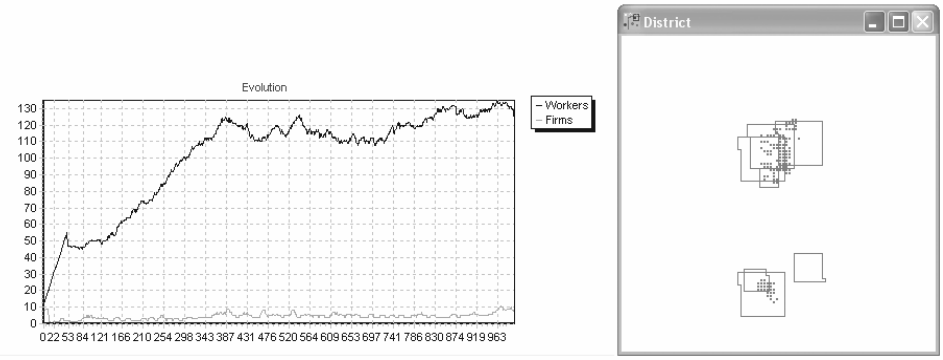
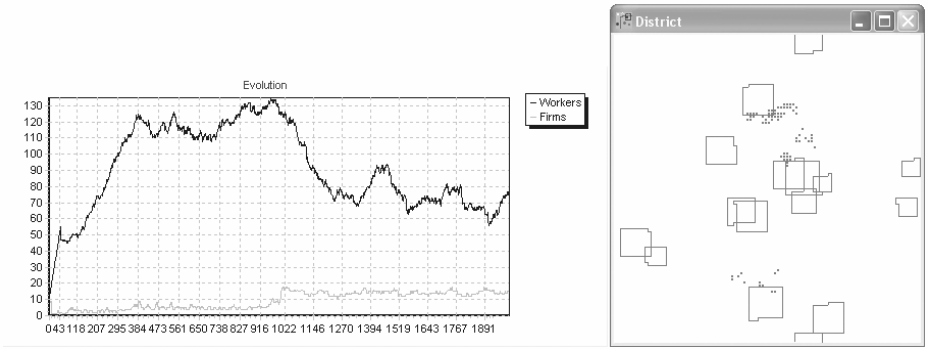


Figure 9. Evolution for 1,000 turns with low entry probability for firms



number of firms and the number of workers to increase. The unexpected result is that this does not happen; on the contrary when increasing the entry probability for firms, while the number of firms increases, the number of workers drops rather abruptly. According to Sterman (2000), one of the causes of *dynamic complexity* lies in the fact that systems are *policy resistant* and as a result "many seemingly obvious solutions to problems fail or actually worsen the situation" (p. 22). In our case, a higher probability for firms to entry increases the number of short-living firms which subtract orders to well-established ones. The fragility of these firms does not allow them to support as many workers as the few stable firms did. This can be observed in Figure 10 where at time 1,000, when the entry probability is increased, while the number of firms rises, the number of workers falls dramatically. Furthermore we

Figure 10. Evolution for 2,000 turns when entry probability for firms increases at turn 1,000



can no longer observe as many workers located within firms as in Figure 5. This is consistent with the empirical literature; for an example relative to Italian textile districts, see Fioretti (2002).

One final note: the cycles we observed in all of our simulations are consistent with the theoretical prediction we presented at the beginning of the session.

CONCLUSION

In this chapter we examined and discussed a model of industrial districts where social agents cooperate symbiotically. The approach we used consisted of the parallel implementation of the model with jESOF and C++. While the model we consider allowed us to find interesting results in terms of the evolution of symbiotic populations, our parallel approach suggests several interesting points per se.

A first result comes from the workers' localization; we found that while clustered workers are not necessary for local district emergence, they actually limit the growth rate of the economic system we considered. On the contrary, after the district emergence we can observe workers' clusterization (see Figures 5 and 6). This raises a not irrelevant question: is district emergence a consequence of workers' clusterization or is it the other way around?

The second effect we found demonstrates the dynamic complexity of our model. In fact we could observe that when increasing the entry probability for firms, while the number of firms increased, the number of workers dropped rather abruptly. This aspect of policy resistance, together with the time delays highlighted in the paragraph devoted to simulation results and the strong interactions between workers and firms, account for the dynamic complexity of the system we studied.

Furthermore we could formulate some theoretical reasons for cycles, confirm them in our simulations, and give them a temporal dimension. Finally we shed some light on the emergence of industrial districts; as a matter of fact, by the decomposition we considered, we could show that the interaction between workers and firms may be sufficient for firms to form districts. This decomposition both in terms of firms/worker and of different skills is coherent with the industrial district literature. In fact, first it develops the "first and fundamental" decomposition Becattini (2003) mentions; second, the results we obtained by considering the different skills and the clusters of workers are

extremely appropriate in terms of how Rullani (2003) considers the territory as a unique resource which stamps specific characteristics, in our case the prevalence of skills.

The double implementation approach allowed the results comparison and a complete discussion of the model assumptions. We were able to evaluate and compare two modeling approaches on the same model. At least in our case, while the jESOF implementation is certainly more straightforward, the C++ implementation is more flexible.

Another interesting aspect was the replicability of the results; while the results we obtained were qualitatively the same, there were differences in terms of exact replication. The reasons for these differences can be found both in the error accumulation process involved in sequences of floating point operation and in the different random generators used.

Finally an important prescriptive lesson arises from our experience. To maximize effectiveness in agent-based modeling, the scientist should first consider implementing a prototype model by a fast development platform such as Netlogo or jESOF in order to assess its feasibility. After this phase the serious researcher should consider both someone else rewriting the model with a more customizable platform (either using a generalized ABM such as Swarm or using object-oriented languages) and reengineering its structure, avoiding the constraints that were imposed by the shell quoted above. These are important steps of modeling discussion and are obviously the first ones in order to encourage scientific repeatability.

In further research we plan to obtain the full replication with both implementations, since this should be the only way to be reasonably certain that most of the coding errors have been properly addressed. This technical goal can be achieved considering either a deterministic version of the model or, more reasonably, considering the same random generator, possibly one where sequences can be replicated. Furthermore it would be interesting to investigate the role of workers' clustering in district emergence.

REFERENCES

Auerswald, P., Kauffman, S., Lobo, J., & Shell, K. (2005). *The production recipes approach to modeling technological innovation: An application to learning by doing.* CAE Working Paper (pp. 98-110).

Axelrod, R. (1984). *The evolution of cooperation.* New York: Basic Books.

Becattini, G., Pyke, F., & Sengenberger, W. (Eds.). (1992). *Industrial districts and interfirm cooperation in Italy.* Geneva: International Institute for Labor Studies.

Becattini G. (2003). From the industrial district to the districtualization of production activity: Some considerations. In F. Belussi, G. Gottardi, & E. Rullani (Eds.), *The technological evolution of industrial districts* (pp. 3-18). Dordrecht: Kluwer Academic.

Becker, P. (2005). Bad pointers. *C/C++ Users Journal, 23*(9), 37-41.

Belussi, F., & Gottardi, G. (Eds.). (2000). *Evolutionary patterns of local industrial systems. Towards a cognitive approach to the industrial district.* Sydney: Ashgate.

Brenner, T. (2002, January 18-19). Simulating the evolution of localized industrial clusters—An identification of the basic mechanisms. *Proceedings of IG2002,* Sophia Antipolis, France. Retrieved November 15, 2005, from http://www.idefi.cnrs.fr/IG2002/papers/Brenner.pdf

Bruno, G., & Otranto, E. (2004). *Dating the Italian business cycle: A comparison of procedures*. ISAE Working Paper 41.

Carbonara, N. (2004). Innovation processes within geographical clusters: A cognitive approach. *Technovation, 24,* 17-28.

Carbonara, N. (2005). Information and communication technology and geographical clusters: Opportunities and spread. *Technovation, 25,* 213-222.

Carley, K. M., & Prietula, M. J. (1994). *Computational organization theory*. Hillsdale, NJ: Lawrence Erlbaum.

Cerruti, U., Giacobini, M., & Merlone, U. (2005, April 4-6). A new framework to analyze evolutionary 2x2 symmetric games. *Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games* (pp. 218-224), Essex University, UK.

Fioretti, G. (2002). *Individual contacts, collective patterns—Prato 1975-97, a story of interactions*. Tinbergen Institute Discussion Paper, TI-2002-109/3.

Flake, G.W. (1988). *The computational beauty of nature.* Cambridge, MA: The MIT Press.

Frank, S. A. (1997). Models of symbiosis. *The American Naturalist, 150,* S80-S99.

Garofoli, G. (1981). Lo sviluppo delle aree "periferiche" nell'economia italiana degli anni '70. *L'Industria, 3,* 391-404.

Garofoli, G. (1991). Local networks, innovation and policy in Italian industrial districts. In E. Bergman, G. Mayer, & F. Tödtling (Eds.), *Regions reconsidered. Economic networks innovation and local development in industrialized countries.* London: Mansell.

Garofoli, G. (1992). Industrial districts: Structure and transformation. In G. Garofoli (Ed.), *Endogenous development and Southern Europe* (pp. 49-60). Avebury: Aldershot.

Gaylord, R. J., & D'Andria, L. J. (1998). *Simulating society*. New York: Springer-Verlag.

Hall, R. E. (2005). Employment fluctuations with equilibrium wage stickiness. *American Economic Review, 95*(1), 50-65.

Hastings, A. (1997). *Population biology. Concepts and models*. New York: Springer-Verlag.

Hofbauer, J., & Sigmund, K. (1998). *Evolutionary games and population dynamics*. Cambridge: Cambridge University Press.

Kaitala, V., Ranta, E., & Lindström, J. (1996). Cyclic population dynamics and random perturbations. *Journal of Animal Ecology, 65*(2), 249-251.

Law, A. M., & Kelton, W.D. (2000). *Simulation modeling and analysis*. Boston: McGraw-Hill.

Lazzaretti, L., & Storai, D. (1999). Il distretto come comunità di popolazioni organizzative. Il caso Prato. *Quaderni IRIS,* (6).

Lazzeretti, L., & Storai, D. (2003). An ecology based interpretation of district "complexification": The Prato district evolution from 1946 to 1993. In F. Belussi ,G. Gottardi, & E. Rullani (Eds.), *The technological evolution of industrial districts* (pp. 409-434). Dordrecht: Kluwer Academic.

Lotka, A. J. (1925). *Elements of physical biology*. Baltimore: Williams & Wilkins.

Piva, M., Santarelli, E., & Vivarelli, M. (2003). *The skill bias effect of technological and organizational change: Evidence and policy implications*. IZA Discussion Paper 934.

Polhill, J. G., Izquieredo, L. R., & Gotts, N. M. (2005). The ghost in the model (and other effects of the floating points arithmetic). *Journal of Artificial Societies and Social Simulation, 8*(1), 5. Retrieved November 5, 2005, from http://jasss.soc.surrey.ac.uk/8/1/5.html

Press, W. H., Teulkolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2002). *Numerical recipes in C++*. Cambridge, UK: Cambridge University Press.

Royama, T. (1992). *Analytical population dynamics.* New York: Chapman and Hall.

Rullani, E. (2003). The industrial district (ID) as a cognitive system. In F. Belussi, G. Gottardi, & E. Rullani (Eds.), *The technological evolution of industrial districts* (pp. 63-88). Dordrecht: Kluwer Academic.

Shimer, R. (2005). The cyclical behavior of equilibrium unemployment and vacancies. *American Economic Review, 95*(1), 25-49.

Sorenson, O., Rivkin, J.W., & Fleming, L. (2004). *Complexity, networks and knowledge flow.* Working Paper.

Squazzoni, F., & Boero, R. (2002). Economic performance, inter-firm relations and local institutional engineering in a computational prototype of industrial districts. *Journal of Artificial Societies and Social Simulation, 5*(1), 1. Retrieved November 15, 2005, from http://jasss.soc.surrey.ac.uk/5/1/1.html

Sterman, J. D. (2000). *Business dynamics. System thinking and modeling for a complex world.* Boston: Irwin McGraw-Hill.

Swarm Development Group. (2005). *Documentation set for Swarm 2.2, random library.* Retrieved June 20, 2005, from http://www.swarm.org

Volterra, V. (1926). Variazioni e fluttuazioni del numero d'individui in specie animali conviventi. *Mem. R. Accad. Naz. dei Lincei, VI*(2).

Zhang, J. (2003). Growing Silicon Valley on a landscape: An agent-based approach to high-tech industrial clusters. *Journal of Evolutionary Economics, 13,* 529-548.

KEY TERMS

Agent-Based Simulation (ABS): A simulation technique where some of the simulated entities are modeled in terms of agents.

Cooperation: The association of persons or businesses for common, usually economic, benefit.

Industrial District: A geographic area containing a number of firms producing similar products, including firms operating at different stages of a production process.

Industrial Organization: The way in which industrial activity is organized into firms and industries, and how that organization is related to economic performance.

ENDNOTES

1. Carbonara (2004) points out how Italian geographical clusters are usually referred to as Industrial Districts.
2. In the jES Open Foundation version of the model, we also have instrumental layers showing separately the presence of workers for each type of skill, but obviously the two implementations are equivalent in terms of modeling.
3. For the sake of brevity, we do not report parameters we used for obtaining Figures 5 and 6. They can be obtained adapting those for the first experiment we discuss.

Chapter XXII
Competitive Advantage of Geographical Clusters

Vito Albino
Politecnico di Bari, Italy

Nunzia Carbonara
Politecnico di Bari, Italy

Ilaria Giannoccaro
Politecnico di Bari, Italy

ABSTRACT

This chapter deals with complexity science issues from two sides: from one side, it uses complexity science concepts to give new contributions to the theoretical understanding of geographical clusters (GCs); from the other side, it presents an application of complexity science tools such as emergent (bottom-up) simulation, using agent-based modeling to study the sources of GC competitive advantage. Referring to the first direction, complexity science is used as a conceptual framework to identify the key structural conditions of GCs that give them the adaptive capacity, so assuring their competitive advantage. Regarding the methodological approach, the agent-based simulation is used to analyze the dynamics of GCs. To this aim, we model the main characteristics of GCs and carry out a simulation analysis to observe that the behaviors of GCs are coherent with the propositions built up on the basis of complexity science literature.

INTRODUCTION

This chapter deals with geographical clusters (GCs), which can be defined as geographically defined production systems, characterized by a large number of small and medium-sized firms involved at various phases in the production of a homogeneous product family. These firms are highly specialized in a few phases of the production process, and integrated through a complex network of inter-organizational relationships (Becattini, 1990; Porter, 1998; Maskell, 2001; Pouder & St. John, 1996).

In particular, the chapter analyzes the sources of competitive advantage of GCs, which is a relevant topic in the referred literature (e.g., Porter, 1998). The latter, in fact, has focused much attention on the reasons explain-

ing the GC competitive success such as: the flexible specialization conceptualized by Piore and Sabel (1984); the localized external economies concept anticipated by Marshall (1920), and further formalized by Becattini (1990) and Krugman (1991); the industrial atmosphere notion conceived by Marshall (1919); and the innovative milieux notion developed by the GREMI (e.g., Maillat, Lecoq, Nemeti, & Pfister, 1995).

These studies have identified the main critical factors governing the success of GC firms. These can be traced back to the following features that successful GCs possess: the physical and cultural proximity of many small and medium-sized firms; the division of labor among firms; the presence within the area of complementary competencies and skills; the high degree of specialization of both firms and workforce; the existence of a dense network of inter-firm relationships where firms cooperate and compete at the same time; the presence of a dense network of social relationships mainly based on face-to-face contacts; and the easy and fast circulation of knowledge and information in the area.

These features, which assure the competitive advantage of GCs when the competitive context is characterized by increasing and not particularly sophisticated demand, seem to be insufficient to guarantee the GC success in the current competitive scenario, which is much more dynamic, unpredictable, and instable. In such a context many GCs are undergoing a decline phase.

As a result, the attention of scholars and policymakers has been shifted and is now much more oriented to develop theories on GC survival in the new competitive scenario by looking for new sources of competitive advantage for GCs (Baptista, 2000; Sull 2003).

Recent studies have in fact pointed out that changes in the GC organizational structure and in their strategies are necessary to guarantee the GC competitiveness. For example, some GCs have internationalized their production system by delocalizing their production process in foreign countries, so determining profound changes in the GC structure (Corò & Rullani, 1998; Biggiero, 2002). Some GCs have introduced new innovation strategies much more focused on developing radical innovations by creating alliances with universities and research centers (Belussi & Arcangeli, 1998; Carbonara, 2004; Corò & Grandinetti, 1999). Therefore, these studies suggest how GCs have to change to survive. This means that the competitive advantage of GCs is associated with a new set of features. Taking things to the extreme, GCs possessing these features are competitive and survive, the others not.

This approach, which is consistent with the traditional studies on GCs, presents some limitations. It adopts a static perspective aimed at identifying a set of features explaining GC competitive advantage in a given particular context. In this way every time the competitive scenario changes, it is necessary to identify a new set of features. In addition, the dynamics that have forced the changes in features cannot be investigated in depth by using this approach. These can in fact be analyzed and identified only after that they are already in practice. Furthermore, this approach considers the entire GC as the unit of analysis, so failing to determine how the global properties of the whole system result from the behaviors of different local parts. New theoretical and methodological approaches overcoming these limitations are then needed.

Complexity science offers both: it investigates properties and behaviors of complex adaptive systems (CASs) and aims to explain how heterogeneous agents "self-organize" to create new structures in interactive systems, with the goal of understanding how such structures emerge and develop (Casti, 1994, 1997; Coveney & Highfield, 1995; Holland, 1995, 1998; Johnson, 2001).

By adopting a complexity science approach, the GC competitive advantage is not the result of a set of pre-defined features characterizing GCs, but it is the result of dynamic processes of adaptability and evolution of GCs with the external environment. In other words, the source of competitive advantage for GCs is not limited to the posses of a given set of features. The competitive advantage resides in the GC capabilities of adaptability and evolution with the external environment. The result of this evolutionary process in terms of new features exhibited from GCs is not known *a priori,* but spontaneously emerges from the interactions among the system components and between them and the environment. This view is consistent with the recent studies on strategic management about new dynamic sources of competitive advantage based on resources, capabilities, and knowledge (Teece, Pisano, & Shuen, 1997).

In this chapter complexity science is specifically used to develop propositions on the key conditions of GCs that give them adaptive capacity, so assuring their competitive advantage.

As regards the methodological approach, complexity science offers a new simulation technique to study CASs and their dynamics, namely agent-based simulation (ABS) (Holland, 1995, 2002). ABS is characterized by a collection of autonomous, heterogeneous, intelligent, and interacting agents that operate and exist in an environment (Epstein & Axtell, 1996). In ABS, agents interact with each other in a non-linear manner with little or no central direction. The large-scale effects determined by the locally interacting agents are called emergent properties of the system. These thus are the spontaneous result of the interactions among agents according to a bottom-up approach, rather than a top-down one. The main goal of ABS is to enrich the understanding of fundamental processes regulating and determining dynamics of CASs (Axelrod, 1997). Furthermore, it is a valuable tool to theory building and to theory testing (Axelrod, 1997; Carley & Gasser, 2000).

In the chapter, agent-based simulation is used to analyze the dynamics of GCs. To this aim, we model the main characteristics of GCs, namely the co-location into a restricted area of many small and medium-sized firms, the specialization of firms in a few phases of the production process, the network of inter-firm relationships, the existence of competition and collaboration among firms, and the exchange of knowledge and information among firms. Then, we use the developed agent-based model to observe via simulation that the behaviors of GCs are coherent with the propositions built up on the basis of complexity science literature.

The chapter is organized as follows. First, we briefly present the theoretical background of this study by reviewing the traditional studies on the sources of GC competitive advantage and the knowledge-based view of GC competitive advantage. Then, we discuss the complexity science approach to GC competitive advantage and formulate the theoretical propositions on the key conditions of GCs that give them adaptive capacity, so assuring their competitive advantage. We then present a brief review of the studies that have used agent-based models to investigate GCs, and we describe the proposed agent-based model developed to analyze the dynamics of GCs and to study their competitive advantage. Finally, simulation analysis is illustrated and results are discussed.

THEORETICAL BACKGROUND

Traditional Sources of Competitive Advantage for Geographical Clusters

Traditional studies on GCs have analyzed the main advantages of GCs that explain the reasons of their success.

In particular, the studies of economic geography have pointed out the benefits associated with the "agglomeration external economies," mainly due to the lower input costs, the development of common suppliers, specialist labor pools, spillover of technical know-how, and the development of a greater comprehension of the workings of the particular industry by individuals and firms (Becattini, 1990; Marshall, 1920).

Studies on industrial economics have highlighted the reduction of the transactional costs due to geographical proximity of firms, and informal and face-to-face contacts among them, as one of the most important benefits of GCs (Mariotti, 1989; Powell, 1987).

Other studies have stressed that one of the key sources of GC competitive advantage is their capacity to develop product and process innovations. In particular, many authors have pointed out that the GC innovative capacity mainly results from the presence of highly specialized technical competencies, the existence of networks of formal and informal relationships, and the geographical proximity that creates an environment wherein information, codes, languages, routines, strategies, and knowledge are easy to transfer and share (Cooke, 1999; Cooke & Morgan, 1998; Henry & Pinch, 2002; Lundvall & Johnson, 1994; Storper, 1997).

Synthesizing the results of these studies, the key source of the GC competitive advantage is the static efficiency, namely cost advantages gained by clustered firms due to a set of features characterizing them: the specialization of firms, the presence of a specialized workforce, the division of labor among firms, the accumulation of specific knowledge in the local area, the networking processes among both the economic and social system, the development of a widespread innovative capacity, the presence into the local area of a common system of social-cultural values.

However, in recent years these factors determining the success of GCs in a competitive context characterized by both increasing and not particularly sophisticated demand seem to be insufficient to guarantee competitive advantage to both the system and its firms. In this new situation, new sources of competitive advantage based not only on the paradigm of the static efficiency are needed.

Knowledge-Based Competitive Advantage of Geographical Clusters

Recent strategic management literature has pointed out that in today's economy the source of sustainable competitive advantage for firms cannot be limited to cost and differentiation advantages, and has recognized the importance of knowledge as a fundamental factor in creating economic value and competitive advantage for firms (Barney, 1991; Grant, 1997; Leonard-Barton, 1995). What a firm knows, how it uses what it knows, and how fast it can develop new knowledge are key aspects for firm success (Hamel & Prahalad, 1994; Prusak, 1997). Therefore, knowledge is a key asset for competing firms, and consequently, learning is a key process. This in fact increases the firm cognitive capital (knowledge stock).

These new strategic management theories have forced new studies on GCs. In particular, in the last few years, some scholars have analyzed the role of knowledge in GCs and proposed a knowledge-based theory of GCs (Malmberg & Maskell, 2004; Maskell, 2001). These works have investigated the nature of knowledge circulated in GCs, the knowledge transfer and creation processes embedded in GCs, and the learning processes activated by firms in GCs (Albino, Carbonara, & Giannoccaro, 2005; Tallman, Jenkins, Henry, & Pinch, 2004). This superior capacity of GCs to

support processes of learning and knowledge transfer and creation has been identified as the key source of their competitive advantage.

Opposite of the traditional studies on GCs where the source of competitive advantage is static based on the possess of given features, in these knowledge-based studies on GCs the competitive advantage results from dynamic processes activated by GC firms, namely the learning and knowledge management processes.

In line with this new perspective, we seek new dynamic sources of competitive advantage by adopting a different theoretical approach, namely complexity science.

THE COMPLEXITY SCIENCE APPROACH TO GC COMPETITIVE ADVANTAGE

Complexity Science Studies and Their Dynamics

CASs consist of an evolving network of heterogeneous, localized, and functionally integrated interacting agents. They interact in a non-linear fashion, can adapt and learn, thereby evolving and developing a form of self-organization that enables them to acquire collective properties that each of them does not have individually. CASs have adaptive capability and co-evolve with the external environment, modifying it and being modified (Axelrod & Cohen, 1999; Choi, Dooley, & Rungtusanatham, 2001; Gell-Mann, 1994; Lane, 2002).

During the 1990s, there was an explosion of interest in complexity science as it relates to organizations and strategy. The complexity science offers a number of new insights that can be used to seek new dynamic sources of competitive advantage. In fact, application of complexity science to organization and strategy identifies key conditions that determine the success of firms in changing environments associated with their capacity to self-organize and create a new order, learn, and adapt (Levy, 2000; McKelvey & Maguire, 1999; Mitleton-Kelly, 2003).

In particular, complexity science suggests what causes emergent order and self-organization inside organizations. Kauffman (1993) claims that organizations poised at the edge of chaos might give rise to self-organization and emergent order, which enable them to succeed in an era of rapid change. Extending the arguments on the requisites of internal variety of a system developed by Ashby (1956), McKelvey (2004) observes that organizations have to have an internal variety that matches their external variety so as to self-organize and adapt to the changing environment.

In line with the works above, complexity science is used in this study to identify what conditions of GCs enable them to adapt to the external environment. Therefore, the basic assumption of this study is that GCs are CASs, given that they exhibit different properties of CASs such as the existence of different agents (e.g., firms and institutions), the non-linearity, different types of interactions among agents and between agents and the environment, distributed decision making, decentralized information flows, and adaptive capacity (Albino et al., 2005).

In the following, three theoretical propositions concerning the GC adaptive capacity are formulated by using CAS theory. First, we identify the three main properties of CASs that affect the adaptive capacity—namely the interconnectivity, the heterogeneity, and the level of control—and we define how the value of these properties influence the adaptive capacity. Then, we associate these properties with specific GC characteristics, so obtaining the key conditions of GCs that give them adaptive capacity, so assuring their competitive advantage.

So doing, GC competitive advantage is not the result of a set of pre-defined features characterizing GCs, but it is the result of adaptability process of GCs with the external environment.

Interconnectivity

CAS theory identifies the number of interconnections within the system as a critical condition for self-organization and emergence. Kauffman (1995) points out that the number of interconnections among agents of an ecosystem influences the adaptive capacities of the ecosystem. He uses the NK model to investigate the rate of adaptation and level of success of a system in a particular scenario. The adaptation of the system is modeled as a walk on a landscape. During the walk, agents move by looking for positions that improve their fitness represented by the height of that position. A successful adaptation is achieved when the highest peak of the landscape is reached. The ruggedness of the landscape influences the rate of adaptation of the system. When the landscape has a very wide global optimum, the adaptive walk will lead toward the global optimum. In a rugged landscape, given that there are many peaks less differentiated, the adaptive walk will be trapped on the many suboptimal local peaks.

By using the concept of tunable landscape and the NK model, Kauffman (1995) demonstrates that the number of interconnections among agents (K) influences the ruggedness of the landscape. As K increases, the ruggedness raises and the rate of adaptation decreases. Therefore, in order to assure the adaptation of the system to the landscape, the value of K should not be high.

This result has been largely applied in organization studies to modeling organizational change and technological innovation (Kauffman, Lobo, & Macready, 2000; Levinthal, 1997; Rivkin & Siggelkow, 2002). In organization studies the K parameter has an appealing interpretation, namely, the extent to which components of the organization affect each other.

Similarly, it can be used to study the adaptation of GCs, by considering that the level of interconnectivity of GCs is determined by the social and economic links among the GC firms. When the number of links among firms is high, the behavior of a particular firm is strongly affected by the behavior of the other firms.

On the basis of the discussion above, we formulate the following proposition:

Proposition 1. A medium number of links among GC firms assures the highest GC adaptive capacity.

Heterogeneity

Different studies on complexity highlight that variety destroys variety. As an example, Ashby (1956) suggests that successful adaptation requires a system to have an internal variety that at least matches environmental variety. Systems having agents with appropriate requisite variety will evolve faster than those without. The same topic is studied by Allen (2001), LeBaron (2001), and Johnson (2000). Their agent-based models show that novelty, innovation, and learning all collapse as the nature of agents collapses from heterogeneity to homogeneity. Dooley (2002) states that one of the main properties of a complex system that supports the evolution is diversity. Such a property is related to the fact that each agent is potentially unique not only in the resources that they hold, but also in terms of the behavioral rules that define how they see the world and how they react. In a complex system, diversity is the key towards survival. Without diversity, a complex system converges to a single mode of behavior.

Referring to firms, the concept of agent heterogeneity can be associated to competitive

strategy of firms. This in fact results from the resources that a firm possesses, and defines the behavior rules and the actions of firms in the competitive environment (Grant, 1998).

Therefore, we assume that:

Proposition 2. The greater the differentiation of the competitive strategies adopted by GC firms, the higher the GC adaptive capacity.

Level of Control

The governance of a system is a further important characteristic influencing CAS self-organization and adaptive behaviors.

Le Moigne (1990) observes that CASs are not controlled by a hierarchical command-and-control center and manifest a certain form of autonomy. The latter is necessary to allow evolution and adaptation of the system. A strong control orientation tends to produce tall hierarchies that are slow to respond (Carzo & Yanousas, 1969) and invariably reduce heterogeneity (Jones, 2000; Morgan, 1997). The presence of "nearly" autonomous subunits characterized by weak but not negligible interactions is essential for the long-term adaptation and survival of organizations (Sanchez, 1993; Simon, 1996). Furthermore, Granovetter's (1973) research finding is that novelty and innovation happen more frequently in networks consisting mostly of "weak ties" as opposed to "strong ties." The latter tend to produce group thinking.

The level of control in GCs is determined by the governance of the GC organizational structure. The higher the degree of governance, the higher the level of control exerted by one or more firms on the other GC firms (Storper & Harrison, 1992).

Therefore, we assume that:

Proposition 3. A medium degree of governance of the GC organizational structure improves the GC adaptive capacity.

THE AGENT-BASED MODEL OF GEOGRAPHICAL CLUSTERS

The second way we use complexity science in the chapter concerns the methodological approach, namely the agent-based simulation that is a technique developed to study CASs. We use it to observe via simulation that the behaviors of GCs are coherent with the propositions above.

Recently, a small number of studies have applied agent-based computational models to study GCs. Albino et al. (2006) propose a multi-agent system model to study cooperation and competition in the supply chain of a GC. They carry out a simulation analysis to prove the benefits of a selected kind of cooperation for the GCs, and to evaluate the benefits of the cooperation in different competitive scenarios and diverse GC organizational structures.

Boero and Squazzoni (2002) suggest an agent-based computational approach to describe the adaptation of GCs to the evolution of market and technology environments. Their model considers three different evolutionary technological regimes, with growing production costs sustained by firms and increasing performance shaped by the market. The district firms need to adapt to this evolution. In particular, the authors study how the GC organizational structure affects the economic performance (e.g., level of firms surviving in the market over time) and technological adaptation of firms over time. Moreover, they compare two different GC organizational structures: the market-like and the partnership district.

Brusco, Minerva, Poli, and Solinas (2002) develop a three-dimensional cellular automaton model of a GC which represents the dynamic by which the GC firms share information about technology, markets, and products. In the model, different scenarios characterized by different degrees of information sharing among firms are compared. In particular, the following performances are measured: the dimensional growth

of both the firm and the system over time, industry concentration, and wealth sharing.

Zhang (2002) studies the formation of the high-tech industrial clusters, such as the Silicon Valley, showing that the high-tech clusters can spontaneously emerge as a result of specific dynamics. He emphasizes that high-tech industrial clusters are characterized by concentrated entrepreneurship. The cluster is then explained by the social effect through which the appearance of one or a few entrepreneurs may inspire many followers locally. Once a cluster comes into existence, it tends to reinforce itself by attracting more firms. The author proposes a Nelson-Winter model with an explicitly defined landscape and uses agent-based computational modeling to show the cluster dynamics.

Fioretti (2001) develops a spatial agent-based computational model to study the formation and the evolution of the Prato industrial district. He focuses on the structure of the information flows that underlie the creation of production chains within the district and on the role of the middlemen.

Brenner (2001) develops a cellular automata model of the spatial dynamics of entry, exit, and growth of firms within a region. He first identifies the mechanisms that modify a firm's state and then carries out a simulation analysis to investigate the influence of all the mechanisms on the geographic concentration.

The Agent-Based Model to Study the Competitive Advantage of Geographical Clusters

In this section the agent-based model that we have developed to analyze the dynamics of GCs and to study their competitive advantage is described.

We propose a stylized model, built on the GC literature, by considering some of the main GC features defined in the models of flexible specialization by Piore and Sabel (1984), localized external economies (Becattini, 1987; Marshall, 1920), and industrial atmosphere by Marshall (1919). In particular, we take into account the co-location into a restricted area of many small and medium-sized firms, the specialization of firms in a few phases of the production process, the network of inter-firm relationships, the existence of competition and collaboration among firms, and the exchange of knowledge and information among firms. How we reproduce these building blocks in the model is described later on.

The agent-based model of the GC consists of agents and an environment. Agents perform actions to accomplish their own goals. Two types of agents are considered, namely the firm-agent and the market-agent. The environment represents the geographic local area in which the agents operate. The agent-based model is built using the software *Agentsheets* (http://agentsheets.com/).

Agents

Firm Agents

Firm-agents stand for the firms inside the GC. They are characterized by a set of properties and actions which are defined looking at the literature on GCs and strategic management. We assume that the goal of the firm is to survive. The survival is determined by the satisfaction of customer needs. To accomplish this goal, firms adopt a competitive strategy. This competitive strategy can be modified over time (Grant, 1998). Competitive strategy is then based on knowledge and competencies possessed by the firms (Hamel & Prahalad, 1991). We assume that the knowledge about customers are relevant to survive.

Firm-agents are classified on the basis of the size and the performed phase of the production process. In particular, we distinguish small firm-agents and medium firm-agents. Moreover, we distinguish firm-agents into *client-*

agents, those that perform the final phase of the production process, and *supplier-agents,* those that carry out an intermediate phase of the production process.

The firm-agent is modeled by an object located into a grid. The firm-agent is characterized by a moving direction (up, down, right, and left) and a color (grey, pink, or red). The position into the grid changes at each simulation step according to the moving direction. The moving direction represents the competitive strategy. This can be successful if it allows the firm-agent to satisfy the customer needs. The red color represents the knowledge of the market. At the begin of the simulation, all firm-agents are grey—namely they do not have knowledge on the market, then they cannot satisfy the customer needs.

Market Agents

Market-agents stand for customer needs that GC firms must satisfy. They search for firms able to satisfy them. Among these they will randomly select one.

Actions

In the following, the actions performed by firm-agents are described. These actions have been built based on the literature on strategic management and GCs. We define each action as a stylized fact derived from literature.

Looking for the Customer

At each simulation step the firm-agent moves from the current position into a new one. In particular, the client-agent looks for the market-agent, and the supplier-agent looks for the client-agent. The new position is determined by the moving direction—namely the firm-agent will occupy the adjacent cell coherently with its moving direction. The moving direction represents the firm-agent competitive strategy, therefore firm-agent movement into the grid represents the result of the strategy implementation. This action is modeled on the basis of the competitive behavior of firms. This is defined by the competitive strategy and aims at satisfying the customer needs.

Forming the Supplier-Client Relationship

At each simulation step the client-agent scans the environment looking for an adjacent supplier-agent that is needed to carry out the final product. When they meet each other, they form a supplier-client relationship and can serve the market.

Acquiring Knowledge on the Market

At each simulation step the client-agent scans the environment looking for an adjacent market-agent. When the client-agent meets the market-agent, it acquires knowledge about the market. This knowledge allows it to satisfy the customer needs and then to become successful. When the firm-agent knows the customer needs, it changes its color. In particular, if the firm-agent is a small one, we assume that it is able to acquire just a piece of knowledge on the market. In the case of medium firm-agent, all knowledge on the market is acquired. In the first case the small firm-agent changes its color to pink, in the second case to red.

This different behavior depends on the different absorptive capacity characterizing small and medium-sized firms. According to Cohen and Levinthal (1990), the greater the amount of firm resources, the greater its absorptive capacity.

Changing the Competitive Strategy

We consider that the firm-agent can change the competitive strategy. In fact, the geographical proximity of firms and the presence of a dense

network of inter-firm relationships facilitate the circulation of information favoring the imitation process. In particular, GC firms tend to emulate the strategies of the most successful firms (Belussi & Arcangeli, 1998; Carbonara, 2002). Therefore, we assume that when the client-agent (supplier-agent) meets another client-agent (supplier-agent), it can imitate its competitive strategy—that is, it can change its moving direction, assuming the moving direction of the met firm-agent.

Learning Customer Needs by Interacting

Firm-agents can learn customer needs not only when they meet the market-agent, but also by interacting with a successful client-agent, one that knows the customer needs. In particular, we assume that both client-agents and supplier-agents when they meet a successful client-agent can acquire knowledge on the market, changing their color. The same assumption on the different capacity of firm-agents to acquire knowledge on the market on the basis of their size holds.

Dying

Due to the strong competition in a narrow area, we assume that the firm-agent can die. This is reasonable given that when the number of competitors increases, competition becomes more intense and firms more aggressively compete so determining the death of the less competitive firms.

The Geographical Cluster Building Blocks in the Model

Co-Location into a Restricted Area of Small and Medium-Sized Firms

This GC building block is modeled by considering different agents positioned in a grid. Agents represent GC firms, and the grid represents the geographical bounded area. To take into account the different size of firms, we have introduced two kinds of agents: small firm-agents and medium firm-agents.

Specialization of Firms in a Few Phases of the Production Process

The different specialization of firms, due to the labor division characterizing the GC production model, is modeled by distinguishing the firm-agents into client-agents and supplier-agents.

Network of Inter-Firm Relationships

The network of inter-firm relationships involves client- and supplier-agents. These two types of firm-agents need to interact and to form supplier-client relationship to carry out the final product and serve the market.

Existence of Competition and Collaboration among Firms

Referring to this CG characteristic, in the model we have considered the horizontal competition that takes place between similar firms serving the same market, and the vertical cooperation that takes place between different firms operating along the supply chain. In particular, we have modeled the horizontal competition by considering that a firm-agent dies when in its neighbor there are many successful firm-agents (more than three). The vertical cooperation is modeled by allowing client-agents and supplier-agents to share knowledge and information.

Exchange of Knowledge and Information among Firms

This building block is modeled by allowing firm-agents to learn from other firm-agents the knowledge on the market and their successful competitive strategies.

SIMULATION ANALYSIS

The simulation analysis is defined coherently with the three theoretical propositions, and it is aimed at verifying whether the behavior of GCs is consistent with our propositions.

To do this, we first need to define into the agent-based model of GC the variables of the propositions, namely the number of links among GC firms, the differentiation of competitive strategies adopted by GC firms, and the degree of governance of the GC organizational structure.

Number of Links among GC Firms

In the agent-based model we have represented this variable through the probability that firm-agents meet each other. When the probability is zero, firm-agents that are adjacent do not meet each other. The higher this probability, the higher the number of links among GC firms. Therefore, when the number of links increases, there is a higher influence among agents, given that there is higher probability that agents exchange the competitive strategy and the knowledge on market.

Differentiation of Competitive Strategies

In our agent-based model, the competitive strategy is an agent property modeled by its moving direction that defines how the firm agent moves into the grid. The level of differentiation of competitive strategies is modeled by assigning different moving directions to firm-agents at the beginning of simulation.

Degree of Governance of the GC Organizational Structure

We model the degree of governance by defining a number of agents that are linked by strong ties. The level of governance increases as the number of agents linked by strong ties increases. We assume that agents linked by strong ties exchange competitive strategy and knowledge on the market with high probability (100%). On the contrary, if the agents are not linked by strong ties, the probability is lower (30%).

Plan of Experiments and Output of Simulation

The plan of experiments consists of 64 experiments defined by assuming four different values for each variable, namely the number of links among GC firms, the level of differentiation of competitive strategies adopted by GCs firms, and the degree of governance of the GC organizational structure. In particular, we assume four increasing values of the probability that the agent meets another agent (i.e., 0%, 33%, 67%, and 100%). When the value is 0%, the agent never meets another agent. When the value is 100%, every time two agents are adjacent, they meet each other.

Regarding the level of differentiation of the competitive strategy, we assume that agents can have from one up to four different moving directions (i.e., up, down, left, and right). In particular, all agents can have the same moving directions (up), can have two different moving directions (50% of agents go up and 50% of agents go down), can have three different moving directions (33% of agents go up, 33% go down, and 34% of agents go left), and finally can have four different moving directions (25% of agents go up, 25% go down, 25% of agents go left, and 25% go right).

Regarding the degree of governance of the GC organizational structure, we assume four increasing values for the number of agents linked by strong ties (i.e., 0%, 33%, 67%, and 100%). When the value is 0%, none of the agents are linked by strong ties. When the value is 100%, all the agents are linked by strong ties.

At the end of simulation (after 2,000 steps), we measure the percentage of final successful firm agents on the total (*N%*), namely the final red agents, which is a proxy of the GC competitive success in a particular competitive context.

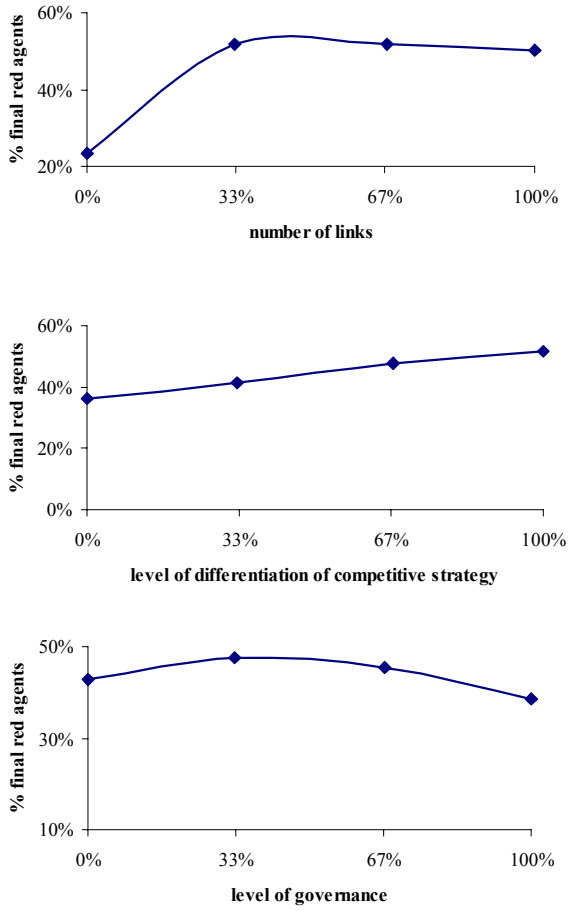
Results

Results are summarized in Figure 1 by showing the percentage of final successful firms as the number of links, the differentiation of competitive strategies, and the degree of governance increase. Simulation results confirms the three propositions.

Regarding the first one, the number of final successful firms in the first run raises as the number of links increases, takes the maximum in 33%, and then slightly decreases. Thus, a medium number of links assures the highest number of survival firms. To a certain extent, this result may be empirically confirmed by looking at the evolution of the most competitive Italian GCs, which moved from a configuration characterized by a dense network of formal and informal inter-firm and social relationships (development stage) to another one characterized by structured inter-firm networks (maturity stage) (Carbonara, Giannoccaro, & Pontrandolfo, 2002).

Referring to the second proposition, simulation results show that the higher the level of differentiation on competitive strategies among firms, the higher the number of final successful firms. The result is empirically verified by several cases of successful GCs characterized by a high diversification among firms, such as the media-cluster in London (Grabher, 2002; Nachum & Keeble, 2003) and in New York City (Krätke, 2003), the high-tech cluster in Cambridge (DTI, 2001; Garnsey & Heffernan, 2005), and the sports shoes cluster of Montebelluna, Italy, (Durante, 2004; OSEM, 2004).

Figure 1. Simulation results



Finally, with regard to the third proposition, simulation results confirm that in the first run, as the degree of governance increases, the number of final successful firms raises, but then decreases. Therefore, a medium degree of governance of GC organizational structure balancing the complete autonomy of firms has a positive effect on the competitiveness. For example, in some GCs such a level of control is exercised by large firms with a leader position in GCs through hierarchical inter-firm relationships, or by organizations with a role of "meta-management" of GC firms (consortia) or by institutions (local government, trade associa-

tions, unions, and local banks). However, the outcome of simulation shows that by increasing the degree of governance over a certain value, the GC competitiveness decreases. Therefore, it is important to preserve the autonomy characterizing GC firms, as this creates variety, competition, cooperation, and continuous learning.

CONCLUSION

This chapter has dealt with complexity science issues from two sides: from one side, it has used complexity science concepts to give new contributions to the theoretical understanding of geographical clusters (GCs); from the other side, it presents an application of complexity science tools such as emergent (bottom-up) simulation, using agent-based modeling to study the sources of GC competitive advantage.

Referring to the first direction, the complexity science has been used as a conceptual framework to investigate the reasons for the success of GCs. This approach is particularly valuable given that it allows the limits of traditional studies on GCs to be overcome. In particular, the GC competitive advantage is not the result of a set of pre-defined features characterizing GCs, but it is the result of dynamic processes of adaptability and evolution of GCs with the external environment. Therefore, GCs' success is linked to the system adaptive capacity that is a key property of a complex adaptive system (CAS).

Using the theory of CAS, the key conditions of GCs that give them the adaptive capacity have been identified, namely the number of links among GC firms, the level of differentiation of competitive strategies adopted by GCs firms, and the degree of governance of the GC organizational structure. The theory of CAS has then been used to identified the value that these variables should have to increase the system adaptive capacity. In this way, three theoretical propositions concerning GC adaptive capacity have been formulated.

The second way complexity science has been used in the chapter has regarded the methodology. We have adopted agent-based simulation, which is a technique developed to study CASs and their behaviors. This is particularly useful when the system to be investigated is characterized by non linearity, emergence, and evolution.

In particular, we carried out an agent-based simulation analysis to observe whether the behaviors of GCs were coherent with the theoretical propositions. We first developed a stylized agent-based model of GC based on the literature that reproduces the relevant properties of GCs—namely the co-location into a restricted area of many small and medium-sized firms, the specialization of firms in a few phases of the production process, the network of inter-firm relationships, the existence of competition and collaboration among firms, and the exchange of knowledge and information among firms. Then we have conducted a simulation analysis to investigate the influence on the GC competitive success of the number of links among GC firms, of the level of differentiation of competitive strategies adopted by GC firms, and of the degree of governance of the GC organizational structure.

Simulation results have confirmed the theoretical propositions showing that: (1) a medium number of links among GC firms assures the highest number of successful firms; (2) the more differentiated the GC competitive strategies, the higher the number of successful firms, and (3) a limited degree of governance of the GC organizational structure determines the highest number of successful firms.

Further research should be devoted to investigating real cases of GCs so as to confirm with empirical data these results.

REFERENCES

Albino, V., Carbonara, N., & Giannoccaro, I. (2005). Industrial districts as complex adaptive systems: Agent-based models of emergent phenomena. In C. Karlsson, B. Johansson, & R. Stough (Eds.), *Industrial clusters and inter-firm networks* (pp. 73-82). Cheltenham, UK: Edward Elgar.

Albino, V., Carbonara N., & Giannoccaro, I. (2006). Supply chain cooperation in industrial districts: A simulation analysis. *European Journal of Operational Research,* forthcoming.

Allen, P. M. (2001). A complex systems approach to learning, adaptive networks. *International Journal of Innovation Management, 5,* 149-180.

Ashby, W.R. (1956). *An introduction to cybernetics.* London: Chapman & Hall.

Axelrod, R. (1997). Advancing the art of simulation in social sciences. In R. Conte, R. Hegselmann, & P. Terna (Eds.), *Simulating social phenomena.* Berlin: Springer-Verlag.

Axelrod, R., & Cohen, M.D. (1999). *Harnessing complexity: Organizational implications of a scientific frontier.* New York: The Free Press.

Baptista, R. (2000). Do innovations diffuse faster within geographical clusters? *International Journal of Industrial Organization, 18,* 515-535.

Barney, J. B. (1991). Firm resources and sustained competitive advantage. *Journal of Management, 17,* 99-120.

Becattini, G. (1987). *Mercato e forze locali: Il distretto industriale.* Bologna: Il Mulino.

Becattini, G. (1990). The Marshallian industrial district as a socio-economic notion. In G. Becattini, F. Pyke, & W. Sengenberger (Eds.), *Industrial districts and inter-firm co-operation in Italy* (pp. 37-52). Geneva: International Institute for Labor Studies.

Belussi, F., & Arcangeli, F. (1998). A typology of networks: Flexible and evolutionary firms. *Research Policy, 27*(4), 415-428.

Biggiero, L. (2002). The location of multinationals in industrial districts: Knowledge transfer in biomedicals. *Journal of Technology Transfer, 27,* 111-122.

Boero, R., & Squazzoni, F. (2002). Economic performance, inter-firm relations and local institutional engineering in a computational prototype of industrial districts. *Journal of Artificial Societies and Social Simulation, 5*(1).

Brenner, T. (2001). Simulating the evolution of localized industrial clusters—An identification of the basic mechanism. *Journal of Artificial Societies and Social Simulation, 4*(3).

Brusco, S., Minerva, T., Poli, I., & Solinas, G. (2002). Un automa cellulare per lo studio del distretto industriale. *Politica Economica, 18*(2).

Carbonara, N. (2002). New models of inter-firm network within industrial district. *Entrepreneurship and Regional Development, 14,* 229-246.

Carbonara, N. (2004). Innovation processes within geographical clusters: A cognitive approach. *Technovation, 24*(1), 17-28.

Carbonara, N., Giannoccaro, I., & Pontrandolfo, P. (2002). Supply chains within industrial districts: A theoretical framework. *International Journal of Production Economics, 76*(2), 159-176.

Carley, K. M., & Gasser, L. (2000). Computational organizational theory. In G. Weiss (Ed.), *Multi-agent systems. A modern approach to*

distributed artificial intelligence. Cambridge, MA: The MIT Press.

Carzo, R., & Yanousas, J.N. (1969). Effects of flat and tall structure. *Administrative Science Quarterly, 14,* 178-191.

Casti, J.L. (1994). *Complexification.* New York: HarperCollins.

Casti, J.L. (1997). *Would-be worlds: How simulation is changing the frontiers of science.* New York: John Wiley & Sons.

Choi, T. Y., Dooley, K.J., & Rungtusanatham, M. (2001). Supply networks and complex adaptive systems: Control vs. emergence. *Journal of Operations Management, 19,* 351-366.

Cohen, W., & Levinthal, D. (1990). Absorptive capacity: A new perspective on learning and innovation. *Administrative Science Quarterly, 35,* 128-152.

Cooke, P. (1999). The cooperative advantage of regions. In T. Barnes & M. Gertler (Eds.), *The new industrial geography: Regions, regulation, and institutions* (pp. 54-73). London: Routledge.

Cooke, P., & Morgan, K. (1998). *The associational economy: Firms, regions, and innovation.* Oxford: Oxford University Press.

Corò, G., & Grandinetti, R. (1999). Evolutionary patterns of Italian industrial districts. *Human Systems Management, 18*(2), 117-130.

Corò, G., & Rullani, E. (Eds.). (1998). *Percorsi locali di internazionalizzazione. Competenze e auto-organizzazione nei distretti industriali del Nord-est.* Milano: F. Angeli.

Coveney, P., & Highfield, R. (1995). *Frontiers of complexity: The search for order in a chaotic world.* New York: Ballantine.

Cowan, G. A., Pines, D., & Meltzer, D. (Eds.). (1994). Complexity: Metaphors, models, and reality. *Proceedings of the Santa Fe Institute* (Vol. XIX). Reading, MA: Addison-Wesley.

Department of Trade and Industry. (2001). *Business clusters in the UK: A first assessment.* London: Department of Trade and Industry.

Dooley, K. J. (2002). Organizational complexity. In M. Warner (Ed.), *International encyclopedia of business and management.* London: Thompson Learning.

Durante, V. (2004). *Sportsystem, tra fashion e performance.* Montebelluna: Danilo Zanetti.

Epstein, J. M., & Axtell, R. (1996). *Growing artificial societies from the bottom-up.* Cambridge, MA: MIT Press.

Fioretti, G. (2001). Information structure and behavior of a textile industrial district. *Journal of Artificial Societies and Social Simulation, 4*(4).

Garnsey, E., & Heffernan, P. (2005). High-technology clustering through spin-out and attraction: The Cambridge case. *Regional Studies, 39*(8), 1127-1144.

Gell-Mann, M. (1994). *The quark and the jaguar.* New York: Freeman & Co.

Grabher, G. (2002). The project ecology of advertising: Tasks, talents and teams. *Regional Studies, 36*(3), 245-262.

Granovatter, M. (1973). The strength of weak ties. *American Journal of Sociology, 78,* 1360-1380.

Grant, R.M. (1997). The knowledge-based view of the firm: Implications for management in practice. *Long-Range Planning, 30*(3), 450-454.

Grant, R.M. (1998). *Contemporary strategy analysis. Concepts, techniques, applications.* Oxford: Blackwell.

Hamel, G., & Prahalad, C. K. (1994). *Competing for the future.* Boston: Harvard Business School Press.

Henry, N., & Pinch, S. (2002). Spatializing knowledge: Placing the knowledge community of Motor Sport Valley. In A. S. Huff & M. Jenkins (Eds.), *Mapping strategic knowledge* (pp. 137-169). London: Sage.

Holland, J. H. (1998). *Emergence: From chaos to order.* Cambridge, MA: Perseus.

Holland, J. H. (1995). *Hidden order: How adaptation builds complexity.* Reading, MA: Addison-Wesley.

Holland, J. H. (2002). Complex adaptive systems and spontaneous emergence. In A. Q. Curzio & M. Fortis (Eds.), *Complexity and industrial clusters.* Heidelberg: Physica-Verlag.

Johnson, N. L. (2000). Developmental insights into evolving systems: Roles of diversity, non-selection, self-organization, symbiosis. In M. Bedau (Ed.), *Artificial life VII.* Cambridge, MA: MIT Press.

Johnson, S. (2001). Emergence: *The connected lives of ants, brains, cities, and software.* New York: Scribner.

Jones, G. R. (2000). *Organizational theory.* Reading, MA: Addison-Wesley.

Kauffman, S. A. (1993). *The origins of orders: Self-organization and selection in evolution.* New York/Oxford: Oxford University Press.

Kauffman, S. A. (1995). *At home in the universe: The search for laws of self-organization and complexity.* New York: Oxford University Press.

Kauffman, S. A., Lobo, J., & Macready, W. G. (2000). Optimal search on a technology landscape. *Journal of Economic Behavior and Organization, 43,* 141-166.

Krätke, S. (2003). Global media cities in a worldwide urban network. *European Planning Studies, 11*(6), 605-628.

Krugman, P. R. (1991). *Geography and trade.* Cambridge, MA: MIT Press.

Lane, D. (2002). Complexity and local interactions: Towards a theory of industrial districts, complexity and industrial districts. In A. Q. Curzio & M. Fortis (Eds.), *Complexity and industrial clusters.* Heidelberg: Physica-Verlag.

Le Moigne, J. L. (1990). *La modélisation des systèmes complexes.* Paris: Dunod.

LeBaron, B. (2001). Financial market efficiency in a co-evolutionary environment. *Proceedings of the Workshop on Simulation of Social Agents: Architectures and Institutions* (pp. 33-51), Argonne National Laboratory and The University of Chicago.

Leonard-Barton, D. (1995). *Wellsprings of knowledge.* Boston: Harvard Business School Press.

Levinthal, D. A. (1997). Adaptation on rugged landscapes. *Management Science, 43,* 934-950.

Levy, D. L. (2000). Applications and limitations of complexity theory in organization theory and strategy. In J. Rabin, G. J. Miller, & W. B. Hildreth (Eds.), *Handbook of strategic management.* New York: Marcel Dekker.

Lundvall, B., & Johnson, B. (1994). The learning economy. *Journal of Industry Studies, 1*(2), 23-42.

Maillat, D., Lecoq, B., Nemeti, F., & Pfister, M. (1995). Technology district and innovation: The case of the Swiss Jura Arc. *Regional Studies, 29*(3), 251-263.

Malmberg, A., & Maskell, P. (2004). The elusive concept of localization economies: Towards a knowledge-based theory of spatial clustering. In G. Grabher & W. W. Powell (Eds.), *Networks.* Cheltenham, UK: Edward Elgar.

Mariotti, S. (1989). Efficienza dinamica e sistemi di imprese. *Economia e Politica Industriale, 64,* 91-123.

Marshall, A. (1919). *Industry and trade.* London: Macmillan.

Marshall, A. (1920). *Principles of economics.* London: Macmillan.

Maskell, P. (2001). Towards a knowledge-based theory of the geographical cluster. *Industrial and Corporate Change, 10,* 921-943.

McKelvey, B. (2004). Toward a 0th law of thermodynamics: Order creation complexity dynamics from physics and biology to bioeconomics. *Journal of Bioeconomics, 6,* 65-96.

McKelvey, B., & Maguire, S. (1999). Complexity and management: Moving from fad to firm foundations. *Emergence, 1*(2), 19-61.

Mitleton-Kelly, E. (2003). Ten principles of complexity and enabling infrastructures. In E. Mitleton-Kelly (Ed.), *Complex systems and evolutionary perspectives of organizations: The application of complexity theory to organizations.* Oxford: Elsevier.

Morgan, K. (1997). The learning region: Institutions, innovation and regional renewal. *Regional Studies, 31,* 491-503.

Nachum, L., & Keeble, D. (2003). MNE linkages and localized clusters: Foreign and indigenous firms in the media cluster of Central London. *Journal of International Management, 9*(2), 171-193.

OSEM. (2004). *Strumenti di analisi e monitoraggio del settore della calzatura sportiva.* Retrieved from http://museo.ath.cx/distretto/osem2004/

Piore, M., & Sabel, C.F. (1984). *The second industrial divide.* New York: Basic Books.

Porter, M. (1998). Clusters and the new economics of competition. *Harvard Business Review, 76,* 77-90.

Pouder, R., & St. John, C.H. (1996). Hot spots and blind spots: Geographical clusters of firms and innovation. *Academy of Management Review, 21*(4), 1192-1225.

Powell, W. W. (1987). Hybrid organizational arrangements: New form or transitional development. *California Management Review, 19*(4), 67-87.

Prusak, L. (1997). *Knowledge in organizations.* Washington, DC: Butterworth-Heinemann.

Rivkin, J. W., & Siggelkow, N. J. (2002). Organizational sticking points on NK landscape. *Complexity, 7*(5), 31-43.

Rullani, E. (2002). The industrial cluster as a complex adaptive system. In A. Q. Curzio & M. Fortis (Eds.), *Complexity and industrial clusters.* Heidelberg: Physica-Verlag.

Sanchez, R. (1993). Strategic flexibility, firm organization, and managerial work in dynamic markets. *Advances in Strategic Management, 9,* 251-291.

Simon, H.A. (1996). *The sciences of the artificial.* Cambridge, MA: MIT Press.

Storper, M. (1997). *The regional world: Territorial development in a global economy.* New York: Guilford Press.

Storper, M., & Harrison, B. (1992). Flessibilità, gerarchie e sviluppo regionale: La

ristrutturazione organizzativa dei sistemi produttivi e le nuove forme di governance. In F. Belassi & P. Bianchi (Eds.), *Nuovi modelli d'impresa gerarchie organizzative e imprese rete*. Milano: F. Angeli.

Sull, D.N. (2003). *The co-evolution of technology and industrial clusters: The rise and fall of the Akron tire cluster.* Working Paper, Harvard Business School, USA.

Tallman, S., Jenkins, M., Henry, N., & Pinch, S. (2004). Knowledge, clusters, and competitive advantage. *Academy of Management Review, 29*(2), 258-271.

Teece, D., Pisano, G., & Shuen, A. (1997). Dynamic capabilities and strategic management. *Strategic Management Journal, 18*(7), 509-533.

Zhang, J. (2002). *Growing Silicon Valley on a landscape: An agent-based approach to high-tech industrial clusters.* Working Paper, Public Policy Institute of California, USA.

KEY TERMS

Agent: An object with various attributes that interprets and interacts with its environment through behavioral rules.

Agent-Based Simulation: A collection of autonomous, heterogeneous, intelligent, and interacting software agents, which operate and exist in an environment. These software agents interact with each other in a non-linear manner with little or no central direction. The large-scale effects determined by the locally interacting agents are called emergent properties of the system. The main goal of agent-based simulation is to enrich the understanding of fundamental processes regulating and determining dynamics of complex adaptive systems.

Complex Adaptive System: A collection of agents, interconnections, and flows where aggregate system behavior is determined from the complex, local interactions of agents.

Computational Model: A simplification of a real system that can be analytically understood and/or run as a computer simulation.

Emergence: The behavior that surfaces out of interaction of a group of agents/people whose behavior cannot be predicted on the basis of individual and isolated actions and is not externally imposed.

Geographical Cluster: A geographically defined production system, characterized by a large number of small and medium-sized firms involved at various phases in the production of a homogeneous product family. These firms are highly specialized in a few phases of the production process and integrated through a complex network of inter-organizational relationships.

NK Model: A binary particle model developed by Kauffman and Weinberger to understand how organisms evolve by undertaking adaptive walks similar to *hill climbing* to achieve maximize fitness.

Chapter XXIII
A Simulation of Strategic Bargainings within a Biotechnology Cluster

Alain Berro
Toulouse University, France

Isabelle Leroux
Le Mans University, France

ABSTRACT

This chapter introduces artificial life as a means of exploring strategic relations dynamics between firms and local authorities within a local biotechnology cluster. It argues that artificial life, combined with a conception of bioclusters as complex adaptive systems, offers a significant approach to understanding the co-evolution of strategies and the potential vulnerability of such systems. The simulation model involves firms and local government administrations that negotiate to share a quasi-rent, and which, to this end, use strategies that are to a greater or lesser extent sophisticated or opportunistic. The results show that the firms adjust their bargaining strategies according to their assessment of gains which might be collectively generated. The results also bring to light that the local authorities play a regulatory role against opportunism and that they are the key players in local coordination. Stemming from these simulations, the authors develop promising new avenues of theoretical and empirical research.

INTRODUCTION

Confronted with their tardiness, compared to the United States, the countries of Europe have put in place voluntary biotechnology development policies over the last ten years. As a consequence, geographic clusters linked to healthcare activities, the environment, and seed production have appeared or increased in Europe, for example the Medicon Valley on the border of Denmark and Sweden, and the Evry Génopole in France. The cluster concept is defined by Porter (1998) as a group of geographically close companies and institutions

whose activities are complementary and characterized by a high degree of specialization and technology transfer. The cluster is based on dense networks of inter-firm relations, characterized by cooperative and competitive links. This strong bond produces collective benefits, such as "quasi-rents," owing to the operation of licenses or the effects of the agglomeration (Zucker & Darby, 1997; Dyer & Singh, 1998).

While the literature on biotech clusters is centered particularly on the strong competitiveness of such innovative systems, certain studies relativize these successes by underlining coordination difficulties linked to conflicts about the sharing and redistribution of the collective benefits (Owen-Smith & Powell, 2003). These coordination flaws are linked structurally to features of biotech clusters. They present as opportunistic behaviors favored by cross-sector cooperation-competition or by a dual-market structure (Roijakkers, Hagedoorn, & Van Kranenburg, 2005). Equally, these coordination difficulties arise from differences of interests which divide public and private players.

However, although there are many studies of the links between firms and public laboratories (Audretsch & Stephan, 1996; MacMillan et al., 2000), there are few that raise the issue of coordination difficulties involving firms and local government administrations (Chataway, Tait, & Wield, 2004; Leroux, 2004; Rausser, Simon, & Ameden, 2000). These links are fundamental, however, given the controlled and much-debated nature of activities connected with biotechnologies. Indeed, local government authorities play an important role in local industrial policy, because they have to guarantee the ethical nature of the research undertaken. This results in highly complex negotiation strategies, as firms seek to appropriate the collective benefits by putting pressure on local authorities, while at the same time currying favor with them. Concentrating on this angle, this chapter will focus on an analysis of negotiation strategies linking companies and local public authorities. Which negotiation strategies occur most frequently? How do these strategies develop together over time? Do they contribute to strengthening or altering the cluster's performance? By addressing these questions, this chapter aims to offer a dynamic quantitative analysis, based on an artificial life simulation and enabling a first evaluation of the occurrence of particular coordination mechanisms within biotech clusters.

The theoretic positioning used here is the evolutionary perspective, which is based on the analysis of complex evolving systems. This permits an understanding of the emergence of the combined properties of a system of agents from the interaction of its constituent elements (Arthur, Durlauf, & Lane, 1997). The system is characterized by a great number of interconnected heterogeneous agents who choose their action according to the choices of the other participants, such that a variety of complex dynamics can be observed. The system's dynamic is sensitive to environmental disturbance. Thus it is possible to analyze its instabilities and potential vulnerability.

With this in mind, we decided on an exploratory simulation model with a heuristic aim. The model involves firms and local government administrations that negotiate to share a quasi-rent, and which, to this end, use strategies that are to a greater or lesser extent sophisticated or opportunistic. The simulation results confirm that the negotiation strategies adopted by the players have an impact on cluster performance. The firms adjust their bargaining strategies according to their assessment of gains that might be collectively generated. The results also show that the local authorities play a regulatory role and that they are the key players in local coordination, even if the situation does not necessarily favor them at the outset.

The second part of the chapter is devoted to a review of the literature and to the propositions that underpin the model. The third part shows the model in action aiming to test these propositions. We then presents the results of the simulations, followed by a discussion of the results and of lines of future research.

TOWARDS AN EVOLUTIONARY APPROACH TO STRATEGIC BARGAININGS WITHIN BIOTECH CLUSTERS

The coordination difficulties that might call into question cluster performance are linked to conflicts about the appropriation of resources and collective rents. In this regard, an initial area examined by researchers relates to the negative impacts of the cooperation-competition duality within biotech clusters. While this duality may be a source of competitive spirit (Jorde & Teece, 1989; Gulati, Nohria, & Zaheer, 2000), it can, on the other hand, lead to opportunistic free-riding behaviors, which translate into unequal gaining of resources or of collective rents (Nooteboom, 1999; Stuart & Sorensen, 2003). This phenomenon occurs particularly frequently in biotech clusters because of the cross-sector (health, environment, food-processing) and fragmented nature of their activities, as Argyres and Liebeskind (2002) demonstrate.

A second issue raised by studies is that of coordination difficulties linked to the heterogeneity of the firms involved (Powell, White, Koput, & Owen-Smith, 2005; Saviotti, 1998; Roijakkers et al., 2005). Biotech clusters are organized as a *dual market*, based on partnerships between big, international firms and small and medium enterprises (SMEs). Nevertheless, these partnerships, based on survival strategies, and strategies to achieve both pecuniary and information rents, lead to imbalances of power and to opportunistic behaviors. This results in instability, which is sometimes *chronic* in the links between small and large firms, which can cast doubt over the performance of biotech clusters.

A third subject raised and demonstrated by research works is that of coordination flaws linked to the highly regulated and controversial nature of innovative biotechnology activities, both public and private. First of all, a number of studies, such as those by Lawson (2004) or Sherry and Teece (2004), emphasize the issue of conflicts about property rights in relation to resources and incomes. Other studies, focused on network dynamics, discuss the notion of "partial embeddedness" between the public and private spheres. Owen-Smith and Powell (2003) show that research laboratories need to establish a strategic balance between academic and industrial priorities in order to avoid "the danger of capture by industrial interests" (p. 1695). Finally, some works concentrate on the role of national and local institutions (Dasgupta & David, 1994; Teece, 1996; Etzkowitz & Leydesdorff, 2000; Lehrer & Asakawa, 2004). Some of these authors are particularly interested in the controversial nature of the biotechnologies (the fight against genetically modified organisms¾GMOs), and in public and private management problems from the viewpoint of the ethical and regulatory concerns (Chataway et al., 2004). While the role of public players in systemic risk reduction is mentioned (Dohse, 1999; Peters & Hood, 2000), it nevertheless forms the basis of no research program.

This literature review shows that few works are devoted to the essential links between companies and local public authorities. As Chataway et al. (2004) and Rausser et al. (2000) say, strategic relations between firms and the public administrative bodies are crucial. Leroux (2004) shows that firms develop bar-

gaining strategies with the aim of capturing quasi-rents and influencing the decisions of the local public authorities, who are guarantors of the general, public interest. The evolution trajectories of the clusters depend on this in a technological and partnership context which is constantly changing. In this study, we develop the first exploratory research to clarify these strategic links, with the objective of better understanding their nature and of measuring their impact on the performance of the cluster. We will consider more precisely the bargaining strategies aimed at gaining the collective benefits. What kind of bargaining strategies do the firms and the local authorities take part in? How do the players adapt their strategies according to environmental change? Do these bargaining strategies play a part in the medium and long-term performance of the cluster?

The theoretic position chosen in order to answer these questions is evolutionary theory, based on the complex adaptive systems paradigm (Arthur et al., 1997; Kirman, 1997). Evolutionary theory, linked to cognitive economics (Simon, 1955; Walliser, 2000), is based on the following principles that challenge the fiction of the representative agent and limited rationality: (1) *heterogeneity* of agents; (2) *variability* that corresponds to the endogenous capacity of the system to produce new directions depending on behavioral mutations of the agents involved; (3) *path dependency* that results from learning effects and auto-reinforcement mechanisms leading to the irreversibility of the cluster's evolutionary dynamics; (4) *inductive learning,* according to which agents are individually involved in a cognitive process of problem-solving—they learn and adapt themselves with experience in a complex evolving system; and (5) *situated rationality* inspired by Simon's (1955) work and taken up by Walliser (2000), concerning a rationality that is constructed through interaction and that involves rationally adaptive agents.

From this evolutionary perspective, the biotech cluster can be understood as a complex evolving system (Janszen & Degenaars, 1998). The significance of this approach is that it takes into account internal mechanisms of decision making and adaptation, both in their development and in their reversal. This is important because, built on a wide variety of partnerships and strategies linking private and public players, the cluster's evolution trajectory can prove to be unstable and even chaotic in certain cases (Luukkonen, 2005; Mangematin et al., 2003; Stuart & Sorensen, 2003). Now, following a series of research questions about evolution trajectories (Mangematin et al., 2003) and the strategic importance of coordination (Chataway et al., 2004; Rausser et al., 2000; Etzkowitz & Leydersdorff, 2000), this approach allows a dynamic analysis of the different bargaining strategies used depending on environmental uncertainty. As stylized facts show (Leroux, 2004), local authorities tend to make concessions when faced with relocation or closure threats, by opportunistic firms which are trying to appropriate the quasi-rents generated by the cluster. The question then is whether the firms' strategies depend on the certain or uncertain nature of the future collective rent. Three propositions can be put forward in order to look for answers to this question. The first seeks to test agents' behavior when the quasi-rents are known. If we distinguish firms' motivations, which satisfy their private interests, and the local authorities' motivations, which satisfy the general interest, then the aim is to grasp the nature of the evolution of the sharing mechanisms when there is no uncertainty about gains.

Proposition 1: When the quasi-rents are known and stable over time, firms tend to develop opportunistic negotiation strategies in order to appropriate them. Firms benefit from concessions granted by local authorities

in response to the firms' threats of possible disengagement.

The second proposition aims to test the development of agents' sharing strategies when there is an uncertainty on the amount of the quasi-rents, and they have to estimate it. While stylized facts show bargaining strategies to be prudently less opportunistic in an uncertain situation (Leroux, 2004), it is necessary to find the determining factors in agents' sensitivity to environmental disturbance.

Proposition 2: When there is uncertainty over the value of quasi-rents, firms develop less opportunistic behaviors, while still benefiting from the concessions made by the local authorities.

The aim of the third proposition is to test the behavior of the local authorities when the amount of the quasi-rents is uncertain and firms' opportunism can call the performance of the cluster into question. A further aspect is to discover if authorities have the capacity to reduce systemic risk.

Proposition 3: When the value of the quasi-rents is uncertain and opportunistic behavior could contribute to a reduction in cluster performance, the local authorities overcome the harmful effects of the firms' strategies.

In order to test these three propositions, we develop an artificial life simulation based on a genetic algorithm involving mutation and crossover operators (Holland, 1975; Goldberg, 1989). The significance of a simulation lies in the endogenous capacity of agents involved to search systematically for new behavioral rules and to include them within their own "world model" according to an adaptative process (Marney & Tarber, 2000; Vriend, 2000). The research methodology associated with this tool is a "theoretic exploratory" approach (Snow & Thomas, 1994). The objective is to explore and develop theoretic teachings by testing research propositions, in order to open the way to new research questions. The simulation consists then in exploring a metaphoric world which generates artifacts for analysis in a heuristic perspective.

THE MODEL: INTERACTION WITHIN BIOTECH CLUSTERS AND THE *STATE OF THE WORLD*

In accordance with the three given propositions, the model consists of three simulations of processes of bargaining involving firms and local government administrations. They bargain to share a collective benefit, affiliated to a quasi-rent, represented in the model by a pie. This is a strategic game under ultimatum (Ellingsen, 1997). When the two transactors involved both want to appropriate an over-large part of the pie using opportunistic means, the negotiation fails. So two kinds of transactors take part in the model as a *state of the world*.

Firms are modeled as *obstinate agents* (Obs) whose demands are independent of those of the adversaries. As they participate in the cluster's performance, they want to appropriate the part of the pie that they fixed themselves depending on their profitability objectives. Some of them expect a large part (more than 50 %) whereas others expect a less significant part (less than 50 %). The part expected also depends on the more or less powerful and opportunistic behaviors adopted by these firms.

Local authorities are modeled as "sophisticated agents" (Soph) which adapt their demand to that hoped for by their adversaries rather than gain nothing. As they answer for the "general interest," they adapt themselves to the

firms' expectations. The stake here is to fix firms within the cluster, to avoid relocations, to stimulate research-innovation links and territorial performance. So they are under firms' ultimatum because the latter sometimes make relocation or employment threats to gain advantages. However, when two local authorities bargain together, they share the pie in a 50/50 proportion with respect to the "general interest" and to their common stake—local development and performance.

DEMAND DETERMINATION

Firms' Demands

The obstinate firm's demand d_i is broken down into two components, the size of the pie expected and the portion demanded. Thus:

d_i = expected size of the pie (teg) * demanded portion (i)

with

T: the real size of the pie

$teg \in [0,TG]$, minimum value and maximum value of teg

$i \in I \subset [0,1]$, I set of portions demanded

of which

$d_i \in D \subset [0,TG]$, D finite set of possible demands.

The strategy d_i with $i = 0.5$ is called a *fair strategy*. Any strategy for which $i > 0.5$ is called a *greedy strategy*. And the other strategies for which $i < 0.5$ are called modest strategies (Ellingsen, 1997).

Local Government Demands

Local authorities, whose strategies we called r, are supposed to identify the adversary's strategy and adapt their demand to that expected by the adversary. Consequently, when an authority bargains with a firm whose demand is d_i, it demands:

$$r = teg_r - d_i$$

Nevertheless, local authorities can also risk a failure situation if they overestimate the size of the pie. So the set of possible strategies is $S = D \cup \{r\}$, with d_i the obstinate demand and r the sophisticated demand.

PAY-OFF FUNCTION

If firm i asks for d_i and firm j asks for d_j, then firm i receives the following pay-off:

$$\Pi_{ij} = \begin{cases} d_i \text{ if } d_i + d_j \leq T \\ 0 \text{ if not} \end{cases}$$

If the total of d_i and d_j is greater than the real size of the pie T, then the bargaining has failed and neither firm obtains any gain. The surpluses are not redistributed and are considered as lost.

A local authority that negotiates with a firm thus obtains:

$$\Pi_{ri} = teg_r - d_i \text{ if } teg_r \leq T \text{ et } d_i \leq T$$

And when two authorities meet, they obtain:

$$\Pi_{r1r2} = \frac{teg_{r1}}{2} \text{ if } \frac{teg_{r1} + teg_{r2}}{2} \leq T$$

Table 1. The pay-off matrix

	Obstinate d_i	Sophisticated r1
Obstinate d_j	d_i d_j	teg_{r1}-d_j d_j
Sophisticated r2	d_i teg_{r2}-d_i	$teg_{r1}/2$ $teg_{r2}/2$

So the pay-off matrix presented in Table 1is obtained.

SIMULATIONS

Implementation of the Genetic Algorithm

Each agent is determined by its *genotype*, broken down into two components: its strategy and the expected size of the pie. The obstinate population (firms) is divided into seven profiles which correspond to seven discrete intervals between 0 and 100.[1] Each profile has been arbitrarily fixed and corresponds to the portion demanded.[2]

The simulations[3] are based on the following parameters: the initial size of the pie is 1; the pie can vary according to the interval [0.1 : 2.0]; the number of agents within the population is 1,000; the mutation rate is 10%; the crossover rate is 50%; the initial distribution of the different populations involved at the start of the game is 12.5%. The choice of these parameters is the result of a compromise between keeping a selective pressure on the population in order to ensure algorithm convergence, and maintaining genetic diversity in the population to avoid a too rapid convergence (Bäck,1994; Schoenauer & Sebag, 1996).[4] In a cluster, agents do not systematically bargain with the whole population, but only with some agents when necessary. Consequently, this constraint has been introduced into the model. At each bargaining an agent bargains with a representative sample of 10% of the whole population. Each agent is next assessed according to the gains he can generate. Each simulation was carried out 1,000 times.

Table 2. The seven obstinate profiles

Obs 7	Firms whose demand is 7%	Modest
Obs 21	Firms whose demand is 21%	
Obs 35	Firms whose demand is 35%	
Obs 50	Firms whose demand is 50%	Fair
Obs 64	Firms whose demand is 64%	Greedy
Obs 78	Firms whose demand is 78%	
Obs 92	Firms whose demand is 92%	

Relationship Proximity

The model introduces a *relationship proximity* linking some agents within this artificial world. Although firms and local authorities can bargain with every agent within the cluster (*notation step*), they exchange information on the pie size only with partners that they have noticed adopting the same strategy as them during the bargaining phase (*crossover step*). Consequently, if certain agents are not able to recognize at the outset the strategy of their adversary, they are nevertheless in a position to know it at the end of the bargaining process. So an internally generated, close relationship occurs which links agents in a common, mutual, self-protection strategy. This strategy enables the selective exchange of information on the pie size between agents that have the same strategic approach.

Simulation S1: The Size of the Pie Is Known and Does Not Change

In this first simulation, we test proposition 1. The size of the pie is T = 1 and does not

change during the bargaining. Using a simulation enables step-by-step observation of the strategies adopted by the agents involved and of the changes of direction possible over 500 periods.

Simulation S2: The Size of the Pie Is Unknown

In this second simulation, we test proposition 2. Uncertainty is introduced into the bargaining game. The pie size is not known and the agents have to try to estimate it. The size of the pie is fixed at T = 1. Here, firms and local authorities are endowed with an endogenous capacity to modify their respective demands d. More precisely, a learning process in respect of teg, based on the use of evolutionary operators such as crossover and mutation, enables them to estimate the size of the pie. Each agent has the capacity to assess the expected size of the pie (teg), and each new assessment leads to a changed demand d. The possibility of failure is higher because agents can tend to overestimate or underestimate the size of the pie during the evaluation process.

Simulation S3: The Size of the Pie Varies Depending on the Behaviors of the Agents Involved

In this third simulation, we test proposition 3. The pie size becomes variable and represents the performance of the cluster. The more firms and local authorities choose opportunistic strategies leading to the failure of negotiations, the greater the negative impact on the global performance of the cluster and thus on its viability over time (the size of the pie decreases). On the other hand, the more the agents choose strategies supporting the success of negotiations (as concessions, or fair vs. modest strategies), the more positive the impact on the global performance of the cluster (the size of the pie increases). So we need to observe how the different strategies develop under these conditions and to see whether the local public authorities manage to get around the firms' opportunism.

Technically, a parameter of influence k here affects the real size T of the pie. If at the preceding step (n-1) the number of successful bargainings is higher than the number of failures, then the size of the pie increases by 0.01. In the opposite case, it decreases by 0.01. The choice of this parameter k = 0.01 is arbitrary and fixed at a low level. Its purpose is to illustrate that bargaining failure influences the cluster performance but in a non-radical way; it will not induce a major economic crisis or bring about the closure of a company which is a major source of orders. In these last two cases, the performance of the cluster can be radically disrupted, which is not our case here.

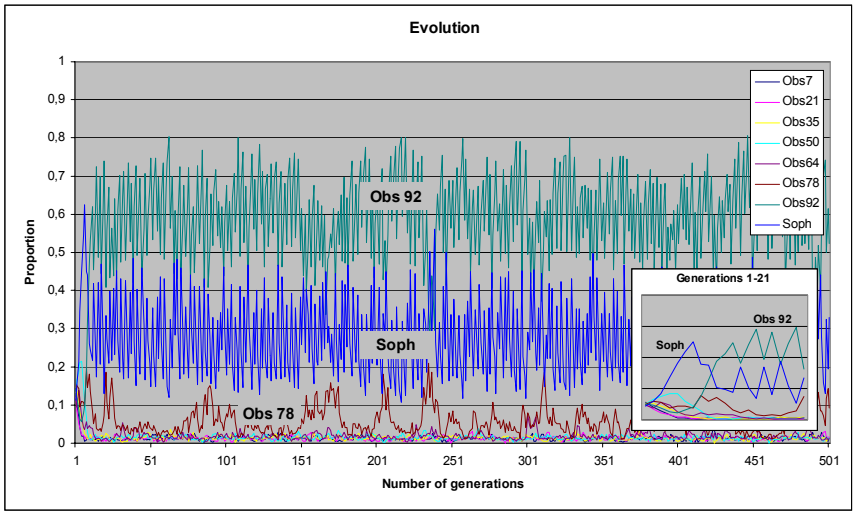
RESULTS

Simulation S1

The simulations show that bargaining behaviors evolve in two distinct phases. First, local authorities making concessions are more important than the other transactors during the first 20 generations. During these periods, bargaining leads mainly to an equal share of the pie (50/50). Second, this superior tier of local authorities which make concessions then contributes to the emergence of the greediest firms, which demand 92% of the pie, and a smaller proportion of firms which demand 78% of the pie.

So it is important to note that in the medium and long terms, bargaining is stabilized around a majority of greedy firms whose existence is maintained by the presence of local authorities making concessions. Without uncertainty on the pie size, the greediest firms take advantage of the situation. We can further assert that the local

Figure 1. Simulation S1[5]



authorities play a distributive role, albeit neither directly nor deliberately, since by making concessions they contribute to enabling the greediest firms to gain quasi-rents to the detriment of the modest and fair firms. Without local authorities making concessions then, the greediest firms could not take advantage of the situation.

This simulation validates proposition 1. When quasi-rents are known and stable over time, firms' behavior is very opportunistic, and they make use of concession-giving local authorities.

Simulation S2

After 1,000 tests, the results show a variety of possible outcomes:

- In 46.20% of cases, the bargaining is stabilized around the greediest firms, whose demand is for 92%, and the local authorities. The existence of these firms in the game is maintained by the presence of these public authorities, as in simulation S1.
- In 29.6% of cases, the bargaining is stabilized around firms demanding either 78% or 64%, and the local institutions.
- In 22.4% of cases, the bargaining is stabilized around the fair firms, whose demand is 50% of the pie, and the local institutions.
- In 1% of cases, the bargaining is stabilized around the modest firms which demand less than 35% of the pie, and the fair firms. In these very rare cases, local public authorities are missing.
- The last 0.8% consists of errors or accidents in the evolutionary process which sometimes occur.

So when the pie size is unknown, results can be very different and depend on the capacity of the agents involved to find and appropriate the right size of pie quickly. The "winners" are those who succeed in correctly estimating the size of the pie as soon as possible, and who exchange information with the most successful agents. In 46.20% of cases, the greediest firms rapidly benefit from the very large presence of

Figure 2. Average results on 1,000 generations

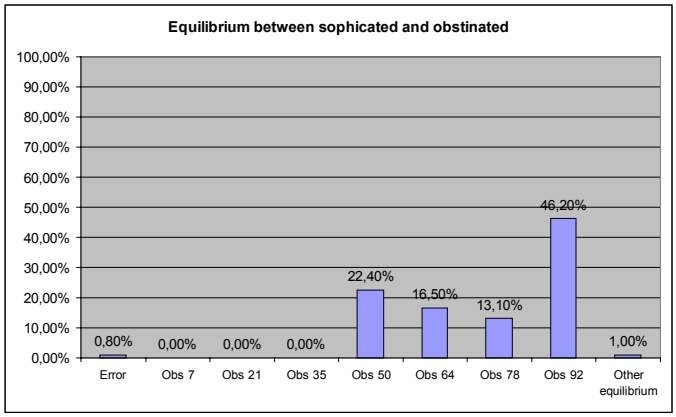
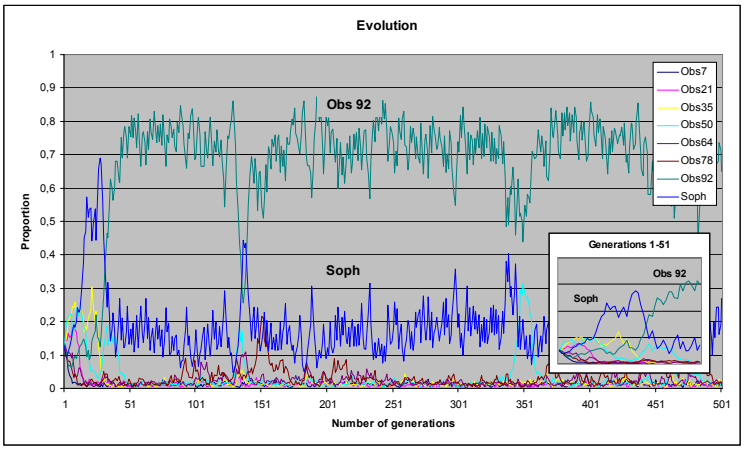
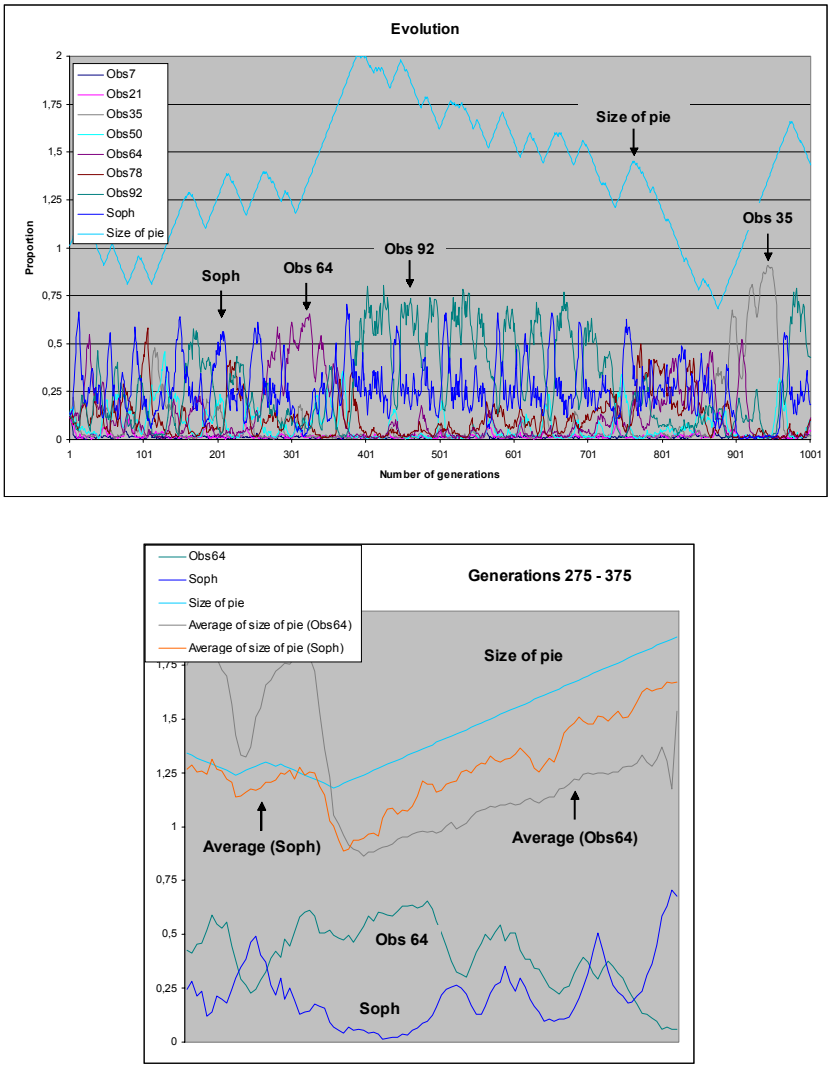


Figure 3. Simulation S2—average results on 1.000 generations



local authorities during the first 40 generations. The authorities have such a significant presence because of the pie size researching process. Concessions facilitate correct evaluations; the low demands of modest and fair firms contribute as well to this and help to avoid failure. In other cases, prudent and fair strategies are prioritized, such as the downward revision of demands. Concessions are plentiful at the start of the period and then give way to more prudent, obstinate strategies once the size of the pie is approximately estimated. So when the pie size is unknown, the distributive role of local authorities is very high. In only 1% of cases can modest firms survive without the local public administrations because they themselves play a distributive part. This simulation validates proposition 2. Firms develop generally less opportunistic strategies, but continue to draw benefit from concession-giving local authorities.

Figure 4. Simulation S3 and zoom



Simulation S3

The results show that agents adapt their behavior in order to preserve the pie size. As the bargaining game has no stability, 1,000 periods are represented here. They show primarily that firms tend to exploit the bargaining process according to their evaluation of the pie size, its evolution, and the more or less significant presence of local public administrations and modest/fair firms. Thus, once the pie has reached a size near the maximum expected threshold, the move from prudent/fair/modest strategy to the greediest strategy can be observed. As an example, the periods [275 : 375] are characterized by a strong presence of the prudent firms whose demand is 64%. During this period, prudent behaviors contribute significantly to making the pie grow towards the maximum threshold. Once the pie has reached this size, we can observe the move from the prudent

64% strategy to the greediest 92% strategy. This is possible because of the presence of the local authorities which reduces conflict between the greediest agents and thus averts any radical reduction of the collective performance. So when the cluster is threatened by too much greedy behavior, which can considerably alter its global performance, local authorities appear and play a regulatory role, allowing the pie to grow. In this way, cluster performance is maintained principally because of the role of local authorities. This simulation then validates proposition 3. When there is uncertainty about the quasi-rents and opportunism threatens cluster performance, the local authorities overcome the harmful effects of opportunistic behaviors.

DISCUSSION AND CONCLUSION

The results of these simulations validate our propositions and open the way to new research questions. The evolutionary approach allows a first reflection on dynamics and phases of strategic behaviors within a biotech cluster. The simulations show that agents modulate their behavior through time according to various parameters: the profits withdrawn during bargaining, the effects of their own behavior on the global performance of the cluster, and the uncertainty and their capacity to observe their partners' strategies and to make concessions. Supported by the assumption of *situated rationality*, the cluster can be viewed as an "emergent processual regularity" that is highly individual and differentiated owing to the different combinations of strategies. This model can then contribute to a deeper analysis focused on clusters' intrinsic characteristics (Waluszewski, 2004; Carbonara, 2004; Peters & Hood, 2000).

These simulations also call into question the assumption that *rivalry-cooperation* systematically produces emulation, and raise questions about the vulnerability of such local systems. As the idea of cluster vulnerability through collusion and lock-in effects is developed in the literature (Floysand & Jakobsen, 2002; Peters & Hood, 2000), so this chapter offers an analysis centered on strategic opportunistic behaviors. Artificial life simulations enable us to observe step by step how the agents *instrumentalist* their relationships and modify their strategies in a complex environment so as to appropriate benefits or to preserve collective performance. At any moment, an "accident" of system evolution, viewed as an artifact, can considerably affect the composition of the cluster. This suggests that further analysis of clusters' survival and perennially through the strategic approach is a promising avenue for future research.

As we have argued, only a few studies develop the part played by local government in biotech clusters (Dohse, 1999). The evolutionary perspective can contribute to correcting these inconsistencies and show the importance of the regulating role of local administrations, which have a power that can be said to be the *power of the weak* as developed by Schelling (1960). Without these local authorities, the cluster's performance could not be maintained. The "power of the weak" follows from the fact that they are the key players in the groupings, even if the situation (disengagement threats and concessions) is not favorable to them at the beginning. Further, this model raises one of the main ambiguities of public-private coordinations that can occur within biotech clusters (Leroux, 2004). On the one hand, they are supported by local government because of the uncertainty which can be caused by firms' behaviors (relocation and closure threats) and by environmental evolution. On the other hand, supporting firms can in some cases contribute to the emergence of the greediest strategies and to the ousting of the less opportunist firms. A fruitful direction for further empirical analy-

sis may be a deeper probing of the question of clusters' strategic local governance in relation to the networks' strategic analysis (Powell et al., 2003; Gulati et al., 2000).

From an exploratory point of view, such a model with a heuristic aim opens new avenues of theoretic and empirical research. From a theoretical point of view, it can lead to a reflection focused on conflict and power analysis within clusters, according to an evolutionary approach centered on the analysis of emerging rules. As part of the research carried out in the institutionalist framework (Hodgson, 1998), the analysis of micro-regularities emerging from interaction certainly brings to light the arbitral dimension of bargaining rules. They are both a constraint for collective action and the outcome of this collective action. Thus the theoretic question is about the emergence and the evolution of these rules under uncertainty, and about the links that can be established with the cluster's performance and survival. From an empirical point of view, an interesting avenue to develop would be an analysis focused on conflicts within biotech clusters, such as the nature of conflict, the different resolution processes adopted by the local actors involved and dependent on environmental constraints (market, legal regulation), and finally their impact on the evolution system.

However, while these heuristic simulations lead towards new areas of research, their limitations, too, point to other directions for future research. The first limitation of the model is that it proposes and develops only two kinds of agents, firms on one hand and local government authorities on the other hand. So the scope needs to be enlarged, taking into account a greater diversity of agents (research laboratories, development agencies) and a greater diversity of exchanges (sellers-suppliers). Second, this model is limited to relationships developed within the cluster, so it is important to take into account a more complex environment including the embeddedness of the actors involved in complex social relations outside the cluster (relations with client companies, relations with shareholders, and European policies) such as is developed within some earlier empirical studies (Peters & Hood, 2000; Floysand & Jakobsen, 2002). The questioning here is focused on the various levels of decision and their impact on the cluster performance and survival. The third limitation of the model is that it introduces proximity as a relational and communicative distance, but not as a geographic one. With this in mind, we now intend to introduce geographical distance such as in the studies of Brenner (2001) and Zimmermann (2002). Computing tools such as multi-agent systems can also contribute to the reinforcement of the mechanisms of inductive reasoning, while introducing geographic proximity parameters. Future research may examine lock-in effects and the intrinsic vulnerability of biotech clusters by reducing these constraints.

REFERENCES

Argyres, N., & Liebeskind, J. (2002). Governance inseparability and the evolution of U.S. biotechnology industry. *Journal of Economic Behavior and Organization, 47,* 197-219.

Arthur, B. (1994). *Increasing returns and path dependence in the economy.* Ann Arbor: University of Michigan Press.

Arthur, W. B., Durlauf, S. N., & Lane, D. (1997). *The economy as an evolving complex system II.* Reading, MA: Addison-Wesley (SFI Studies in the Sciences of Complexity XXVII).

Brenner, T. (2001). Simulating the evolution of localized industrial clusters. An identification of the basic mechanisms. *Journal of*

Artificial Societies and Social Simulation, 4(3).

Carbonara, N. (2004). Innovation processes within geographical clusters: A cognitive approach. *Technovation, 24,* 17-28.

Chataway, J., Tait, J., & Wield, D. (2004). Understanding company R&D strategies in agro-biotechnology: Trajectories and blind spots. *Research Policy, 33,* 1041-1057.

Dasgupta, P., & David, P. (1994). Toward a new economics of science. *Research Policy, 23,* 487-521.

Dyer, J., & Singh, H. (1998). The relational view: Cooperative strategy and sources of interorganizational competitive advantage. *Academy of Management Review, 23,* 660-679.

Dohse, D. (2000). Technology policy and the regions. The case of the BioRegio contest. *Research Policy, 29,* 1111-1113.

Ellingsen, T. (1997). The evolution of bargaining behavior. *Quarterly Journal of Economics, 112*(2), 581-602.

Etzkowitz, H., & Leydesdorff, L. (2000). The dynamics of innovation: From national systems and mode 2 to a triple helix of university-industrie-government relations. *Research Policy, 29,* 109-123.

Floysand, A., & Jakobsen, S. E. (2002). Clusters, social fields and capabilities. *International Studies of Management and Organization, 31*(4), 35-55.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Reading, MA: Addison-Wesley.

Gulati, R., Nohria, N., & Zaheer, A. (2000). Strategic networks. *Strategic Management Journal, 21*(3), 203-215.

Hodgson, G. M. (1998). The approach of institutional economics. *Journal of Economic Literature, XXXVI*(March), 166-192.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: An introduction with application to biology, control and artificial intelligence.* Ann Arbor: University of Michigan Press.

Janszen, F., & Degenaar, G. (1998). A dynamic analysis of the relations between the structure and the process of national systems of innovation using computer simulation; the case of Dutch biotechnological sector. *Research Policy, 27,* 37-54.

Kirman, A. P. (1997). The economy as an interactive system. In W. B. Arthur, S. N. Durlauf, & D. Lane (Eds.), *The economy as an evolving complex system II* (pp. 491-531). Reading, MA: Addison-Wesley (SFI Studies in the Sciences of Complexity XXVII).

Lawson, C. (2004). Patenting genetic materials' unresolved issues and promoting competition in biotechnology. *Information Economics and Policy, 16,* 92-112.

Lehrer, M., & Asakawa, K. (2004). Rethinking the public sector: Idiosyncrasies of biotechnology commercialization. *Research Policy, 33,* 921-938.

Leroux, I. (2004). Les ambivalences des coordinations locales entre négociation, conflits et enjeux de pouvoir. Le cas des partenariats constitutifs d'une génopole à Toulouse. *Revue d'Economie Régionale et Urbaine, 4,* 513-538.

Luukkonen, T. (2005). Variability in organizational forms of biotechnology firms. *Research Policy, 34,* 555-570.

Mangematin, V., Lemarié, S., Boissin, J.P., Catherine, D., Corolleur, F., Coronini, R., &

Trommetter, M. (2003). Development of SMEs and heterogeneity of trajectories: The case of biotechnology in France. *Research Policy, 32,* 621-638.

McMillan, G., Narin, F., & Deeds, D. (2000). An analysis of the critical role of public science in innovation: The case of biotechnology. *Research Policy, 29,* 1-8.

Marney, J. P., & Tarber, F. E. (2000). Why do simulation? Towards a working epistemology for practitioners of the dark arts. *Journal of Artificial Societies and Social Simulations, 3*(4).

Nooteboom, B. (1999). The dynamic efficiency of networks. In A. Grandori (Ed.), *Interfirm networks: Organization and industrial competitiveness* (pp. 91-119). London: Routledge.

Owen-Smith, J., & Powell, W. (2003). The expanding role of university patenting in the life sciences: Assessing the importance of experience and connectivity. *Research Policy, 32,* 1695-1711.

Peters, E., & Hood, N. (2000). Implementing the cluster approach. *International Studies of Management and Organization, 30*(2), 68-92.

Porter, M. (1998). Clusters and the new economics of competition. *Harvard Business Review, 76,* 77-90.

Powell, W., White, D., Koput, K., & Owen-Smith, J. (2005). Network dynamics and field evolution: The growth of interorganizational collaboration in the life sciences. *American Journal of Sociology, 110*(4), 1132-1205.

Rausser, G., Simon, L., & Ameden, H. (2000). Public-private alliances in biotechnology. Can they narrow the knowledge gaps between rich and poor? *Food Policy, 25,* 499-513.

Roijakkers, N., Hagedoorn, J., & Van Kranenburg, H. (2005). Dual market structures and the likelihood of repeated ties—Evidence from pharmaceutical biotechnology. *Research Policy, 34,* 235-245.

Saviotti, P. (1998). Industrial structure and the dynamics of knowledge generation in biotechnology. In J. Senker (Ed.), *Biotechnology and competitive advantage* (pp. 9-43). Cheltenham, UK: Edward Elgar.

Schelling, T. (1960). *The strategy of conflict.* Boston: Harvard University Press.

Sherry, E., & Teece, D. (2004). Royalties, evolving patent rights, and the value of innovation. *Research Policy, 33,* 179-191.

Simon, H. (1955). A behavioral model of rational choice. *Quarterly Journal of Economics, 69,* 99-118.

Snow, C., & Thomas, J. B. (1994). Field research methods in strategic management. *Journal of Management Studies, 31,* 457-479.

Stuart, T., & Sorenson, O. (2003). The geography of opportunity: Spatial heterogeneity in founding rates and the performance of biotechnology firms. *Research Policy, 32,* 229-253.

Vriend, N. J. (2000). An illustration of the essential difference between individual and social learning and its consequences for computational analyses. *Journal of Economic Dynamics and Control, 24,* 1-19.

Walliser, B. (2000). *L'économie cognitive.* Paris: Odile Jacob.

Waluszewski, A. (2004). A competing or cooperating cluster or seven decades of combinatory resources? What's behind a prospecting biotech valley? *Scandinavian Journal of Management, 20,* 125-150.

Zimmermann, J. B. (2002). Des clusters aux small-worlds: Une approche en termes de proximités. *Géographie, Economie, Société, 4,* 3-27.

Zucker, L. G., & Darby, M. R. (1997). Present at the biotechnology revolution: Transformation of technological identity for a large incumbent pharmaceutical firm. *Research Policy, 26,* 429-446.

KEY TERMS

Clusters: According to the Porter (1998) definition, clusters are geographic concentrations of interconnected companies, specialized suppliers, service providers, firms in related industries, and associated institutions (universities, standards agencies, trade associations) in a particular field that compete but also cooperate. Clusters can be specialized in biotechnology, computing, wine, electronics, and so forth. One of the best known is the Silicon Valley.

Cooperation-Competition: The duality between cooperation and competition is very high within clusters because firms are linked locally both by complementary relations and vigorous competition. This process can contribute to innovation and collective performance. In this case, cooperation and competition can coexist because they are on different dimensions.

Inductive Learning: The agents are individually involved in a cognitive process of problem solving. They learn step by step by experience and they adapt their behaviors according to the different situations previously observed.

Path Dependency: Path dependency is the dependence of the outcome of a dynamic process on historical events. So learning process and auto-reinforcement mechanisms lead to the irreversibility of the cluster dynamics.

Relational Quasi-Rent: The relational quasi-rent is a surplus generated by a cooperation process involving several actors linked by complementarities (firms, institutions). This surplus cannot be appropriated by one actor to the detriment of the others because of its collective nature. So the sharing of the quasi-rent between the participants can be a big problem because no objective criteria exist for sharing it. Thus, opportunist behaviors can often occur such as "hold up" strategies.

Situated Rationality: The situated rationality of an agent is defined as a *limited rationality* fundamentally linked to the context of interaction. The problem-solving process of each agent takes into account the links with his environment, such as for example the proximity links built with some of the agents involved.

ENDNOTES

1 The discrete intervals allow precise statistical evaluation of the results.

2 The choice of seven profiles is sufficiently high to be representative of the principal large categories of possible demands, while guaranteeing legible results on a histogram.

3 Each simulation is performed according to the following steps: (1) *initialization:* random or deliberate choice of strategies; (2) *notation:* bargaining process and notation—that is, assessment of each agent according to the gains he can generate; (3) *selection:* process through which agents are chosen to be replicated, the most favored being those with the highest level of notation; (4) *crossover:* crossover and reproduction of the most successful agents; (5) *mutation:* random deterioration of one or several genetic characters of an agent; (6) return to 1.

[4] Simulations have been tested with mutation rates varying from 0% to 15%. When the mutation rate is less than 5%, the percentage of algorithm errors is more than 10% and distorts the analysis of the results. At 15%, the mutation rate becomes destructive and does not match the propositions' realism, favoring the greediest profiles.

[5] The abscissa represents the successive generations or periods of bargaining. The Y-axis represents the proportion of each population within the total population.

Chapter XXIV
Knowledge Accumulation in Hayekian Market Process Theory

Nicole J. Saam
Ludwig-Maximilians-Universität, Germany

Wolfgang Kerber
Philipps-Universität Marburg, Germany

ABSTRACT

This simulation model is an example of theory-driven modeling that aims at developing new hypotheses on mechanisms that work in markets. The central aim is to model processes of knowledge accumulation in markets on the theoretical basis of Hayek's concept of "competition as a discovery procedure," in which firms experiment with innovations that are tested in the market, and the superior innovations are imitated by other firms through mutual learning. After an overview on the structure of these simulation models and important results of previous research, we focus on the analysis of the severe negative effects that limited imitability has for this Hayekian process of knowledge accumulation. We show that limited imitability can hamper this process through the emergence of a certain kinds of lock-in situations which reduces the number of changes in the position of the leading firm.

INTRODUCTION

Economic growth theory as well as evolutionary theory of economic change has shown that technological progress generated by processes of innovation is the most important determinant of economic growth (Nelson & Winter, 1982; Aghion & Howitt, 1998). The search for factors that foster or impede the generation and spreading of innovations is a central theme in modern innovation economics. Since innovations (as new products or production technologies) emerge primarily in competition processes between firms in markets, the analysis of the dynamics and the interaction between competing firms is crucial for explaining technological progress.

The simulation models that are presented in this chapter are based upon evolutionary approaches to market competition, particularly on

Hayek's concept of "competition as a discovery procedure" (Hayek, 1948, 1978, p. 179). In this approach competition is seen as a process of parallel experimentation, in which rivalrous firms on the supply side of a market generate and test different hypotheses about the best way to fulfill the consumers' preferences. From an Hayekian perspective, which particularly emphasizes knowledge problems, the ultimate test of which competitors have superior products or services in regard to preferences of consumers is the market itself (via its feedback through profits or losses). And the firms with the relatively inferior products and services can learn from the leading firms by imitation. This evolutionary process of variation and selection of hypotheses leads to a path of knowledge accumulation in the market, driven by permanent innovation and mutual learning between the firms (Kerber, 1997).

In our simulation experiments we analyze central mechanisms and interactions between the firms, particularly the extent of mutual learning, in this Hayekian knowledge-generating competition process. One of the important results in our previous research was that the growth rate of knowledge accumulation in competition depends critically on the extent of the imitability of the activities of the firms by their competitors, because imitation is necessary for mutual learning between the firms. The case of limited imitability is empirically very relevant, because imitation of successful firms can be a time-consuming, complex, and risky activity, which also can fail (Dosi, 1988). *In this chapter we analyze more deeply the causes for the severe reduction of knowledge accumulation through the non-imitability of particular activities.* Since such non-imitabilities can lead to some kinds of lock-in situations with regard to the change of the leading firms, we test three different hypotheses about the correlations between the number of emerging lock-in situations, the changes of the leading firms, and the growth rate of knowledge accumulation. The simulation model is approximated by a new meta-model, multivariate regression analysis, to verify these hypotheses. We find that the non-imitability of one activity does *not only* reduce the extent of mutual learning, because *this activity* cannot be imitated. Rather the non-imitability of activities generates phenomena that hamper learning additionally: (1) lock-in situations hamper mutual learning on the side of the *competitors,* and (2) the reduction in the number of changes of the leading firm hampers mutual learning on the side of the *leading firm.* In Hayekian market process theory, these are the effects that slow down knowledge accumulation in competition under limited imitability assumptions in a severe way.

THEORETICAL BACKGROUND

Evolutionary Concepts of Competition

The theoretical background of our simulation models are evolutionary approaches to market competition, which are based upon Schumpeter, Hayek, and evolutionary innovation economics. Schumpeterian concepts of competition characterize competition as a rivalrous, dynamic process between firms which consists of innovations by entrepreneurs and their imitation by the competitors (Schumpeter, 1934). In the Hayekian concept of "competition as a discovery procedure," competition is seen as a process of parallel experimentation, in which the firms try out new products and technologies in the market, implying the generation of new knowledge through this market test (Hayek, 1978). In modern evolutionary innovation economics, the accumulation of knowledge, which drives economic development, is modeled—in analogy to biological evolution—as the outcome of variation-selection-processes in re-

gard to products and technologies (Nelson & Winter, 1982). Although these theoretical approaches differ in many respects, they have in common that market competition is seen as a process in which new knowledge (in the form of new products and technologies) is generated and spread, leading to a wealth-enhancing process of knowledge accumulation. In contrast to the model of perfect competition in mainstream economics which ignores innovation processes and focuses instead on the problem of price competition (treating products and technologies as exogenous), these evolutionary approaches to market competition encompass innovation processes and see technological progress as a crucial result of competition.

Our approach to model certain aspects of these knowledge-generating processes in competition can be best understood from the perspective of the Hayekian concept of "competition as a discovery procedure." One of the most important insights from Hayek's research in economics is that the assumption in the model of perfect competition that both the firms on the supply side and the consumers on the demand side have perfect knowledge is fundamentally wrong: "The real problem in all this is not whether we will get given commodities and services at given marginal costs but mainly by what commodities and services the needs of the people can be satisfied most cheaply. The solution of the economic problem of society is in this respect always a voyage of exploration into the unknown, an attempt to discover new ways of doing things better than they have been done before" (Hayek, 1948, p. 100). Due to epistemological reasons, nobody can know beforehand what the best solutions for the problems of consumers are. Only through market competition—conceived as a process of parallel experimentation, in which the competing firms try out different new hypotheses about how these problems can be solved best—can the knowledge be generated of what the superior products and technologies are. Therefore the market test is crucial for identifying the superior innovations, which then can be imitated by others implying the spreading of this new knowledge. That is the reason why Hayek characterizes competition as a discovery procedure.[1]

The evolutionary concept of competition as a test of hypotheses (Kerber, 1997) is based upon this Hayekian approach. It is assumed that on the supply side of the market, there are a number of firms that only have subjective, bounded, and fallible knowledge about the preferences of the consumers and how to fulfill them best. The firms have a set of activities (design and quality of products, marketing, production, organization, etc.), which they can carry out in order to ensure the best and/or most worthwhile performance to the consumers, but due to their ignorance they have only conjectures (hypotheses) about how to do this best. As firms do not know the preferences of the consumers, they will develop different conjectures about the best products, leading to a heterogeneity of problem solutions that are offered in the market. The consumers on the demand side of the market decide which of these different offers of the suppliers are most capable of fulfilling their preferences (or solving their problems). Through their choice they reveal which of the problem solutions of the suppliers are better and therefore which hypotheses of the firms can be seen as superior knowledge. Primarily, it is the choice by the demand side which constitutes the market test and therefore decides on the selection of innovations. In markets the successful firms with increasing market shares usually also have larger profits, whereas the firms which lose market shares have lesser profits or even suffer losses. Therefore large incentives exist: (1) to advance in competition with superior products, and (2) to imitate or improve one's performance, if the market test shows that the own

performance is inferior compared to others. The market test does not only imply an informational feedback about the quality of the hypotheses of the competing firms, but through the ensuing profits and losses also a motivational feedback. Therefore the superior knowledge is not only identified and rewarded, but there are also large incentives for its spreading by imitation.

This process of parallel experimentation with different innovative solutions, which are tested in the market and spread by imitation, can also be modeled as an evolutionary process of variation and selection. The innovative search of firms for better solutions under the assumption of their knowledge problems can be interpreted as a variation of problem solutions. The selection is carried out by the market test (including the ensuing imitation processes). The evolutionary growth theory of Nelson and Winter (1982) is driven by such a variation-selection-process. It leads to a path of endogenously produced technological progress, which through increasing productivity fosters economic growth. Long-term processes of variation and selection can therefore lead to a systematic process of accumulation of knowledge. In the general literature on evolutionary epistemology, this knowledge-generating effect of evolutionary variation-selection-processes has been elaborated very clearly (e.g., Popper, 1972).

What is the main mechanism for the knowledge-generating effect in these evolutionary variation-selection processes and therefore also in our Hayekian concept of competition as a test of hypotheses? Variation processes lead to a number of different problem solutions that are tried out with different success. The market test reveals the relatively best solutions, which can be imitated by the other competitors. Therefore the less successful competitors can learn from the firms with the superior knowledge. Or to put it in evolutionary terms: due to the incentives in the market, the rate of reproduction of superior problem solutions is higher than that of inferior solutions. As a consequence, the average quality of the performance of the firms in the market increases systematically. The knowledge-generating effect depends both on the variation of problem solutions and the possibility of mutual learning between the competitors. In the evolutionary approaches of innovation economics, it has been emphasized that the continuous generation of new variety is crucial for ensuring an ongoing process of knowledge accumulation in market processes (Metcalfe, 1989).

Previous Research

Although the basic idea of Hayek's competition as a process of parallel experimentation, in which new knowledge is generated through a trial-and-error-process, is well known and seems to be broadly accepted on a common sense level by most economists, so far there have been no real attempts to analyze his notion of competition as a discovery procedure in a more rigorous, analytical way. In our research we attempt to model at least some core mechanisms of the Hayekian notion of competition by using simulation models. The basic structure of the simulation model is characterized by the knowledge problems of the suppliers, who have only fallible hypotheses (conjectures) about what products the consumers would appreciate most. Their ex-ante ignorance is modeled by the stochastic result of the quality of their innovation activities, which they have to try out in the market (innovation phase). As a consequence, a variety of different hypotheses are tried out in the market. Only the market feedback, through the buying decisions of the consumers, reveals which of these hypotheses is better at fulfilling the consumers' preferences. In the second part of one period of the simulation, after the firms know whose knowledge is superior, all firms attempt to imitate the perfor-

mance of the best (or the leading) firm (imitation phase). But due to the difficulties of imitation, we assume that the firms can only catch up to a certain extent—that is, that also at the end of that period, a considerable heterogeneity between the firms prevails. After this first period, new sequences of stochastic innovation and imitation follow. This rather simple mechanism of variation (through stochastic innovation) and selection (through the revealing of the superior performance and its partial imitation by the non-leading firms) generates an increasing path of the average performance of all firms in the industry, expressed by an abstract fitness parameter. We define fitness as the knowledge that the firms have on how to fulfill the consumers' preferences.[2] In a dynamic environment, fitness may decrease or increase. In that respect, a process of knowledge accumulation through innovative experimentation and mutual learning can be analyzed by this simulation model.

Using the basic structure of this model, a number of hypotheses about the determinants of the knowledge accumulation process can be tested. In our previous research, we have analyzed particularly four hypotheses (Kerber & Saam, 2001; Saam, 2005):

- *The average growth rate of the knowledge accumulation process increases with the number of independently innovating firms.* One of the most important questions in industrial economics and competition policy is: To what extent does the concentration of firms in a market affect the performance of competition on this market? The rationale of merger policy is that increasing the concentration in markets mergers can lead—either through unilateral or coordinated effects—to market power, higher prices, and allocative inefficiencies, but also to potential efficiency gains (Motta, 2004, pp. 231-265). These arguments, however, refer only to the effects of concentration on price competition. In our research we asked for the effects of firm concentration on the Hayekian process of knowledge generation in market competition. Our simulation results suggest that the growth rate of knowledge accumulation increases with the number of independently innovating firms. This result does not come as a surprise, because a larger number of innovating firms leads in our simulation model to a larger variety of products, which are tried out. It can be expected then that the best firm has a higher fitness parameter (than in the case of a smaller number of firms) and the other firms can learn more from the best firm. Although our results are very clear, no quick conclusions should be drawn with regard to merger policy, because this result depends on many specific assumptions such as that the quality of innovations is independent of the number of innovating firms, which implies, for example, that there are no economies of scale in research and development. But our results can support the widespread conjecture that a reduction of the number of independent research paths, for example by mergers or R&D cooperations, can also hamper technological development (Mowery, 1995; Cohen & Malerba, 2001; Carlton & Perloff, 2000, p. 514).
- *The average growth rate of the knowledge accumulation process decreases with the number of activities of the firms which are simultaneously tested in the market.* The basic idea of this hypothesis is that the market test in the form of profits and losses can only feed back information about the overall fitness of all activities of the firms, but not on the fitness of particular activities. The larger the number of activities (e.g., the number

of products the firm sells) and Therefore the larger the bundle of hypotheses which are tested simultaneously in the market, the more unspecific is the information the firms get from the market about the quality of their specific activities. They only know that on average their bundle of hypotheses is better or worse than those of their competitors. Also this hypothesis was confirmed by our simulation results.

- *The non-imitability of at least one activity will have considerable impact on the pattern of the knowledge accumulation process and decrease its average growth rate.* It is a well-known fact in innovation economics that imitation can be time consuming and difficult, or can even fail entirely. Verifying this hypothesis we assume that there is at least one activity of the firms which—due to non-observability or tacit knowledge—cannot be imitated at all by the other firms. Our presumption that the introduction of the non-imitability of activities can have a serious impact on the knowledge accumulation process has been verified clearly by simulation experiments. The results show that even the non-imitability of only one activity can lead to a very large decrease in the growth rate. For example, the average growth rate drops to nearly a third, if only one of seven activities is non-imitable—in comparison to the case of full imitability (Kerber & Saam, 2001, p. 4.9, Table 2). This amazing effect will be the starting point of our research in this chapter. Which effects slow down learning in competition if one activity cannot be imitated?
- *Fallible knowledge of consumers will have considerable impact on the knowledge accumulation process.* Consumers may have fallible knowledge. They might initially make wrong decisions with regard to new products until they have collected enough experience. This may imply difficulties and lags concerning the quality of the information feedback about the performances of firms for solving the problems of consumers. The presumption that the introduction of fallible knowledge of consumers can have a serious impact on the knowledge accumulation process has been confirmed clearly by simulation experiments. The simulations reveal that under certain conditions, fallible knowledge of consumers decreases the growth rate of knowledge accumulation in the market, but under certain other conditions, it even increases the growth rate of knowledge accumulation (Saam, 2005).[3] The decrease of knowledge accumulation due to limited imitability is sometimes overcompensated by the second imperfection, fallible knowledge. Theoretically, it hints to the possibility of compensating one type of market failure (limited imitability of innovations by producers) by another (fallible knowledge of consumers).

CHANGE OF "LEADING FIRMS," LOCK-IN PROBLEMS, AND KNOWLEDGE ACCUMULATION IN SITUATIONS OF LIMITED IMITABILITY: HYPOTHESES

Which effects slow down learning in competition if one activity cannot be imitated? Our first explanation (Kerber & Saam, 2001) was that knowledge accumulation is slower because no *mutual learning* can take place with regard to the non-imitable activity. However, we found that this effect cannot explain the whole reduction in knowledge accumulation. So, we looked for additional systematic effects that might explain the remaining difference.

Second, we presumed that there is an effect which is caused by a kind of lock-in situation

which can emerge under certain conditions in knowledge-generating competition processes: The leading firm may have the highest total fitness because the fitness of its non-imitable activity is the *highest among all activities—whether imitable or not—of all firms*. Assume, for example, the leading firm is best because of its superior organization culture, which might be an activity that cannot be imitated by other firms. We call such a situation a "lock-in situation," because it may hamper knowledge-generating in the Hayekian competition processes.[4] The problem is that in case of a lock-in, the competitors have no possibility to catch up through imitation because the best firm is the leading one due to the non-imitable activity. This problem can be aggravated by the effect that the non-leading firms imitate all the imitable activities of the leading firm which even on average may have an inferior fitness than those of the competitors. If this case turns up, the whole process of mutual learning is severely hampered.

Third, we presumed that as the superior hypothesis of the leading firm cannot be imitated, it is much more difficult to switch the leading firm. Only when the leading firm changes frequently does this firm have the chance to learn from its competitors. Take, for example, firm A which was previously the leading firm, but was caught by firm B. Whereas firm B (and also firms C, D, etc.) learned from firm A, now firm A will learn from firm B and may even catch up with firm B in this time period; therefore, firm A may again be in the lead during the next time period. Then, firm B will learn again from firm A and may catch up again, and so on. As a consequence, the Hayekian mechanism of mutual learning in competition works best if the leading firm changes frequently. The change is hampered in case of lock-ins: the leading firm's removal is much more difficult, although it is not impossible. In this chapter, we therefore test the following hypotheses:

- **Hypothesis I:** There is a negative correlation between the number of lock-in situations and knowledge accumulation in markets that are characterized by limited imitability of firms' activities.
- **Hypothesis II:** There is a negative correlation between the number of lock-in situations and the number of changes of the leading firm in markets that are characterized by limited imitability of firms' activities.
- **Hypothesis III:** There is a positive correlation between the number of changes of the leading firm and knowledge accumulation in markets that are characterized by limited imitability of firms' activities.

THE MODEL

Let us assume that in a market, n firms exist. Each of them produces one product that consists of a complex bundle of traits and can be produced and marketed by using a set of m activities. Each of these activities can be performed in a different way. It depends on the specific knowledge of the firms—that is, their hypotheses about the effects of these activities and how they deem it best to apply these activities. Each of these m activities contributes to the total performance of a firm which is decisive for its competitiveness and success in the market. From an evolutionary perspective, the total performance of a firm is interpreted as its total fitness, and the contribution of a particular activity to the total performance is the fitness of this activity. The total fitness of a firm shall be calculated as the arithmetical mean of the fitnesses of all activities.

If f_{ijt} denotes the fitness of activity a_j ($j = 1 \ldots m$) of firm i at time t and F_{it} the total fitness of firm i ($i = 1 \ldots n$) at time t, it follows:

$$F_{it} = \frac{1}{m}\sum_{j=1}^{m} f_{ijt} \qquad (1)$$

Let G_t denote the average total fitness of all firms in the market at time t:

$$G_t = \frac{1}{n}\sum_{i=1}^{n} F_{it} \qquad (2)$$

This variable, the average total fitness of all firms in the market, will be used to measure the knowledge accumulation in the market. In the model, the development of the total fitness of the firms is analyzed over a number of periods in which the firms attempt to improve the fitness of their activities by innovation and imitation. Every period can be seen as consisting of two phases, an innovation phase and an imitation phase, through which the old values of the fitness parameters f_{ijt} are transformed into the new ones f_{ijt+1}.

Basic Model: Innovation Phase

During the innovation phase, all firms search for new and better hypotheses for applying their m activities in order to improve their fitness. These innovations are modeled as stochastic processes of drawing from sets of potential routines for each activity. The existence of m activities implies m independent stochastic innovation processes.[5] This chapter assumes that the fitness of the newly drawn routines of the activities are normally distributed with variance σ^2 and expected value $E(f'_{ijt})$. Standard deviation σ ($\sigma > 0$) can be interpreted as innovation (mutation) rate. f'_{ijt} denotes the fitness of an activity and F'_{it} the total fitness of firm i after innovation but before imitation in time period t. In order to isolate the effect of mutual learning in the competition process, we assume that the expected value of the stochastic innovation process is identical with the fitness value of that activity in the last period:

$$E(f'_{ijt}) = f_{ij,t-1} \qquad (3)$$

This assumption implies that the probability of improving or worsening the fitness of activities by innovation is the same. Consequently, an isolated firm that cannot learn from others cannot expect to improve its fitness in the long run.

Following Hayek, we assume that knowledge about the superiority of newly drawn routines can only stem from market feedback. Only those firms that succeed and make profits are superior to their competitors. The firm with the highest total fitness will make profits, and those with lower ones: losses or smaller profits. Firms are only informed about the profits and losses of all firms. From the ranking of profits, they can deduce the ranking of the total fitness values F'_{it} of all firms. Both kinds of rankings are assumed to be identical.[6] Firms do not know the real values of the total fitness F'_{it}, not even their own. Particularly, they do not get any feedback about the fitness of their single activities f'_{ijt} (or f_{ijt}). The only information the market feeds back to them via profits or losses is whether their total fitness is larger or smaller than that of their competitors. This is the logical consequence of the Hayekian notion that only market feedback by profits and losses can be seen as the real test of the superiority or inferiority of the performance of firms. Therefore, firms are faced with the difficult problem that they do not really know why they are successful or why they fail. They do not know which of their activities (e.g., the quality or design of the product, marketing, service, or price policy) are responsible for the observable fact that they are better or worse than their competitors. They can only conclude that their set of routines (and therefore the quality of their

knowledge) is on average better or worse than those of their competitors. This is the starting point for the second phase, the imitation phase.

Basic Model: Imitation Phase

Since the ranking of the total fitness of firms is revealed by the market test, all firms are able to identify the best firm. We assume that all firms are able to observe the way in which all other firms use their activities—they know their products, production technologies, marketing strategies, and that they have sufficient competence to imitate the activities of the other firms. One strategy for improving one's fitness is to imitate the activities of the best firm. Therefore, we assume that the non-leading firms imitate the activities of the best firm. We know from innovation economics that imitation is not an easy task, because imitation takes time and may even fail. Therefore we assume that the non-leading firms are only able to imitate the activities of the best firm to a certain extent. Therefore, we introduce an imitation rate λ ($0 < \lambda < 1$) which indicates the percentage by which the non-leading firms are able to catch up on the difference between their own fitness values and those of the best firm. For example, an imitation parameter value of 0.5 implies that each non-leading firm can catch up 50% of the difference between its fitness values and those of the best firm.

Let k denote the best (or leading) firm and $f'_{kjt}{}^*$ and $F'_{kt}{}^*$ the fitness value of activity j and total fitness of the best firm k *after innovation*:

$$F'_{kt}{}^* = \max_i F'_{it} \text{ with } (i = 1, \ldots k, \ldots, n) \quad (4)$$

Then, the imitation process is described by equation 5:

$$f_{ijt} = \begin{cases} f'_{ijt} + \lambda\left(f'_{kjt}{}^* - f'_{ijt}\right) & \text{if } F'_{kt}{}^* > F'_{it} \\ f'_{ijt} & \text{else} \end{cases} \quad (5)$$

Whether a firm imitates another firm's activities is a discrete problem. If the total fitness of firm i after innovation in period t is smaller than the total fitness of the best firm F'_{kt}, firm i will try to imitate all activities of the best firm k. As a consequence, all fitness values of its activities will increase by a certain percentage of the difference between the fitness values of its activities and those of the leading firm (imitation rate λ). However, the best firm will not change its activities ($f_{kjt} = f'_{kjt}$).

Our information assumptions have an important implication. We cannot conclude that all fitness values of activities of the non-leading firms are always smaller than those of the leading firm. The leading firm has only a higher value of the total fitness. This implies that in regard to specific activities, non-leading firms might have higher fitness values than the leading firm. For example, a non-leading firm may have a better marketing strategy than the best firm, but may have an inferior total performance because of low quality of the product it sells. This implies that in regard to particular activities, the non-leading firms might reduce their total fitness by imitating the best firm. And the larger the number of activities, the higher is the probability for this case. But, since firms do not know which of their activities are better or worse, they will improve their total fitness only by imitating all activities of the best firm. Table 1 gives an overview of all variables and parameters of the model, as well as their interpretation and initialization.

Modification of the Basic Model: Limited Imitability

The imitability of all activities can be a critical assumption that need not be fulfilled in competition processes. For analyzing the impact of the non-imitability of activities of firms on the average total fitness of the firms, this chapter modifies the basic simulation model by differ-

Table 1. Variables and parameters of the formal model

Variable	Meaning	Initialization at Time t = 0
f_{ijt}	fitness of activity j of firm i at time t	1.0 for each activity
F_{it}	total fitness of firm i (i = 1 ... k, ... n) at time t	1.0 for each firm
F_{kt*}	total fitness of best firm k at time t	1.0
G_t	average total fitness of the industry at time t	1.0
Parameters	**Interpretation**	
λ	imitation rate of the firms ($0 < \lambda < 1$)	$\lambda = 0.5$
σ	innovation rate of the firms ($\sigma > 0$)	$\sigma = 0.05$
m	number of activities of the firms	$1 \geq m \geq 11$*; m = 7**
n	number of firms in the market	$2 \geq n \geq 10$*; n = 6**

* *Initialization in Kerber and Saam (2001) and Saam (2005—our previous research) for which the above reported results hold*

** *Initialization in the experiments which are analyzed in Basic Model: Imitation Phase section (see Modification of the Basic Model: Limited Imitability section)*

entiating between imitable and non-imitable activities. For simplicity, we assume that one of the activities—for example, the organization culture of a firm—cannot be imitated.[7] Let m_v denote the number of imitable and m_s the number of non-imitable activities ($m_v + m_s = m$). In the modified simulation model, we analyze the impact of one ($m_s = 1$, $m_v = m - 1$) non-imitable activity on knowledge accumulation.

EXPERIMENTS AND RESULTS

We analyze the simulation results[8] of two scenarios—full imitability and limited imitability of activities—applying a new meta-model, multivariate regression analysis. We assume a market with $n = 6$ firms and $m = 7$ activities ($m_s = 1$ and $m_v = 6$ in the limited imitability scenario). In each simulation run, 100 periods which consist of an innovation and an imitation phase are simulated. The results of each scenario are mean values and standard deviations of 80 simulation runs which start from different random seeds, all else being equal. Two auxiliary variables have been coded as dummy variables that count the incidence of a lock-in or of a change of the leading firm during each simulation run. A lock-in is operationalized as a situation in which the non-imitable activity of the leading firm is the *highest among all activities of all firms*. Then, the competitors cannot catch up through imitation.

As can be seen from Table 2, the average number of changes in the leading firm decreases dramatically from 61.2 in the full imitability scenario to 30.3 in the limited imitability scenario. This means that in 61 (respectively 30) out of 100 periods of time, the leading firm changes from the previous to the present period of time. In both cases we have a process of knowledge accumulation, because the average total fitness of firms (G_t) rises from 1.0 in period 1 to 2.39 (respectively 1.57) in period 100. But in the limited imitability scenario, this increase is much smaller than in the full imitability scenario (1.57 instead of 2.39).

In the full imitability scenario, lock-ins are not possible. The simulation runs of the limited imitablility scenario are classified into three types depending on the number of lock-ins during each simulation run. Sixty out of 80 simulation runs have no lock-in or few, short-term lock-ins (0-10). Seven out of 80 simulation

*Table 2. Lock-ins, changes in the leading firm and average fitness of firms on the market (n = 6 firms, m = 7 activities)**

		Full Imitability ($m_v = 7, m_s = 0$)	**Limited Imitability ($m_v = 6; m_s = 1$)**			
Simulation runs		80	80	thereof 60	7	13
				0-10 Lock-ins	*11-25 Lock-ins*	*Lock-ins > 25*
Changes in the leading firm	Mean (Stddv.)	61.2 (4.9)	30.3 (11.1)	33.5 (9.5)	22.0 (8.6)	19.8 (10.1)
Average total fitness of firms (G_t)	Mean (Stddv.)	2.39 (0.11)	1.57 (0.23)	1.64 (0.18)	1.44 (0.16)	1.33 (0.24)

* *Initialization of parameters:*
Full imitability scenario: $\lambda = 0.5$, $\sigma = 0.05$, $n = 6$, $m_v = 7$, $m_s = 0$;
Limited imitability scenario: $\lambda = 0.5$, $\sigma = 0.05$, $n = 6$, $m_v = 6$; $m_s = 1$;
Initialization of variables: see Table 1 for both scenarios.

runs have frequent lock-ins, sometimes long lasting (11-25). Thirteen out of 80 simulation runs have long-lasting lock-ins (> 25). The higher the number of lock-ins, the smaller is the number of changes in the leading firm and the average total fitness of firms.

The data of these 160 simulation runs have been analyzed applying multivariate regression analysis. In case of heteroscedasticity we repeated the regressions and used a robust estimator (White's procedure). As can be seen from Table 3, in the full imitability scenario there is a non-significant, small, negative effect of the number of changes in the leading firm on the average total fitness of firms. In case of limited imitability, there is a significant, small, and negative effect of the number of lock-ins on the average total fitness of firms (which confirms our Hypothesis I). There is a significant, positive effect of the number of changes in the leading firm on the average total fitness of firms (which confirms our Hypothesis III). There is a significant, negative effect of the number of lock-ins on the number of changes in the leading firm (which confirms our Hypothesis II). In the multivariate regression that includes the number of lock-ins and the number of changes in the leading firm as independent variables, the number of lock-ins has a significant, negative effect on the average total fitness of firms, whereas the number of changes in the leading firm has a significant, positive effect on the average total fitness of firms.

The effect of lock-ins on the average total fitness of firms can be explained as follows. If an activity cannot be imitated, lock-ins reduce the average total fitness of firms directly as well as indirectly by reducing the number of changes in the leading firm. The direct effect is that the competing firms cannot imitate the best activity of the leading firm because it is the non-imitable one. They can only imitate the other activities of the leading firm. The indirect effect comes as a consequence of the direct effect. The competitors need more time to catch up to the leading firm. The leading firm will lead for more periods of time. The changes in the leading firm are reduced.

In case of limited imitability, changes in the leading firm increase the average total fitness

Table 3. Multivariate regression analysis of a simulated market (n = 6 firms, m = 7 activities)

Imitability	**Full ($m_v = 7$, $m_s = 0$)**		**Limited ($m_v = 6$; $m_s = 1$)**							
Dependent variable	Average total fitness of firms		Average total fitness of firms		Average total fitness of firms		Changes in the leading firm		Average total fitness of firms	
	b-Value*	T-Value	b-Value	T-Value**	b-Value	T-Value	b-Value	T-Value	b-Value	T-Value
Changes in the leading firm	-0.002 (-0.076)	-0.67			0.013 (0.647)	7.49			0.009 (0.451)	4.80 (4.46)
Lock-ins			-0.007 (-0.608)	-6.76 (-9.36)			-0.294 (-0.533)	-5.57 (-7.36)	-0.004 (-0.367)	-3.91 (-4.43)
R-Square	0.006		0.369		0.418		0.285		0.515	
F-Value	0.453		45.683		56.087		31.013		40.849	
Significance***	0.503		0.000		0.000		0.000		0.000	

* *Standardized coefficients in brackets*

** *Robust T-Value according to White's procedure in brackets*

*** *The data come from a parsimonious simulation model and not from a far more rich reality. Therefore, the variability of the data is comparatively small. As a consequence, significances are far more distinct than they were in empirical social or economic research.*

of firms, because lock-ins occur more seldom if changes in the leading firm happen more frequently. It is less probable that the non-imitable activity is always the best in the market, if the leading firm changes often. In the full imitability scenario, changes in the leading firm do not interact with lock-ins. This explains why here changes in the leading firm have no significant effect on the average total fitness of firms.

In a less technical language, in the limited imitability scenario, the Hayekian mechanism of mutual learning in competition is hampered. As the superior hypothesis of the leading firm cannot be imitated, it is much more difficult to switch the leading firm. Only when the leading firm changes frequently does this firm have the chance to learn from its competitors. Take, for example, firm A which was previously the leading firm, but has been caught by firm B. Whereas firm B (and also firms C, D, etc.) learned from firm A, now firm A will learn from firm B and may even catch up with firm B within this time period; therefore firm A may be the leading firm again during the next time period. Then, firm B will learn again from firm A and may catch up again, and so on. As a consequence, the Hayekian mechanism of mutual learning in competition works best if the leading firm changes frequently. The change is hampered in case of lock-ins: the leading firm's removal is much more difficult although it is not impossible. Since stochastic innovation processes still take place with all imitable and non-imitable activities, it is not probable that the best firm will be leading permanently due to its superior non-imitable activity. Consequently, lock-in situations will also terminate. But, during all those consecutive periods when firm A is leading, yet not caught, it will only improve its fitness by chance. The knowledge accumulation process of the whole industry will slow down, because firm A gets no more chance to learn from others.

CONCLUSION

Although the basic idea of Hayek's competition as a process of parallel experimentation, in which new knowledge is generated through a

trial-and-error-process, is well known and seems to be broadly accepted on a common sense level by most economists, so far there have been no real attempts to analyze his notion of competition as a discovery procedure in a more rigorous, analytical way. In this chapter we presented a formalized model of Hayek's concept which was analyzed by using a simulation approach. Our simulation model is an example of theory-driven modeling that aims at developing new hypotheses on mechanisms that work in markets.

In previous research (Kerber and Saam, 2001), it was shown that the growth rate of knowledge accumulation in this Hayekian approach to competition is positively correlated with the number of independently innovating firms and negatively correlated with the number of hypotheses, which are simultaneously tested in the market. The starting point for the analyses in this chapter was the surprising result in our previous research that the non-imitability of only one activity leads to a very severe reduction in the growth rate of knowledge accumulation. Which effects slow down learning in competition if one activity cannot be imitated? In this chapter we have shown that the non-imitability of one activity not only reduces the extent of mutual learning, because this activity cannot be imitated, but the non-imitability of activities generates phenomena which hamper learning additionally:

1. A certain kind of lock-in situation can emerge, if the leading firm is the best because of this non-imitable activity. Then, the *competitors* cannot catch up through imitation. They may only catch up through innovation, which takes more time because it is a mere trial-and-error-process. Mutual learning is hampered on the side of the competitors.
2. These lock-in situations reduce the number of changes of the leading firm. Only when the leading firm changes frequently does this firm have the chance to learn from its competitors. This time, mutual learning is hampered on the side of the *leading firm*. The leading firm will for long periods of time innovate without getting the chance to learn from others. As a result, the whole process of mutual learning is hampered. In Hayekian market process theory, these are the effects that slow down knowledge accumulation in competition under limited imitability assumptions in a severe way.

REFERENCES

Aghion, P., & Howitt, P. (1998). *Endogenous growth theory*. Cambridge, MA: MIT Press.

Arthur, B. (1989). Competing technologies, increasing returns, and lock-in by historical events. *Economic Journal, 99,* 116-131.

Carlton, D. W., & Perloff, J. M. (2000). *Modern industrial organization* (3rd ed.). Reading, MA: Addison-Wesley.

Cohen, W. M., & Malerba, F. (2001). Is the tendency to variation a chief cause of progress? *Industrial and Corporate Change, 10,* 587-608.

David, P. A. (1985). Clio and the economics of QWERTY. *American Economic Review, 75,* 332-337.

Dosi, G. (1988). Sources, procedures and microeconomic effects of innovation. *Journal of Economic Literature, 26,* 1120-1171.

Harper, D. A. (1996). *Entrepreneurship and the market process, an enquiry into the*

growth of knowledge. London/New York: Routledge.

Hayek, F. A. (1948). The meaning of competition. In F. A. Hayek (Ed.), *Individualism and economic order* (pp. 92-106). Chicago: Chicago University Press.

Hayek, F. A. (1978). Competition as a discovery procedure. In F. A. Hayek (Ed.), *New studies in philosophy, politics, economics and the history of ideas* (pp. 179-190). Chicago: University of Chicago Press.

Helbing, D., Treiber, M., & Saam, N.J. (2005). Analytical investigation of innovation dynamics considering stochasticity in the evaluation of fitness. *Physical Review E, 71,* 067101.

Kerber, W. (1997). Wettbewerb als hypothesentest: Eine evolutorische konzeption wissenschaffenden wettbewerbs. In K. Delhaes & U. Fehl (Eds.), *Dimensionen des wettbewerbs: Seine rolle in der entstehung und ausgestaltung von wirtschaftsordnungen* (pp. 29-78). Stuttgart: Lucius & Lucius.

Kerber, W., & Saam, N. J. (2001). Competition as a test of hypotheses: Simulation of knowledge-generating market processes. *Journal of Artificial Societies and Social Simulation, 4*(3). Retrieved from http://jasss.soc.surrey.ac.uk/4/3/2.html

Kirzner, I. M. (1997). Entrepreneurial discovery and the competitive market process: An Austrian approach. *Journal of Economic Literature, 35,* 60-85.

Mantzavinos, C. (2001). *Individuals, institutions, and markets*. Cambridge, UK: Cambridge University Press.

Metcalfe, S. J. (1989). Evolution and economic change. In A. Silberston (Ed.), *Technology and economic progress* (pp. 544-585). London: Macmillan.

Motta, M. (2004). *Competition policy*. Cambridge, UK: Cambridge University Press.

Mowery, D. (1995). The practice of technology policy. In P. Stoneman (Ed.), *Handbook of the economics of innovation and technology change* (pp. 513-557). Oxford: Blackwell.

Nelson, R. R., & Winter, S. G. (1982). *An evolutionary theory of economic change*. Cambridge: Belknap Press.

Popper, K. (1972). *Objective knowledge. An evolutionary approach*. Oxford: The Clarendon Press.

Saam, N. J. (2005). The role of consumers in innovation processes in markets. *Rationality & Society, 17,* 343-380.

Schumpeter, J. A. (1934). The theory of economic development. An inquiry into profits, capital, credit, interest, and the business cycle. Cambridge, MA: Harvard University Press.

ENDNOTES

[1] See Hayek (1978) for concepts that are based upon or very close to this Hayekian concept; also see Harper (1996), Kirzner (1997), Kerber (1997), Mantzavinos (2001).

[2] We treat fitness as an abstract concept, like the utility concept. In a general model like the one we are presenting here, it is difficult to operationalize. However, in applications of our model, it can be operationalized. We suggest use of indicators that measure the (indirect) effects of knowledge, for example in terms of costs instead of (direct) measures of knowledge. An effect of increasing knowl-

edge might be decreasing production cost—that is, average costs could be used as an indicator of fitness.

[3] Modifying some of the assumptions, this result can also be obtained by an analytical model without simulation (see Helbing, Treiber, & Saam, 2005).

[4] Please note that this concept of "lock-in" situations differs from those of David (1985) and Arthur (1989), who define a lock-in as a standardization on an inefficient technology.

[5] This is the most simple assumption one can start with. Whether the activities are independent or not, whether their weight is equal (as we assume) or not, are open questions that might also depend on the specific market. Testing the implications of alternative assumptions is a topic for future research.

[6] The assumption that the ranking of the total fitness of the firms is identical with the ranking of their profits implies that the selection of the market works appropriately—that is, there are no systematic distortions by technological external effects, market power, rent seeking revenues, and so on. This is doubtlessly a critical assumption, which has to be held in mind for further scrutiny.

[7] Please note that the non-imitable activity is the same for all firms. This is not a necessary assumption, because it also might be possible that, due to different competences of the firms, some firms might be able to imitate certain activities and others may not.

[8] The simulations have been implemented on an agent-based framework.

Chapter XXV
On Technological Specialization in Industrial Clusters

Herbert Dawid
Bielefeld University, Germany

Klaus Wersching
Bielefeld University, Germany

ABSTRACT

In this chapter an agent-based industry simulation model is employed to analyze the relationship between technological specialization, cluster formation, and profitability in an industry where demand is characterized by love-for-variety preferences. The main focus is on the firms' decisions concerning the position of their products in the technology landscape. Different types of strategies are compared with respect to induced technological specialization of the industry and average industry profits. Furthermore, the role of technological spillovers in a cluster as a technological coordination device is highlighted, and it is shown that due to competition effects, such technological coordination negatively affects the profits of cluster firms.

INTRODUCTION

Regional or local agglomeration of firms can be identified in almost every economy of the world: Hollywood, Silicon Valley, and Route 128 in the United States, the finance sector in London (GB) or Frankfurt (D), and the pharmaceuticals near Basel (CH) are only a few examples. Since Marshall (1920), the economic rationales for agglomeration of firms are discussed in the economic literature, and more recently there emerged an amount of literature which focused on the relationship between agglomeration and innovation (see Audretsch, 1998; Asheim & Gertler, 2005).

Main arguments in favor of a geographic concentration of economic activity are positive externalities associated with the proximity of related industries (e.g., Porter, 1998). Beside transactional efficiencies, knowledge spillovers are one form of these externalities. The concentration of firms and workers allows the

exchange of tacit knowledge, which is characterized as vague, difficult to codify, and uncertain in its economic use (Dosi, 1988). Because of these properties tacit knowledge is best transmitted via face-to-face interaction and through frequent and repeated contact. Examples of transfer channels of knowledge spillovers are planned or unscheduled communication between people or the flow of workers from one company to another. Despite the advances in telecommunication technology, there is evidence that geographic proximity leads to a faster diffusion of knowledge (e.g., Jaffe, Trajtenberg, & Henderson, 1993) and that the costs of transmitting tacit knowledge rise sharply with geographical distance (Audretsch, 1998). Hence geographical proximity favors the flow of knowledge.

Arguably, the flow of knowledge between companies is, however, not only influenced by their geographical but also by their technological distance. Accordingly, the degree of technological specialization in a cluster should be of relevance for the intensity of technological spillovers, and several authors have studied the impact of technological specialization on the size of local knowledge spillovers.

One view of the topic was described as the Marshall-Arrow-Romer model by Glaeser, Kallal, Scheinkman, and Shleifer (1992). It is argued that technological specialization facilitates knowledge spillovers between firms of the same industry. The similarity of knowledge and activities promotes the learning effect between individuals and firms. Empirical support for these claims was given, for example, by van der Panne (2004). In a recent study Cantner and Graf (2004) provide further empirical evidence concerning specialization and cooperation. In their work, cooperation is measured in the way that the number of participating firms on assigned patents is counted. The authors find that technological moderately specialized regions show the highest number of research cooperatives, and the higher a region's specialization, the more cooperatives are formed between partners outside that region. Taking cooperatives as a proxy for knowledge spillover, this result indicates that the exchange of knowledge is highest in a moderately specialized cluster.

By contrast, Jacobs (1969) argues that knowledge may spill over between complementary rather than similar industries. Ideas developed by one industry can be useful for other industries, and therefore technological diversity favors knowledge spillovers. According to Jacobs, a variety of industries within a cluster promotes knowledge externalities. The diversity thesis was also supported in empirical works (e.g., Feldman & Audretsch, 1999).

Although there is no complete consensus, there is some evidence that some technological specialization among firms of the same industry in a cluster has positive effects on spillovers. In this chapter we adopt this view, but also take into account that firms in a cluster are not only producers and receivers of knowledge flows, but also competitors in the market. Strong technological specialization within a cluster leads to little differentiation between the products of the cluster firms and hence to increased competition among them. This is particularly true if we think of clusters which primarily serve local markets, or industry-dominating clusters like the Silicon Valley. Hence the positive effect of intensive knowledge exchange in specialized clusters may be countered by negative competition effects. Competition considerations are also an important factor in determining which firms decide to enter a cluster in the first place. Knowledge spillovers always flow in two directions. Thus a firm cannot prevent knowledge from spilling over to possible competitors in the cluster. A firm inhabiting a particularly profitable market niche or enjoying

a technological lead might have strong incentives to choose geographical isolation in the periphery rather than joining a cluster.

The agenda of this chapter is to gain a better understanding of the interplay of positive externalities due to knowledge spillovers in a cluster, market competition, and the self-selection effects generated by endogenous locational choices of producers. In particular, we study how different firm strategies concerning their choice of technology and product position influence the degree of specialization in a cluster, the relative performance inside and outside the cluster, and the overall industry performance. Our research strategy is to employ a dynamic agent-based industry simulation model where firms compete by offering various differentiated product-variants which are based on different production technologies. The number and location of product variants offered are determined by product innovation activities in the industry and vary over time. Firms make decisions concerning their geographic location (inside or outside a cluster) and concerning the location of their products in the technology space. Positioning in a cluster means that a firm receives technological spillovers from other cluster firms, but also implies knowledge flows from the firm to other ones in the cluster. Our simulation results highlight significant downsides of strong specialization. In particular we observe that positioning strategies leading to highly specialized clusters perform poorly with respect to absolute firm profits and with respect to the relative performance of cluster firms compared to those in the periphery.

It should be mentioned that in our analysis, only the interplay of positive knowledge externalities generated by strong specialization and negative competition effects are analyzed, whereas other potential positive effects of specialization within clusters are not taken into account. Among these potentially positive effects are the availability of specific input factors like machinery, business services, and personnel (e.g., Krugman, 1991; Porter, 1998).

The chapter is organized as follows: the next section describes the main features of our agent-based simulation model. Our findings concerning specialization and firm strategy are then presented, and we conclude with some discussion of the results and possible future research topics.

AN AGENT-BASED SIMULATION MODEL OF CLUSTER FORMATION AND INDUSTRY EVOLUTION

This section introduces an agent-based simulation model to study the technological development and the economic performance of firms with potential knowledge spillover in a differentiated industry. The analysis is based on the interaction and behavior of firms, which might share knowledge but at the same time are competitors in the goods markets. The model is partly based on previous work by Dawid and Reimann (2004, 2005). Due to space constraints, no full description of the model can be provided here, but we will restrict ourselves to highlighting some main features. See Wersching (2005) for a detailed discussion of the model.

Main Features of the Model

We consider an industry consisting of F firms,[1] where each firm in each period offers a range of differentiated products. To keep things simple it is assumed that product differentiation is entirely due to technological differences between products, and hence product positioning is equivalent to technological positioning of firms. At any point in time, the industry consists of several sub-markets located in the one-

dimensional (circular) technological space. Demand is derived from love-for-variety preferences of consumers augmented by an attractiveness parameter for each product variant. The attractiveness of a variant is influenced by the technological distances to the closest technological neighbors currently employed in the industry. The greater the product of the distances, the greater is the market niche and therefore the larger the attractiveness parameter. The overall amount of money allocated by consumers to purchase goods produced in the industry increases with the number and total attractiveness of product variants, however at a decreasing rate. Concerning geographic location of firms, the only distinction is between firms in the cluster and firms in the periphery. As described below, the geographic location of a firm determines whether the firm can exchange knowledge with other producers.

An important feature of the model is that we incorporate the accumulation of a structured stock of knowledge of each individual firm. Due to investments for process and product innovation and to technological spillovers, firms build up technology-specific stocks of knowledge for each of the product variants they are currently producing or plan to introduce to the market in the future. Two types of spillovers are considered. First, a firm can transfer some knowledge internally from one technology to another. Second, there are external spillovers between all firms in the cluster. Two factors influence the intensity of the knowledge flow between firms. First, the flow increases with decreasing technological distance (measured on the circular technology space) between the technologies employed by the firms. Second, there is a hump-shaped relationship between the technological gap, which refers to the difference in the amount of (technology-specific) knowledge that has been accumulated by the two firms, and the intensity of knowledge flows. Accordingly, knowledge spillovers are particularly high if the cluster consists of firms which employ a similar set of technologies, but where in each of these technologies one firm has considerable knowledge advantages.

Knowledge can be used either for process or product innovations. Process innovations reduce production cost as the specific knowledge stock for a certain sub-market increases. While process innovations are certain, investments in product innovations are risky, but expenses enhance the probability for product innovation. Successful product innovations generate a new sub-market in the industry, which is located on the circular technological space between two existing technologies. If the product innovation is radical, the circular technological space expands, so there will be a new market which initially is far from its neighbors. The firm with a new product innovation is for a certain time monopolist on this new market. Afterwards, other companies can enter and produce this product variant.

The firms' behavior is based on decision rules in the tradition of evolutionary modeling (Nelson & Winter, 1982). Firms have different ways of evaluating sub- markets and locations. Depending on this evaluation they will invest in product and process innovations, enter or exit sub-markets, change their location from cluster to periphery, or vice versa. Each sub-market is associated with a technology, and hence the technological location is set either by market entry in a promising sub-market or by product innovation. For the evaluation of the sub-markets, two aspects are taken into account: the average profits on this market and the technological distance to the main technological expertise. The weights assigned to each of these aspects are important strategy parameters and their impact will be studied in the next section. Quantity decisions are made based on local estimations of demand elasticity and the as-

sumption that all competitors will adapt quantities by an identical factor.

Measuring Specialization

One aim of this chapter is to describe the specialization of the industry. There are two dimensions of specialization in this context, namely the distribution of the locations of the technologies employed in the industry and the variance in the amount of knowledge firms have with respect to different technologies. Hence, the following two properties make an industry *specialized*. First, the technologies associated with the active sub-markets used in the industry are clustered (in one or several clusters) rather than being evenly distributed in the technology space. Second, there is a significant variance in the stock of technology-specific knowledge of firms across active sub-markets with accentuated peaks at certain technologies. In order to capture these different aspects of specialization, two different versions of a specialization index are constructed where both are transformations of the Hirschmann-Herfindahl-Index, which has been used in past empirical (Henderson, Kuncoro, & Turner, 1995) or simulation (Jonard & Yildizoglu, 1998) work dealing with specialization. To be able to compare degrees of specialization across different settings with different numbers of existing variants, the Hirschmann-Herfindahl-Index is normalized in the way that it always has the range [0,1].

In particular we use the index

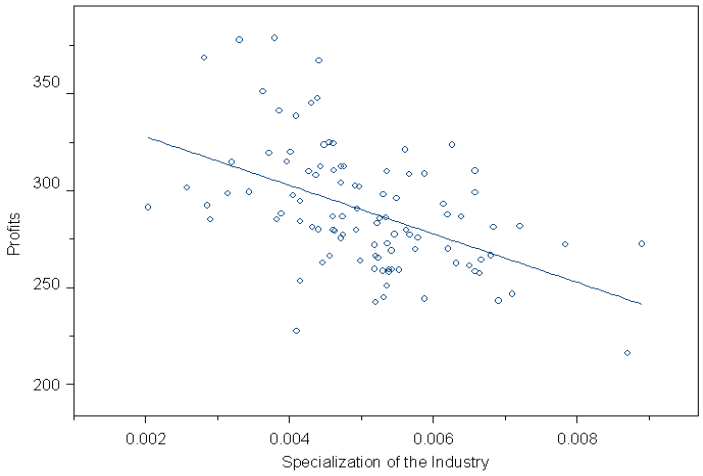
$$Specialization_t = \left[\frac{\sum_{j=1}^{N} {a_{j,t}}^2}{\left(\sum_{j=1}^{N} a_{j,t} \right)^2} - \frac{1}{N} \right] \cdot \frac{1}{1 - \frac{1}{N}},$$

where N is the number of technologies currently used by the considered group of firms, and the interpretation of $a_{j,t}$ depends on the type of specialization. In order to capture the degree of clustering of technologies employed in the population, we set $a_{j,t} = d^{tech}_{j,j+1,t}$, which gives the distance between technology j and the clockwise seen next active technology in the circular technology space in period t. In order to capture specialization in knowledge stock, we set $a_{j,t} = \max_{i=1,..,F} RD_{i,j,t}$, where $RD_{i,j,t}$ denotes the stock of knowledge specific for technology j that firm i has at time t. An aggregated index, which represents the total specialization, is the arithmetic mean of these two indices.

SPECIALIZATION AND FIRM STRATEGY

Before presenting our results we would like to give a brief description of our simulation setup and strategy. The simulation model employed in this chapter is rather complex and has a large set of parameters. We distinguish three types of parameters: (1) the parameters determining the basic characteristics of the industry (number of firms, degree of love-for-variety in the aggregate demand, market size); (2) technological and knowledge flow parameters (costs, learning curve effects, parameters influencing the size of internal and external spillovers, etc.); and (3) strategy parameters determining behavior of the firms in the industry (in particular the weights in the functions evaluating markets, the propensity to enter new markets, investment in product, and process innovation). In order to derive qualitatively robust results that do not hinge on particular parameter settings, we base our observations on batches of 100 simulation runs carried out for different parameter constellations. The industry structure parameters are fixed across these runs, but

Figure 1. Average aggregated profits and specialization of the industry



parameters of the second and third type are chosen stochastically from certain predefined ranges for each firm in each run. This approach allows us to test whether the impact that a restriction of the range of selected strategy parameters has on various variables of interest is statistically significant across stochastically chosen profiles of the other parameters.

As discussed above, the consideration of market competition might imply negative effects of specialization on profits which run counter to the potential positive effects on knowledge externalities typically stressed in the literature. In order to evaluate this trade-off in the framework of our dynamic industry model, we present in Figure 1 the mean aggregated profits of all firms and the level of specialization of the industry after 100 periods in 100 simulation runs where no restrictions on the ranges of strategy parameters are imposed (balanced strategies).

As can be seen in Figure 1, there appears to be a negative relation between the aggregated profits and specialization of the industry. This is confirmed by estimating a linear regression with average profits as dependent and specialization as explanatory variable, which gives a negative and highly significant coefficient of the specialization index.

Several questions arise here. Is the negative correlation between profits and specialization a phenomenon associated with particular firm strategies concerning sub-market selection? Are there any normative implications for firm strategies? Is this effect more pronounced for firms who stay outside the cluster (as might be expected since these firms do not profit from the positive knowledge flow effects from specialization)?

To study the impact of firms' market selection strategies more closely, we compare results from four different batches of simulations runs. In the first batch the decision of a firm about where to position in the technology space in case of entry into an existing market or product innovation is random.[2] We say that firms use a *random* strategy and use this batch as a reference case. In the other three cases, potential locations in the technology space are selected according to the market evaluation function of the firm. The three batches differ with respect to the range of potential values of

the two parameters determining a firm's market evaluation function: the weight put on current profits in a sub-market and the weight put on the distance of the market from the firm's current main technological expertise (given by the technology where the firm has accumulated the largest stock of knowledge). In the case where both strategy parameters are chosen from their full ranges, we refer to this case as the case with a *balanced* strategy. Furthermore, we will consider two batches of runs where the value of a certain strategy parameter is restricted to a sub-interval of the full range. In one batch the parameter determining the weight of current profits is restricted to a sub-interval in the upper region of the full range. In this case we say that firms have *profit-oriented* strategies. In another batch the weight of current technological expertise in the sub-market evaluation function is restricted to a sub-interval of high values. We say that firms use *knowledge-oriented* strategies. In what follows we compare these sets of 100 data points each with respect to crucial variables like profits and degree of specialization.

Figure 2 compares the industry dynamics under random strategies and profit-oriented strategies with respect to both dimensions of specialization. Like in all figures to come, we show mean values of a batch of 100 runs. Due to our initialization, each firm in the first period has knowledge in only one technology and firms are uniformly distributed over all technologies. Hence, initially the industry is perfectly diversified with a specialization index of zero. The specialization rises slightly as the firms invest in research and development activities. In this environment the initial introductions of new products lead to rapid increases in the specialization indices. One can see that the index for location increases much more accentuated than the index for height. This follows from the fact that knowledge can be increased gradually while the technological distance changes very abruptly by the event of product innovations.

Note that under the random strategy, the event of the first new product introduction occurs much later than in the scenario where firms act in a more rational manner. This is due to the fact that firms spread their efforts in the random scenario over several technological regions and therefore reach the needed threshold for a successful product innovation several periods later. Under the random strategy both

Figure 2. Specialization with random and profit-oriented firm strategy

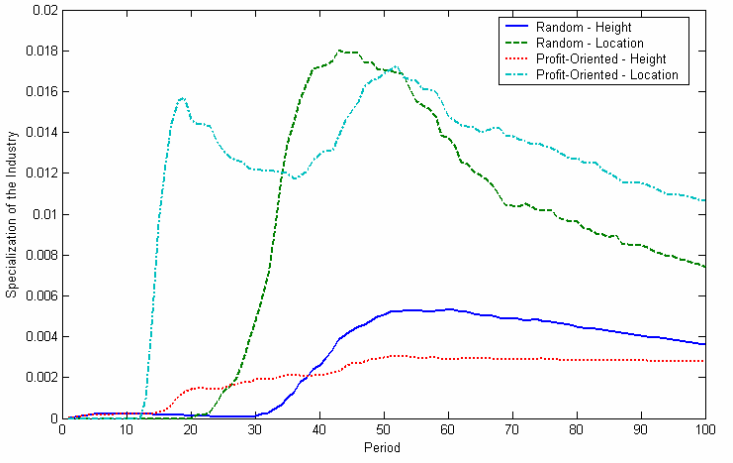
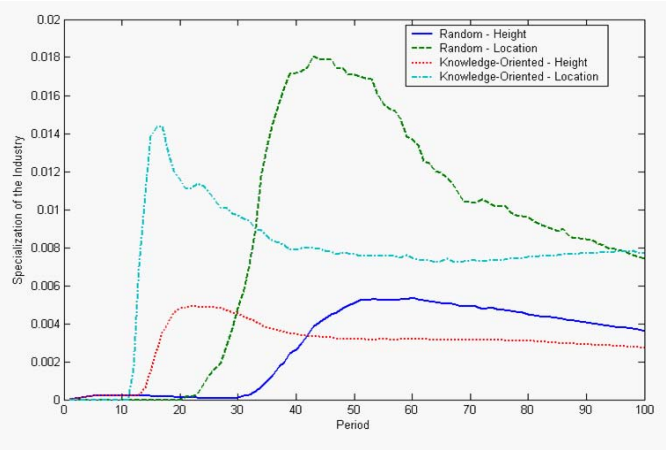


Figure 3. Specialization with random and knowledge-oriented firm strategy



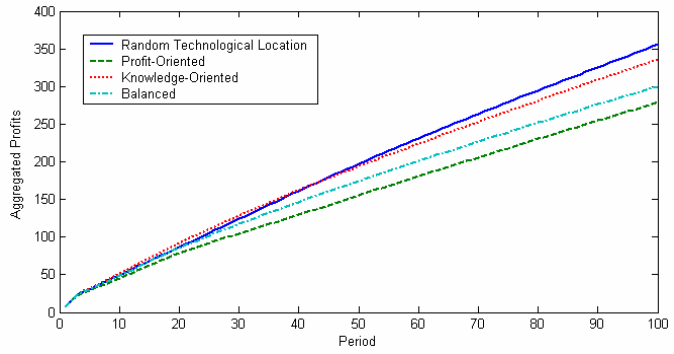
specialization indices slowly decrease after the initial peak. This is quite intuitive, because we should expect that with random strategies, the sub-markets emerging over time will be distributed rather uniformly on the technology space.

If firms use a profit-oriented strategy, a different picture emerges. The specialization of knowledge stock keeps increasing during the entire run, and also the locational specialization index still goes up after the initial peak triggered by the first product innovations. After an initial transient phase, the degree of locational specialization induced by profit-oriented strategies is clearly above the level observed in the reference case with random strategies. On the other hand, the specialization of knowledge stock is smaller than under random strategies.[3] It can be clearly seen that orientation of the firms towards short-run profits leads to almost parallel behavior of firms as far as market selection goes and therefore to strong clustering of the active sub-markets in the industry. Supposedly this has a positive effect on the intensity of spillovers between firms in the cluster. However, the downside of the strongly coordinated behavior induced by profit orientation is that firms also build up knowledge stocks for different technologies at a similar pace, which implies that knowledge stocks are rather uniform in height and there is relatively little specialization of knowledge stock. This has a negative impact on the intensity of spillovers.

The comparison of the effects of random strategies and knowledge-oriented strategies is illustrated in Figure 3. As in the case of profit-oriented strategies, the more strategic behavior leads to earlier introduction of new products which raises the specialization indices. But in this scenario, after an initial phase, both indices lie under the curves of the random strategy.[4] Thus, if firms base their market selection decision mainly on their existing technological strengths, the technological specialization of the industry is smaller compared to the random reference case. Firms initially have heterogeneous technological expertise, and a knowledge-oriented strategy keeps or even reinforces this heterogeneity. This implies that the scope for knowledge spillovers should be small if firms follow a knowledge-oriented strategy.

This discussion shows that the type of market selection strategies firms follow has substantial impact on the degree of specialization arising in the industry. The impact of the differ-

Figure 4. Aggregated profits for different firm strategies



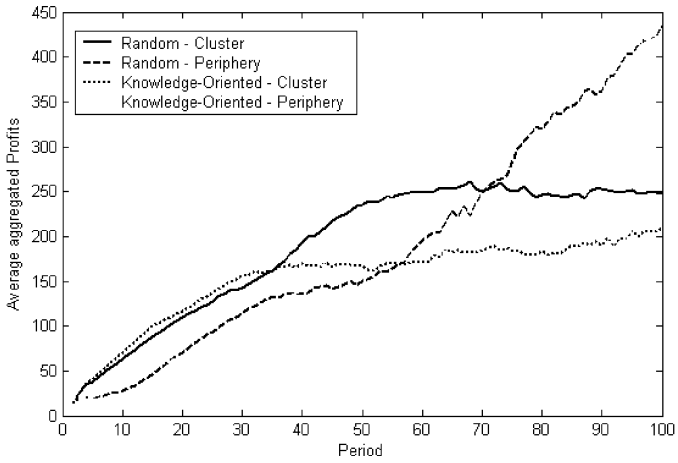
ent strategies on average profits is depicted in Figure 4.

Quite surprisingly, profits are lowest if firms follow a profit-oriented strategy, whereas the largest profits are made in an industry where firms follow a random strategy. The profits under the knowledge-oriented strategy and the balanced strategy lie between these two extremes, where profits under the knowledge-oriented strategy are rather close to those under the random strategy.[5] Considering the effect of competition on the market is crucial to understand this ranking. The assumed love-for-variety structure of demand implies that products far removed in the technology space from the other active sub-markets have high demands and therefore tend to yield high profits. Accordingly, any strategy leading firms to position their products close to the competitors has detrimental effects on revenues. For firms in the cluster, such a strategy on the other hand facilitates cost-reducing technological spillovers, however our results indicate that the negative revenue effect outweighs the positive cost effect. These considerations imply that for all strategies which are not random, the fact that all firms use similar evaluation strategies by itself has negative effects on the average industry performance.[6] We have verified this observation by running simulations where a single firm using the profit-oriented/knowledge-oriented/balanced strategy is inserted into an industry where all other firms employ the random strategy. For all three types of strategies, we found that the non-random strategy performed statistically significantly better than the random strategy.

If we accept the view that the competition effect is dominant in the considered industry scenario and accordingly strong locational specialization has a negative impact on average industry profits, the relative performance of firms in an profit-oriented industry compared to a knowledge-oriented industry can be seen as a direct implication of the different degrees of specialization shown in Figures 2 and 3. However, considering Figure 3, this simple logic fails to explain why the average profits in a knowledge-oriented industry are below those in industries where the random strategy is used. Distinguishing average profits in the cluster and in the periphery under these two strategies sheds some light on this puzzle.

Figure 5 shows that after an initial phase, periphery firms make higher profits than firms in the cluster. Furthermore, firms in the periphery make higher profits in an industry where knowledge-oriented strategies are used com-

Figure 5. Aggregated profits in the cluster and in the periphery under knowledge-oriented and random strategies



pared to the random case. Quite to the contrary, cluster firms are more profitable if all firms employ random strategies. So, the reason why average industry profits are comparatively low under the knowledge-oriented strategy is the poor performance of cluster firms after the initial phase. Essentially, this poor performance is again due to excessive coordination of cluster firms with respect to their technological focus. The reason for the occurrence of such excessive coordination under the knowledge-oriented strategy is the spillover effects. If some cluster firm accumulates a particularly high knowledge stock for a certain technology, spillover effects imply that also all other cluster firms attain a high knowledge stock in that technology region. Hence, the technological focus of many other cluster firms might move into the same technological region, and accordingly these firms will concentrate future production and product innovation in that region. Put differently, the existence of technological spillovers makes it hard for firms to keep their distinctive technological profile if they are in the cluster. So even under strategies like the knowledge-oriented strategy, which per-se does not promote technological uniformity, there is a strong tendency for specialization in the cluster. Indeed, if we restrict attention to cluster firms, the locational specialization index under the knowledge-oriented strategy is larger than under the random strategy for all periods $t \geq 60$. The relatively low industry-wide specialization index under the knowledge-oriented strategy shown in Figure 3 is due to the fact that sub-markets served by the firms in the periphery are far removed from each other and from the technological focus of the cluster firms.

Finally, it should be noted that in spite of the large differences in profits between periphery and cluster firms, there is no strong flow of firms from the cluster to the periphery. Cluster firms could not profit from moving out of the cluster because such a move would not alter the profile of their knowledge stock and accordingly they would still have to produce for sub-markets which are close to the technological focus of the cluster. Even worse, they would lose the technological spillover effects and might eventually be less cost-efficient than the

cluster firms. Hence, there is a technological lock-in effect which ties the firms to the cluster even though periphery firms are more profitable.

CONCLUSION

In this chapter we have used an agent-based simulation model to shed some light on the relationship between technological specialization, cluster formation, and profitability in an industry where demand is characterized by love-for-variety preferences. Our focus was on the effects technological spillovers might have besides the well-documented positive externalities with respect to cost-reducing innovation activities. Our results show that spillovers lead to technological specialization in a cluster even if firms follow strategies which per se do not foster technological uniformity. In the presence of strong competition effects, this has negative implications for the profits of firms, implying that firms in the cluster in the long run perform considerably worse than firms in the periphery. Due to technological lock-in effects, this profit gap persists over time although firms have the option to move between the cluster and the periphery.

So far the literature on technological spillovers and specialization has focused on the question how the intensity of spillovers is influenced by the degree of specialization. Our findings suggest that there might be an opposite effect such that spillovers yield technological specialization, and that under consideration of market competition, such an effect provides negative incentives for joining a cluster. We hope that future research will generalize our analysis to more general industry and demand settings.

REFERENCES

Asheim, B., & Gertler, M. (2005). The geography of innovation: Regional innovation systems. In J. Fagerberg, D. C. Mowery, & R .R. Nelson (Eds.), *The Oxford handbook of innovation* (pp. 291-317). Oxford; New York: Oxford University Press.

Audretsch, D. B. (1998). Agglomeration and the location of innovative activity. *Oxford Review of Economic Policy, 14*(2), 18-29.

Cantner, U., & Graf, H. (2004). Cooperation and specialization in German technology regions. *Journal of Evolutionary Economics, 14*(5), 543-562.

Dawid, H., & Reimann, M. (2004). Evaluating market attractiveness: Individual incentives vs. industrial profitability. *Computational Economics, 24*(4), 321-355.

Dawid, H., & Reimann, M. (2005). *Diversification: A road to inefficiency in product innovations?* Working Paper, University of Bielefeld, Germany.

Dosi, G. (1988). Sources, procedures, and microeconomic effects of innovation. *Journal of Economic Literature, 26*(3), 1120-1171.

Feldman, M. P., & Audretsch, D. B. (1999). Innovation in cities: Science-based diversity, specialization and localized competition. *European Economic Review, 43*(2), 409-429.

Glaeser, E. L., Kallal, H. D., Scheinkman, J. A., & Shleifer, A. (1992). Growth in cities. *Journal of Political Economy, 100*(6), 1126-1152.

Henderson, V., Kuncoro, A., & Turner, M. (1995). Industrial development in cities. *Journal of Political Economy, 103*(5), 1067-1090.

Jacobs, J. (1969). *The economy of cities.* New York: Random House.

Jaffe, A. B., Trajtenberg, M., & Henderson, R. (1993). Geographic localization of knowledge spillovers as evidenced by patent citations. *Quarterly Journal of Economics, 108*(3), 577-598.

Jonard, N., & Yildizoglu, M. (1998). Technological diversity in an evolutionary industry model with localized learning and network externalities. *Structural Change and Economic Dynamics, 9*(1), 33-51.

Krugman, P. (1991). *Geography and trade.* Cambridge, MA: MIT Press.

Marshall, A. (1920). *Principles of economics.* London: Macmillan.

Nelson, R. R., & Winter, S. G. (1982). *An evolutionary theory of economic change*. Cambridge, MA: Harvard University Press.

Porter, M. E. (Ed.) (1998). Clusters and competition: New agendas for companies, governments, and institutions. In *On competition* (pp. 197-287). Boston: Harvard Business School Press.

van der Panne, G. (2004). Agglomeration externalities: Marshall versus Jacobs. *Journal of Evolutionary Economics, 14*(5), 593-604.

Wersching, K. (2005). *Innovation and knowledge spillover with geographical and technological distance in an agent-based simulation model.* Discussion Paper No. 535, Bielefeld University, Germany. Retrieved from http://bieson.ub.uni-bielefeld.de/volltexte/2005/674/

ENDNOTES

1. We abstract from exit and entry of firms on the industry level, and rather focus on exit and entry at the level of sub-markets given by certain technology ranges.
2. To be more precise the firm randomly picks a 'candidate' location in the technology space. However, before actually entering this market, the firm checks whether the market evaluation exceeds a certain minimal threshold.
3. A Wilcoxon test comparing the values of each specialization index after 100 periods under random and profit-oriented strategies confirms these claims with p-values less than 0.01 for locational specialization and knowledge specialization.
4. The p-value of the corresponding Wilcoxon test after 100 periods for knowledge specialization is less than 0.01. The statistics for locational specialization do not support the thesis for period 100, because as shown in the figure, the two graphs overlap in the last periods of the simulation. But locational specialization is significantly lower for $40 \leq t \leq 90$.
5. Wilcoxon tests support the result of this ranking with p-values less than 0.01 for data in period 100.
6. Dawid and Reimann (2004) provide an extensive analysis of this effect in a similar industry setting.

Chapter XXVI
Simulating Product Invention Using InventSim

Anthony Brabazon
University College Dublin, Ireland

Arlindo Silva
Instituto Politecnico de Castelo Branco, Portugal

Tiago Ferra de Sousa
Instituto Politecnico de Castelo Branco, Portugal

Robin Matthews
Kingston University, UK

Michael O'Neill
University College Dublin, Ireland

Ernesto Costa
Centro de Informatica e Sistemas da Universidade de Coimbra, Portugal

ABSTRACT

This chapter describes a novel simulation model (InventSim) of the process of product invention. Invention is conceptualized as a process of directed search on a landscape of product design possibilities, by a population of profit-seeking inventors. The simulator embeds a number of real-world search heuristics of inventors, including anchoring, election, thought experiments, fitness sharing, imitation, and trial and error. A series of simulation experiments are undertaken to examine the sensitivity of the populational rate of advance in product sophistication to changes in the choice of search heuristics employed by inventors. The key finding of the experiments is that if search heuristics are confined to those that are rooted in past experience, or to heuristics that merely generate variety, limited product advance occurs. Notable advance occurs only when inventors' expectations of the relative payoffs for potential product inventions are incorporated into the model of invention. The results demonstrate the importance of human direction and expectations in invention. They also support the importance of formal product/project evaluation procedures in organizations, and the importance of market information when inventing new products.

INTRODUCTION

The importance of invention and innovation as an engine for economic growth, and in shaping market structure, has long been recognized (Schumpeter, 1934, 1943; Nelson & Winter, 1982; Abernathy & Clark, 1985; Maskus & McDaniel, 1999). The invention of new products can enhance the efficiency with which inputs can be converted into outputs (for example, the invention of more efficient production equipment) or can lead to the qualitative transformation of the structure of the economy by creating completely new products (Freeman & Soete, 1997).

Given the economic and social importance of the development of new products, questions of interest naturally arise concerning the dynamics of the process of invention; these form the research domain of this chapter. This domain is distinguished from the study of the commercial implications of inventions once they are created. Fleming and Sorenson (2001) note that while the processes of commercial diffusion of new goods have attracted substantial study, "we lack a systematic and empirically validated theory of invention" (p. 1019).

The lack of a theory of invention leaves open the question: How do inventors actually invent? Given that no inventor can try all possible combinations of even the set of already discovered raw components when attempting to invent a novel product, two further questions arise: What methods do inventors employ to simplify their task? and What are the implications of these methods for the rate of inventive progress?

These apparently simple questions are highly significant. Without a robust theory of invention, managers' ability to create organizations that encourage inventive practices is constrained, and policymakers risk making suboptimal decisions regarding how best to encourage invention in society in order to promote long-term economic growth. This chapter focuses attention on the role of search heuristics in the decision-making processes of inventors.

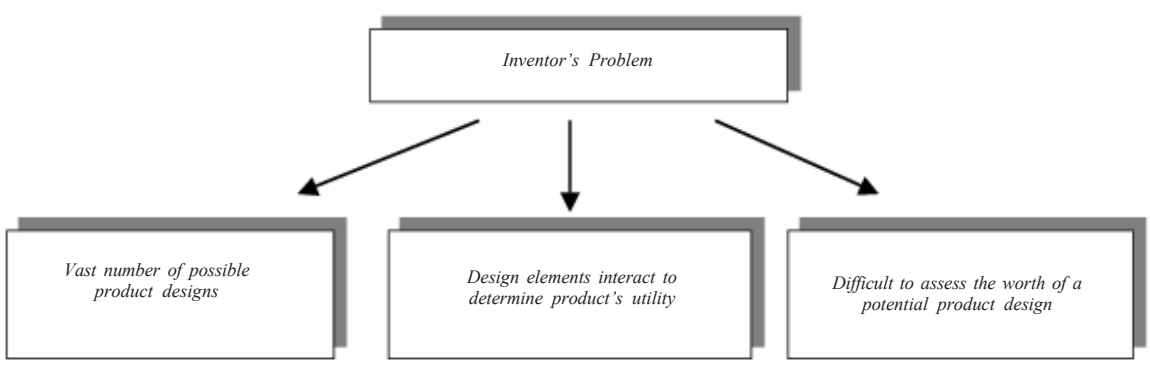
The Inventor's Problem

In many real-world decision scenarios, including product invention, inventors are faced with three key problems, a large decision space (many variables), interconnectedness of solution elements (a complex mapping between individual elements of a solution and the payoff to the whole solution), and difficulties in assessing the likely payoffs to inventive ideas *ex-ante* their physical implementation (see Figure 1).

In the case of product invention, a vast number of possible product designs exist, and inventors must decide which of these possibilities to pursue. Their decision as to where to concentrate their inventive efforts is driven by the *search heuristics* they use, and their expectations as to the profitability of potential products. These expectations are subject to error for two reasons, technical uncertainty as to how a large system of components (a product) will behave, and commercial uncertainty as to what return a product will earn if it is successfully created.

A search heuristic is defined as: *a method for searching for an acceptable solution to a problem without considering all possible solution alternatives.* This definition encompasses a broad range of structures, for example an organizational structure can be considered as a search heuristic (Cohen, 1981). Search heuristics are widely used in everyday decision making, either because of the impossibility of determining and evaluating all possible solutions to a problem, or because the benefits from obtaining the best, rather than a good solution to a problem are outweighed by the extra costs of obtaining the optimal solution. Search heuris-

Figure 1. The inventor's problem



tics guide inventors both in their choice of which product ideas to pursue and how to generate product novelty. These heuristics restrict the size of the space of alternatives considered, and place a strategy on the method of search (Matthews, 2002). Examples of the former include restricting attention to a subset of available components and the use of problem decomposition. Examples of the latter include the strategy of only pursuing a proto-product design which appears to offer better profit potential than your current product design. This study does not consider all possible search heuristics that inventors could adopt, but concentrates on a subset of heuristics that have attracted attention in prior literature.

Relevant Literature

The literature on invention and learning is fragmented and spread across several disciplines, each with its own focus of interest, level of aggregation. and perspective. Relevant literatures include those listed in Table 1.

Nooteboom (2000) comments that it is impossible for any scholar to embrace all of these strands of literature. Inevitably, work on innovation or invention must draw most strongly from one or a small subset of the above literatures. This chapter draws its primary inspiration from the literature of complex adaptive systems. It is also noted that a substantial literature on evolutionary learning exists in computer science (Holland, 1992; Fogel, 2000; Koza, 1992). This chapter also draws from this literature.

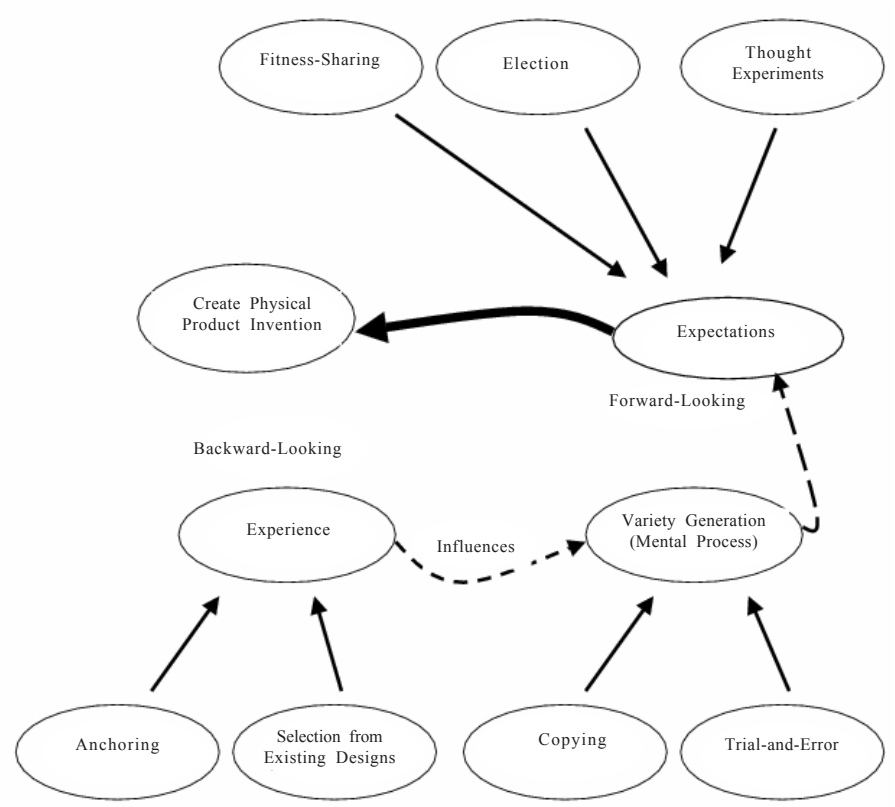
Organization of the Chapter

The rest of this chapter is organized as follows. The next section provides an overview of the conceptual model underlying the simulator, fol-

Table 1. Literatures that cover aspects of innovation and learning

Studies of organizational learning	Literature on innovation systems
Populational ecology in sociology	Literature on institutional economics
Evolutionary and neo-Schumpeterian economics	Marketing
Literature on complex adaptive systems	Literature on cognitive science
Literature on networks	

Figure 2. Conceptual model of product invention



lowed by a section which introduces the simulator. We then describe the results of a series of simulation experiments. The final section concludes the study and makes a number of suggestions for future work.

CONCEPTUAL MODEL OF INVENTION

The conceptual model of invention underlying the simulation experiments is outlined in Figure 2. The model is embedded in a general evolutionary process, but this is adapted for the salient characteristics of the process of product invention.

The conceptual model embeds the search heuristics of anchoring, copying from existing designs, trial-and-error experimentation, thought experiments, election, and fitness sharing. In the model, inventors are conceptualized as starting from their existing product design in each inventive trial (an anchoring heuristic). Their generation of novel proto-product ideas (these are defined as mental product design ideas in the heads of inventors) arises from a combination of copying (imitation) elements of existing product designs, and incremental trial-and-error invention in an effort to improve their existing product design. In each inventive trial, an inventor generates multiple proto-product ideas (a thought experiment heuristic). The expected

payoff to the best of these thought experiments is compared with the expected payoff to the inventor's current product, and if higher, the proto-product is physically made and replaces the inventor's current idea (an election heuristic). All the expected payoffs are discounted by considering the expected degree of competition that the design will face (a fitness-sharing heuristic). Each of these heuristics are discussed in more detail in the following subsections.

Anchoring

Product invention is strongly influenced by, and anchored within, the population of currently existing product designs. Heavy reliance is placed on historical experience by inventors, and the results of past searches become natural starting points for new ones (Nelson & Winter, 1982; Stuart & Podolny, 1996). From a technical standpoint, as populations of engineers and designers build up experience and absorption capacity (Cohen, 1981) with current product architectures and design elements, they will tend to draw on this experience when faced with future design decisions, rather than re-examining all possible alternatives *ab initio* in each inventive trial. Existing products also embed past learning concerning knowledge of customer needs. Hence, it is plausible to assume that inventors employ an anchoring search heuristic, and start their inventive activity from current product designs.

Election Heuristic

Profit-seeking inventors will not discard their current product design until they uncover one that they expect will produce even higher returns. This represents a search heuristic: *do not give up a good idea until a better one comes along*. The economic interpretation of this heuristic is that the inventor carries out a mental simulation (Birchenhall, 1995; Kennedy, Eberhart, & Shi, 2001). If the expected return from the new product design appears unattractive, the bad idea is discarded, and the inventor stays with his current design. In the election step, inventors compare the expected return from the proposed proto-product design with that of a current product design, and if it is less, the proto-product idea is discarded and is not physically created. Examples of election mechanisms abound in business, ranging from formal project appraisal systems to procedures for monitoring the performance of ongoing product development projects.

Thought Experiments

Inventors do not typically consider a single product-design idea in each inventive trial before they attempt to physically create a new product (Otto & Wood, 2001). Thought experiments represent the heuristic: 'generate several mental product ideas, and pick the best of these'. These mental simulations can include the construction of computer simulations and mock-ups (Nooteboom, 2000). Thought experiments can be considered as corresponding to the *openness* of an inventor to new ideas. The greater the number of mental thought experiments or product design ideas that inventors consider when creating new products, the more open they are considered to be to new ideas. Inventors, or in a corporate setting R&D departments, which generate small numbers of product design alternatives are less likely to prosper.

Fitness Sharing

A key factor impacting the return to a new product is the degree of competition it faces from similar existing products. If there are several very similar products in the market-

place, they compete for the same customer segment, and the returns to each product are likely to be lower than they would be in the absence of competition. In the model of invention, it is assumed that inventors employ a heuristic of 'take account of expected competition' (fitness sharing) when forming their expectations as to the likely payoff to a product idea. The fitness-sharing mechanism was based on that of Mahfoud (1995) and is defined as follows:

$$f'(i) = \frac{f(i)}{\sum_{j=1}^{n} s(d(i,j))}$$

where *f(i)* represents the original raw fitness of product design *i* (the fitness of the design is calculated using the NK landscape—see next section) which exists in the marketplace. If this design suffers competition from other very similar products that are active in the market, its raw fitness is reduced from the fitness it would have if it had no competition. The shared (reduced) fitness of design *i* is denoted as *f'(i)* and corresponds to its original raw fitness (*f(i)*), derated or reduced by an amount determined by a sharing function ***s***.

The (sharing) function ***s*** provides a measure of the density of active product designs within a given neighborhood of design *i* on the landscape. For any pair of designs *(i,j)*, the sharing function returns a value of '0' if the two designs *(i,j)* are more than a specified distance (*t*) apart, and therefore are not considered to be competing for the same market niche, a value of '1' if the designs are identical, and a scaled value between 0 and 1 otherwise. The form of the sharing function adopted in this study was taken from Mahfoud (1995), where

$$s(d) = 1 - \left(\frac{d}{t}\right)^{\alpha} \text{ if } d<t,$$

otherwise $s(d)=0$, where t is the neighborhood within which designs are considered to compete, d is the actual distance between two designs, and α is a scaling constant. Fitness sharing impacts on the expectations of inventors as to the likely fitness of proto-designs and acts to discourage inventors from closely imitating products already subject to significant competition. In the simulation experiments, all payoffs are assessed by inventors using a shared fitness heuristic. Therefore, election decisions and selection decisions are based on shared rather than raw fitness values.

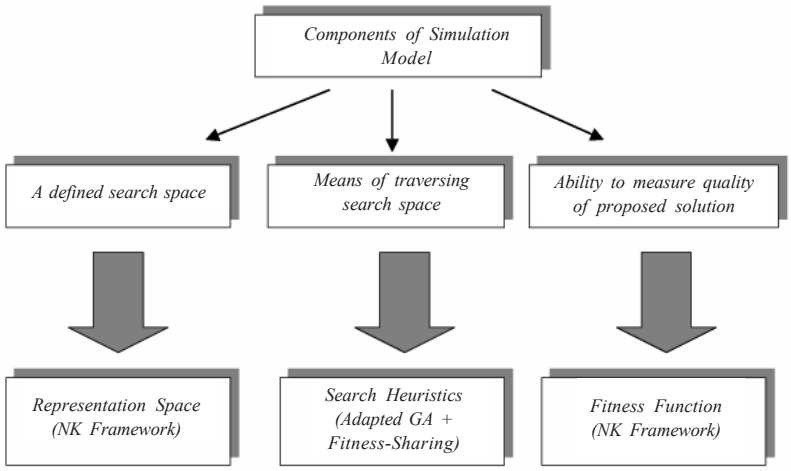
SIMULATION MODEL

The conceptual model is operationalized in the *InventSim* simulator (Brabazon, 2005; Brabazon et al., 2005) using a novel synthesis of two general frameworks from the literature of complex adaptive systems, Kauffman's NK Framework (Kauffman & Levin, 1987; Kauffman 1993, 1995) and the Genetic Algorithm (GA) (Holland, 1992). A particular challenge in seeking to gain insight into the process of invention is that we have limited empirical data concerning invention processes. Unsuccessful inventions leave few footprints on the sands of time, and patent databases form an incomplete record of inventive effort. Hence, it is difficult to examine the role of search heuristics in the process of invention by focusing solely on empirical data. In this chapter, a simulation-based approach is adopted. This allows the creation of a 'closed world' and therefore permits the examination of the role of specific search heuristics in the process of invention.

The components of the simulation model are illustrated in Figure 3.

In order to investigate the utility of differing search heuristics in the process of product invention using a simulation methodology, a

Figure 3. Components of simulation model



representation or product design space must be defined within which inventors are assumed to search. In this study, the NK framework is used to define a product design space. The NK framework, drawn from the literature of complex adaptive systems, represents a general model of systems that comprise interconnected elements. These systems are widespread and include networks of genes, networks of activities within an organization, and networks of components in a physical product. It is noted *ab initio* that application of the NK framework in a simulation model is not atypical, and it has already been applied in this manner in a variety of studies in organizational science (Levinthal, 1997; Levinthal & Warglien, 1999; Rivkin, 2000; Gavetti & Levinthal, 2000) and technological invention (Frenken, 2001; Fleming & Sorenson, 2001).

The GA, drawing inspiration from the process of biological evolution, utilizes a pseudo neo-Darwinian metaphor to simulate the evolution of a population of entities over time. GAs have been applied in two primary areas of research (Mitchell, 1996), combinatorial optimization in which GAs represent a population-based optimization algorithm, and the study of adaptation in complex systems. *Prima facie,* the framework of the GA has the potential to incorporate several salient aspects of the invention process. These are:

- a population of entities (products) which adapt over time,
- competition for resources among inventors (profit-driven motivation),
- a selection process,
- reuse of previously invented components, and
- trial-and-error experimentation.

Simon (1996) notes that the GA can be used to build a simulation model of an adaptive system and that these simulations "can be used to study the relative rates at which fitness will grow under different assumptions about the model, including assumptions about rates of mutation [trial-and-error] and crossover [recombination]" (p. 180). In the simulator, the canonical GA is adapted to embed the heuristics of inventors, as already discussed in the conceptual framework.

Product Design Landscape

In this study the product design landscape is defined using Kauffman's NK model (Kauffman, 1993, 1995). The NK model considers the behavior of systems that comprise a configuration (string) of *N* individual elements. Each of these elements are in turn interconnected to *K* other of the *N* elements (*K*<*N*). In a general description of such systems, each of the *N* elements can assume a finite number of states. In Kauffman's operationalization of this general framework, the number of states for each element is restricted to two (0 or 1). Therefore the configuration of *N* elements can be represented as a binary string. The parameter *K* determines the degree of fitness interconnectedness of each of the *N* elements and can vary in value from 0 to *N-1*. In one limiting case where *K*=0, the contribution of each of the *N* elements to the overall fitness value (or worth) of the configuration is independent of each other. As *K* increases, this mapping becomes more complex, until at the upper limit when *K*=*N-1*, the fitness contribution of any of the *N* elements depends both on its own state and the simultaneous states of all the other *N-1* elements, describing a fully connected graph.

If we let s_i represent the state of an individual element *i*, the contribution of this element to the overall fitness (**F**) of the entire configuration is given by $f_i(s_i)$ when *K*=0. When *K*>0, the contribution of an individual element to overall fitness depends both on its state and the states of the *K* other elements to which it is linked ($f_i(s_i)$: s_{i1}, ..., s_{ik}). A random fitness function (**U**(0,1)) is adopted, and the overall fitness of each configuration is calculated as the average of the fitness values of each of its individual elements. Therefore, if the fitness values of the individual elements are f_1, ..., f_N, the overall fitness of the entire configuration (**F**) is:

$$F = \left[\frac{\sum_{i=1}^{N} f_i}{N} \right]$$

Therefore, for the case where K=0, and where N=3 (for example), the fitness of a specific configuration (0,0,1) is simply the average of the fitness values for each individual locus on the string. In turn, the fitness value of a 0 or a 1 in each locus is modeled as a random draw from U(0,1). In the more complex case where K=2, the fitness value of (say) a 0 in the first locus of the configuration (0,0,1) depends not just on its own state, but also on the simultaneous states of the following two loci to which it is fitness connected. Therefore, the fitness of the zero in the first locus of (0,0,1) is not the same as that of (0,0,0). The fitness value of 0 in each case is determined by the simultaneous states of 2^2 (2^K) following loci, and the value of each of these states is determined by a random draw from U(0,1).

Altering the value of *K* affects the ruggedness of the described landscape (graph), and consequently impacts on the difficulty of search on this landscape (Kauffman & Levin, 1987; Kauffman 1993). As *K* increases, the landscape becomes more rugged, and the best peaks on the landscape become higher, but harder to find.

Mapping Product Designs into the NK Framework

NK landscapes are concerned with systems that consist of multiple, interacting elements. In a similar fashion, products consist of systems of multiple, interacting design elements, and the utility or worth of products depends on both the nature of the included elements and their interconnection structure. The importance accorded to element interconnection in determining the

fitness of a configuration in the NK model finds plausible parallels with real-world product design. The presence of a specific design element places constraints on other design elements. For example, the use of a particular lightweight material to cover an airplane wing may require that an additional internal brace be added to increase wing rigidity. The importance of design element interdependence in product design is noted by several authors, including Baldwin and Clark (2000), who comment that design elements depend on each other, "often in very complex and convoluted ways" (p. 36), and Ulrich and Eppinger (2000) who point out that components and subassemblies can interact with one another in both "planned and unintended ways" (p. 195).

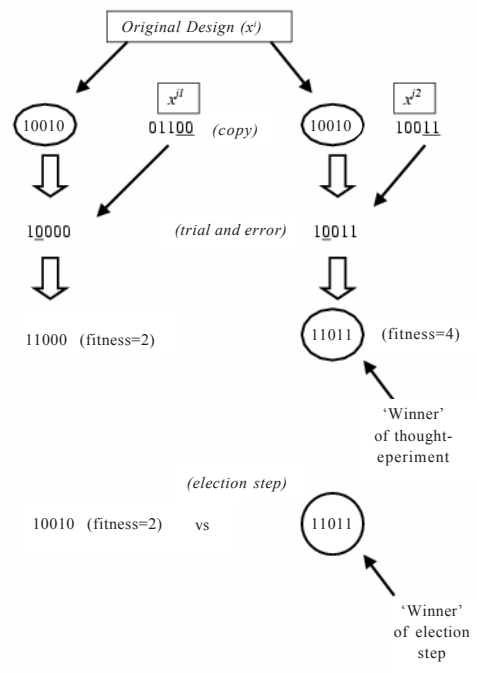
An extreme example could arise where a small change in a single design attribute destroys the value of the entire system. Consider the effect of increasing the voltage output from a transformer in a computer. The increased voltage could destroy several other components, reducing the value of the entire system to zero. The design of pharmaceutical products is another example of a high-K design, as a small change in the formulation of the product can impact dramatically on its utility.

Physical product designs are characterized in this study as consisting of N attributes. Each of these attributes represents a choice of design attribute that an inventor faces. Hence, a specific design configuration **s** is represented as a vector $s_1, \ldots, s_N$, where each attribute can assume a value of 0 or 1. The vector of attributes represents an entire product design, hence it embeds a choice of physical components, ancillary choices concerning these components (color, finish), the choice of configuration of the components (their tolerances, directional orientation, physical linkage structure), and the choice of production technologies required to manufacture the product design (Kauffman, Lobo, & MacReady, 1998). Good,

Example 1.

```
Repeat 'A' times Create Product Landscape
    Repeat for each string (active product design) in the population
        Take string 'i'
        Calculate fitness values for each string in the population
        For x=1:a ('a' thought experiments)
                Select another design 'j' in the population
                Recombine design 'i' and 'j' to produce new design 'k' (simulates
                imitation process)
                Apply mutation operator to new design 'k' (simulates trial-and-error
                process)
                    If design 'k' is best design of thought experiments so far, store
                    design 'k' in design 'best'
        End (for loop)
        If design 'best' is better than the original design 'i', replace design 'i' with
        design 'best' (election operator)
    End (Repeat for each string loop) (end of generation)
Output results for simulation run End (Repeat 'A' loop)
```

Figure 4. Example of model, with two thought experiments and election; the first thought experiment is shown on the left side of the diagram, the second on the right



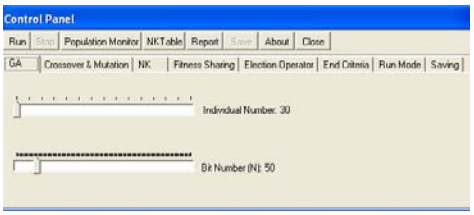
consistent sets of product design correspond to peaks on the product design landscape.

In the simulator, the product design landscape, whereon a population of inventors are searching for ever better designs, is defined using a binary representation. The adaptive efforts of inventors are governed by variety-generating heuristics, including imitation, and trial and error. Experience-based heuristics influence the search process by means of anchoring and selection mechanisms, and expectations-based heuristics influence the search process through thought experiments and election mechanisms.

Pseudo-Code for Simulator

The pseudo-code for the simulator is given in Example 1.

Figure 5. Control screen for selecting the number of individual designs in the simulation, and the length of the binary strings which represent each product design (the choice of N for the NK landscape)



Worked Example of Simulation Model

Figure 4 provides a worked example of a single inventive trial in a simulation experiment. In the diagram, x^i is the binary string '10010'. It is assumed that the inventor undertakes two thought experiments. In the first thought experiment when x^{j1} is selected from the population of active designs, binary string '01100' is chosen, and in the second thought experiment, binary string '10011' is chosen. Assume that in each thought experiment, the copying operator copies the last two bits of x^j into x^i. Also assume that in each thought experiment, the incremental trial-and-error operator flips the second bit of each proto-design after the copying step has taken place. Comparison between the fitness

Figure 6. Control screen for selection of parameters governing copying, imitation, and thought experiment heuristics

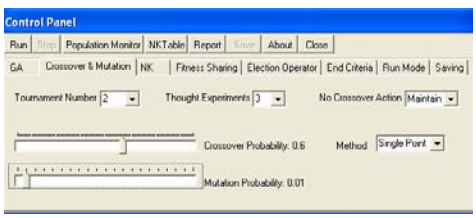


Figure 7. Example of population display option

(assumed on grounds of simplicity in this example to be the number of '1's in a binary string) of the two proto-designs, after both the copying and the incremental trial-and-error steps, shows that the second thought experiment has produced the better design, with a fitness of 4. The final step is the election process, where the fitness of the original design (x^i) is compared with the fitness of the design that resulted from the second thought process. The original design had a fitness of 2, hence it is replaced by design '11011'. This new design then becomes x^i for time period t+1.

Simulator Screen Shots

Although the underlying code for the *InventSim* simulator is written in C++, the user interacts with the simulator through a series of easy-to-use screens (Figures 5 and 6 provide an illustration of two of these screens). These screens allow the user to select and alter a wide variety of parameters that determine the nature of the simulation run. In essence, the simulator allows the user to select choices for four items, namely:

- the form of NK landscape generated,
- the nature of the search heuristics to be employed by inventors,
- the number of simulations to be run, and
- the form of output generated during the simulation run.

During the simulation run, a graphic of the status of the population during the simulation run (see Figure 7) and a run report (see Figure 8) can be displayed. The report display records the full list of simulation parameters chosen by the modeler, as well as providing a running

Figure 8. Sample report produced by simulator

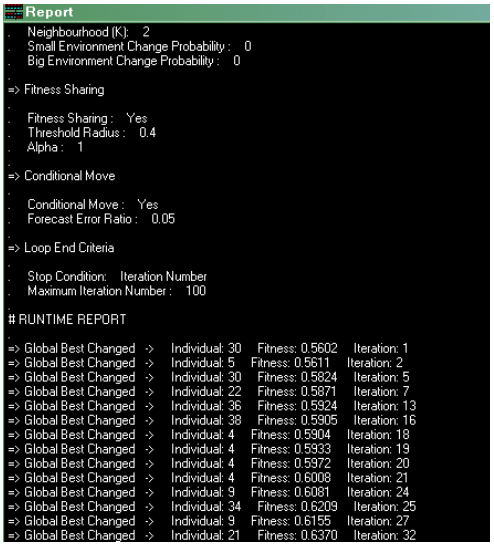


Figure 9. Imitation, trial and error without election vs. imitation, trial and error, and election for three levels of thought experiment (1,3, 5) for K= 4

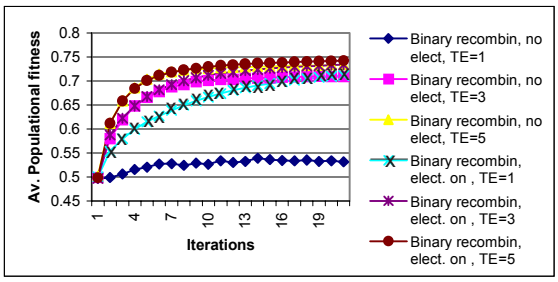


Figure 10. Imitation and trial and error without election vs. imitation, trial and error, and election for three levels of thought experiment (1,3, 5) for K= 6

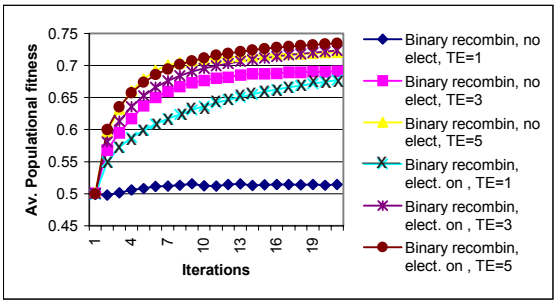


Table 2. Average populational fitness, imitation, trial and error (no election) vs. imitation, trial and error + election for K=4 landscape, and number of thought experiments=1,3,5

Iterations	Imitation and trial and error (no election)			Imitation and trial and error (with election)		
	TE=1	*TE=3*	*TE=5*	*TE=1*	*TE=3*	*TE=5*
1	0.4971	0.4996	0.5014	0.5003	0.4971	0.4983
10	0.4990	0.5803	0.6121	0.5535	0.5873	0.6115
50	0.5270	0.6781	0.7139	0.6272	0.6815	0.7114
100	0.5339	0.7031	0.7280	0.6738	0.7158	0.7319
150	0.5343	0.7082	0.7311	0.6985	0.7277	0.7386
200	0.5317	0.7095	0.7334	0.7120	0.7339	0.7423

Table 3. Average populational fitness, imitation, trial and error (no election) vs. imitation, trial and error + election for K=6 landscape, and number of thought experiments=1,3,5

Iterations	Imitation and trial and error (no election)			Imitation and trial and error (with election)		
	TE=1	*TE=3*	*TE=5*	*TE=1*	*TE=3*	*TE=5*
1	0.4990	0.5009	0.5017	0.4990	0.5003	0.4993
10	0.4980	0.5680	0.5964	0.5497	0.5808	0.6000
50	0.5112	0.6503	0.6918	0.6088	0.6662	0.6853
100	0.5120	0.6801	0.7117	0.6416	0.6995	0.7158
150	0.5140	0.6876	0.7173	0.6626	0.7139	0.7279
200	0.5146	0.6913	0.7205	0.6770	0.7233	0.7342

record of the best design in the population at the end of each iteration. The simulator also facilitates the recording of comprehensive run-data to disk during the simulation.

RESULTS

All results are averaged across 30 separate simulation runs. In each simulation run, the NK landscape is specified anew, and the positions of the initial product designs are randomly selected at the start of each run. A value of N=96, and K values of 4 and 6 are selected in defining the landscapes in the simulation experiments. An imitation heuristic (wherein an inventor can imitate a portion of another design string) is applied with a probability of 0.60 in each inventive trial, and the trial-and-error rate is selected to produce an expected mutation rate of one bit in each product design string in each inventive trial (therefore simulating an incremental trial-and-error invention heuristic).

These simulations compare the rate of in-

ventive advance of a population of inventors using a trial-and-error plus imitation heuristic, when an election heuristic is (vs. is not) employed. The results of these simulations are presented in Figures 9 and 10, where the results are reported after the first iteration and thereafter after each tenth iteration up to 200 iterations, and in Tables 2 and 3.

Looking at the results from the simulations where inventors limit themselves to a single thought experiment, and engage in both trial-and-error experimentation and imitation from existing designs, it is noted that the inclusion of election notably increases the rate of inventive advance. The populational average fitnesses is higher when an election heuristic is used by the 200 iteration end-point of the simulations (the difference is significant at the 5% level).

It can also be observed that when the number of thought experiments is raised from 1 to 3 or 5, the rate of inventive advance increases. Again, the difference in populational average fitnesses is significant (at the 5% level) by the 200 iteration end-point of the simulations when multiple thought experiments are undertaken.

In assessing the results from the differing levels of thought experiments, it is important to remember that the thought experiments mechanism does not require that inventors can make perfect assessments of the payoffs to several potential product designs *ex-ante* their testing in the marketplace. Rather it only requires that inventors can assess the relative payoffs of the designs. A similar comment can be made in respect of the election heuristic in that inventors only need to be able to identify whether the new product design is better than their existing design. It is not necessary that inventors are able to precisely assess the worth of the new design. Once inventors can make reasonably accurate assessments of the relative payoffs of proto-product designs, product fitness advance is assured when inventors use an election mechanism, or when they engage in multiple thought experiments. Election and thought experiments, in other words inventors' expectations, matter. Therefore, the simulation results support the assertion that inventors (managers) should undertake a formal assessment of the worth of a proto-product design before undertaking the inventive step of actually creating the product.

Each thought experiment requires that an inventor consider what elements could be imitated from other products, and what elements of the resulting proto-product design should be incrementally altered by trial and error. Implicitly, the selection process when deciding what to imitate requires that inventors consider what other products already exist, and also the worth of these products. Therefore, the undertaking of multiple thought experiments requires a quality conduit of market information. The better the flow of market information to the inventor, the easier it is to generate multiple proto-product designs and the more accurate the inventor's estimate of their likely fitness. Inventors or organizations with good communication channels between the market and inventors (and in organizations, between members of the design team) will find it easier to undertake multiple thought experiments. Therefore the simulation results are consistent with a proposition that invention produces better results when it is well informed by market information.

CONCLUSION

This chapter developed a novel conceptual model of the process of product invention by considering product invention as an adaptive process that is governed by the search heuristics of inventors. The model was operationalized in a simulation study, in order to examine the utility of a variety of search heuristics, by coherently integrating the NK and GA frame-

works drawn from the literature of complex adaptive systems. The developed simulation model was used to address two broad questions:

- What effect do inventor's search heuristics have on the rate of product invention?
- Does the degree of product-design interconnectedness alter the effectiveness of inventor's search heuristics?

The results suggest that despite the posited importance of the basic variety-generating heuristics of trial and error and imitation, they are not sufficient in themselves to produce substantial progress in product invention. The inventor's expectations and consequent direction of the inventive process play a critical role in ensuring inventive advance.

It is not possible in a single set of simulation experiments to exhaustively examine every possible combination of settings for each parameter in the simulation model. However, the initial results cast an interesting light on the process of product invention, and the development of the *InventSim* simulator extends the methodologies available to researchers to conceptualize and examine product invention. Future work will extend this study by considering additional parameter settings and additional forms of imitation operator, and by incorporating noisy expectations as to proto-product design payoffs.

While this chapter concentrated on the study of product invention, it is noted that the issues that make product invention difficult are also present in important decisions that managers face. For example, in both strategic management and marketing management, managers must select a strategy from a vast array of possible strategies, where elements of these strategies are fitness interconnected, and where the worth of potential strategies must be estimated ex-ante their implementation. These problems are difficult for exactly the same reasons that product invention is. Just as inventors must turn to search heuristics, so too must managers. Therefore, it is believed that the conceptual framework developed in this chapter has potential for more general application outside the domain of product invention.

REFERENCES

Abernathy, W., & Clark, K. (1985). Innovation: Mapping the winds of creative destruction. *Research Policy, 14*(1), 3-22.

Birchenhall, C. (1995). Modular technical change and genetic algorithms. *Computational Economics, 8*(3), 233-253.

Brabazon, A. (2005). *Product invention as a complex adaptive system: An investigation of the impact of inventors' search heuristics on the rate of invention.* Doctoral Thesis, Kingston University, UK.

Brabazon, A., Silva, A., Ferra de Sousa, T., O'Neill, M., Matthews, R., & Costa, R. (2005, September 2-5). A memetic model of product invention. *Proceedings of the Congress on Evolutionary Computation* (CEC 2005), Edinburgh. NJ: IEEE Press.

Cohen, W. (1981). The power of parallel thinking. *Journal of Economic Behavior and Organization, 2*(4), 285-306.

Fleming, L., & Sorenson, O. (2001). Technology as a complex adaptive system: Evidence from patent data. *Research Policy, 30*(7), 1019-1039.

Fogel, D. (2000). *Evolutionary computation: Towards a new philosophy of machine intelligence.* New York: IEEE Press.

Freeman, C., & Soete, L. (1997). *The economics of industrial innovation* (3rd ed.). Cambridge, MA: MIT Press.

Frenken, K. (2001). *Understanding product innovation using complex systems theory.* Unpublished PhD thesis, University of Amsterdam, The Netherlands, and Universite Pierre Mendes, France.

Gavetti, G., & Levinthal, D. (2000). Looking forward and looking backward: Cognitive and experiential search. *Administrative Science Quarterly, 45*(1), 113-137.

Holland, J. (1992). *Adaptation in natural and artificial systems*, (originally published in 1975). Ann Arbor: University of Michigan Press.

Kauffman, S., & Levin, S. (1987). Towards a general theory of adaptive walks on rugged landscapes. *Journal of Theoretical Biology, 128,* 11-45.

Kauffman, S. (1993). *The origins of order.* Oxford, UK: Oxford University Press.

Kauffman, S. (1995). *At home in the universe.* Oxford, UK: Oxford University Press.

Kauffman, S., Lobo, J., & MacReady, W. (1998). *Optimal search on a technology landscape.* Santa Fe Institute Working Paper 98-10-091, USA.

Kennedy, J., Eberhart, R., & Shi, Y. (2001). *Swarm intelligence.* San Mateo, CA: Morgan Kauffman.

Koza, J. (1992). *Genetic programming: On the programming of computers by means of natural selection.* Cambridge, MA: MIT Press.

Levinthal, D. (1997). Adaptation on rugged landscapes. *Management Science, 43*(7). 934-950.

Levinthal, D., & Warglien, M. (1999). Landscape design: Designing for local action in complex worlds. *Organization Science, 10*(3), 342-357.

Mahfoud, S. (1995). Population size and genetic drift in fitness sharing. In L. Whitley & M. Vose (Eds.), *Foundations of genetic algorithms 3* (pp. 185-223). San Francisco: Morgan Kauffmann.

Maskus, K., & McDaniel, C. (1999). Impacts of the Japanese patent system on productivity growth. *Japan and the World Economy, 11*(4), 557-574.

Matthews, R. (2002, September 9-11). Strategy, complex adaptive systems and games: An evolutionary approach. *Proceedings of British Academy of Management Annual Conference 2002,* London.

Mitchell, M. (1996). *An introduction to genetic algorithms.* Cambridge, MA: MIT Press.

Nelson, R., & Winter, S. (1982). *An evolutionary theory of economic change.* Cambridge, MA: Harvard University Press.

Nooteboom, B. (2000). *Learning and innovation in organizations and economies.* Oxford: Oxford University Press.

Otto, K., & Wood, K. (2001). *Product design: Techniques in reverse engineering and new product development.* Upper Saddle River, NJ: Prentice-Hall.

Rivkin, J. (2000). Imitation of complex strategies. *Management Science, 46*(6), 824-844.

Schumpeter, J. (1934). *The theory of economic development* (8th printing, 1968). Boston: Harvard University Press (Harvard Economic Studies XLVI).

Schumpeter, J. (1943). *Capitalism, socialism and democracy* (reprinted 1992). London: Routledge.

Simon, H. (1996). *Sciences of the artificial* (3rd ed.). Cambridge, MA: MIT Press.

Stuart, T., & Podolny, J. (1996). Local search and the evolution of technological capabilities. *Strategic Management Journal, 17*(Summer Special Issue), 21-38.

Volume II

Nature-Inspired Computing and Management

Section IV

Human-Centric Evolutionary Systems in Design and Decision-Making

Chapter XXVII
Human-Centric Evolutionary Systems in Design and Decision-Making

I. C. Parmee
University of the West of England, UK

J. R. Abraham
University of the West of England, UK

A. Machwe
University of the West of England, UK

ABSTRACT

The chapter introduces the concept of user-centric evolutionary design and decision-support systems, and positions them in terms of interactive evolutionary computing. Current research results provide two examples that illustrate differing degrees of user interaction in terms of subjective criteria evaluation; the extraction, processing, and presentation of high-quality information; and the associated improvement of machine-based problem representation. The first example relates to the inclusion of subjective aesthetic criteria to complement quantitative evaluation in the conceptual design of bridge structures. The second relates to the succinct graphical presentation of complex relationships between variable and objective space, and the manner in which this can support a better understanding of a problem domain. This improved understanding can contribute to the iterative improvement of initial machine-based representations. Both examples complement and add to earlier research relating to interactive evolutionary design systems.

INTRODUCTION

Uncertainty and poor problem definition are inherent features during the early stages of design and decision-making processes. Immediate requirements for relevant information to improve understanding can be confounded by complex design representations comprising

many interacting variable parameters. Design constraints and multiple objectives that defy complete quantitative representation and therefore require a degree of subjective user evaluation further inhibit meaningful progression. Machine-based problem representation may, initially, be based upon qualitative mental models arising from experiential knowledge, group discussion, and sparse available data. However, such representations, coupled with user intuition, play a significant role in defining initial direction for further investigation. Concepts based upon current understanding require both quantitative and qualitative exploration to generate relevant information that supports and enables meaningful progress.

The chapter presents research and development relating to powerful machine-based search and exploration systems that, through appropriate user interaction, allow both quantitative and qualitative evaluation of solutions and the extraction of information from complex, poorly understood design and decision-making domains. The integration and capture of user experiential knowledge within such systems in order to stimulate, support, and increase understanding is of particular interest. The objective is the realisation of user-centric intelligent systems that overcome initial lack of understanding and associated uncertainty, support an improving knowledge-base, allow the integration of subjective judgement, and stimulate innovation and creativity.

INTERACTIVE EVOLUTIONARY COMPUTATION (IEC)

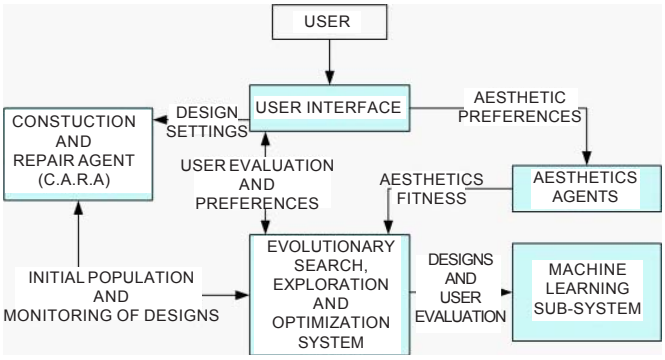
Interactive evolutionary computing (Takagi, 1996) mainly relates to partial or complete human evaluation of the fitness of solutions generated from evolutionary search. This has been introduced where quantitative evaluation is difficult if not impossible to achieve. Examples of application include graphic arts and animation (Sims, 1991), food engineering (Herdy, 1997), and hazard icon design (Carnahan, 2004). Such applications rely upon a human-centred, subjective evaluation of the fitness of a particular design, image, taste, and so forth, as opposed to an evaluation developed from some analytic model.

Partial human interaction that complements quantitative machine-based solution evaluation is also evident—for instance, the user addition of new constraints in order to generate solutions that are fully satisfactory within an evolutionary nurse scheduling system (Inoue, Furuhashi, & Fujii, 1999). Another example is the introduction of new compounds as elite solutions into selected evolving generations of a biomolecular design process (Levine, Facello, & Hallstrom, 1997).

These examples utilise a major advantage of stochastic population-based search techniques—that is, their capabilities as powerful search and exploration algorithms that provide diverse, interesting, and potentially competitive solutions to a wide range of problems. Such solutions can provide information to the user which supports a better understanding of the problem domain whilst helping to identify best direction for future investigation (Parmee & Bonham, 1999), especially when operating within poorly defined decision-making environments. Extracted information supports development of the problem representation in an iterative, interactive evolutionary environment. Interactive evolutionary design systems (IEDSs) represent a human-centric approach (Parmee, 2002; Parmee, Watson, Cvetkovic, & Bonham, 2000) that generate and succinctly present information appertaining to complex relationships between the variables, objectives, and constraints that define a developing decision space.

In an attempt to categorise these various forms of IEC, it is possible to view complete

Figure 1. The user-centric system



human evaluation as explicit, whereas partial evaluation and interaction are less explicit, more subtle forms of human involvement. Completely implicit interaction occurs where users are unaware of their role in the evolution of a system (e.g., the Web-based tutorials of Semet, Lutton, Biojout, Jamont, & Collet, 2003). A simple implicit/explicit spectrum of interactive evolutionary approaches can thus be developed (Parmee & Abraham, 2005).

The following sections concentrate on the manner in which evolutionary search and exploration can generate high-quality information from complex design and decision-making environments. Such information can be utilised interactively to:

- support the identification of high-performance solutions where qualitative as well as quantitative objectives play a major role; and
- modify and refine design problem representation.

The first example relates to aesthetic judgement of EC-generated designs and is closer to the more traditional explicit interaction where user subjective evaluation is evident. However, this subjective evaluation complements detailed, machine-based quantitative evaluation. The second is current IEDS research relating to problem definition and the iterative interactive improvement of machine-based design representations that sits further toward the implicit end of the spectrum.

INTEGRATING AESTHETICS VIA INTERACTIVE EVOLUTIONARY DESIGN PROCESSES

This example brings together agent-based machine learning, evolutionary computing, and subjective evaluation in search for aesthetically pleasing, structurally feasible designs during the conceptual design process. Although significant theoretical work is evident with respect to the inclusion of aesthetics in computer-based design, application-based research has received less attention (Moore, Miles, & Evans, 1996a, 1996b; Saunders, 2001).

Figure 1 illustrates the main components of the system and the manner in which they interact. The user defines initial design requirements and aesthetically evaluates the designs generated by the Evolutionary Search, Exploration, and Optimisation System (ESEO) during the initial generations. The agents have multiple

tasks, which include the creation of the initial population based on design requirements, the monitoring of designs for feasibility during the ESEO processes, and evaluation of machine-based aesthetic criteria. The ESEO identifies design solutions that can be considered high performance in terms of Structural Feasibility and Stability, Materials Cost, and Rule-Based Aesthetics.

The project is initially considering three test domains: bridges; liquid containers such as wine glasses, chemical tanks, and so forth; and street furniture in the form of bench type structures. The following example concentrates on bridge design. Research involves the ACDDM Lab at Bristol UWE and the Institute of Machines and Structures at Cardiff University. The early stages have concentrated on development of highly flexible and robust representations of simple bridge structures and the identification of optimal solutions via basic evolutionary algorithms.

Representation

Problem representation affects search efficiency and evolutionary performance (Rosenman, 1997; Goldberg, 1989). Genetic algorithms (GAs), evolutionary strategies (ESs), and evolutionary programming (EP) representations are generally based on binary or real number parameter strings, but many alternatives are also available. Component-based and hierarchical representations support flexibility and robustness in terms of accommodating complex design entities with many related subsystems/components (e.g., Bentley, 2000; Cramer, 1985; Rosenman, 1996).

It is also necessary to consider which stochastic search process would best suit a chosen representation in terms of efficient negotiation of the design space. As a high degree of solution search and exploration is required, a population-based approach would seem most appropriate. GAs, EPs, and ESs offer high utility with differing operators. GAs use crossover as the main operator (Goldberg, 1989). ESs are similar to real parameter GAs without crossover, although crossover-like operators have been introduced (Deb, 2001). EPs (Fogel, 1988) is a purely mutation-based evolutionary algorithm and therefore perhaps represents the simplest of the three, as the selected representation does not need to support crossover between differing variable strings. Gas, however, require a representation that is robust enough to handle repeated crossovers whilst ensuring offspring feasibility. ESs could be considered to lie somewhere between these two. EPs have been selected in this work primarily to overcome feasibility maintenance problems relating to the component-based representation and crossover.

The initial goal was a flexible enough representation to model all possible designs which is robust enough to be manipulated by design search and exploration processes (Machwe, Parmee, & Miles, 2005). A collection-based object-oriented representation has been developed. A single population member (chromosome) is a collection of primitive elements that represent a design. Different elements with different design properties can be included in the set of possible design primitives. The evaluation of fitness of the structure and checking the structural integrity utilises secondary properties of the particular primitive element type. Figure 2 further clarifies the idea of using an object-based representation. A simple bridge design is basically divided into two separate collections. These are the *span element* collection containing elements which form the span of the bridge and the *support element* collection containing elements which form the support of the bridge. In the case of a simple, end-supported span, the support element collection will be empty.

The rectangular elements can either be part of a supporting element collection or a span element collection. Simple beam or angled beam

Figure 2. Details of the object-based representation

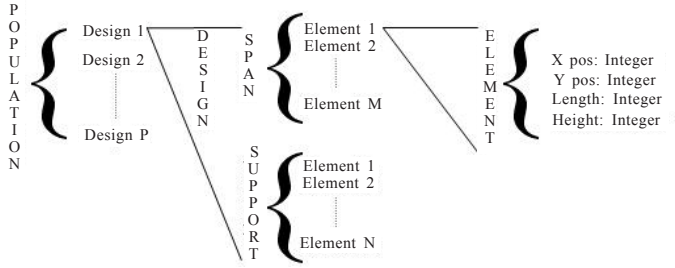
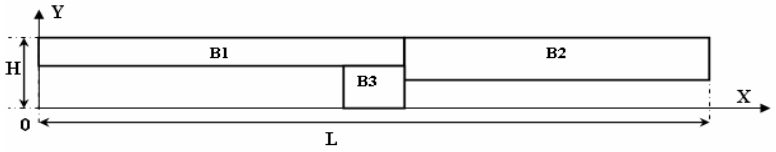


Figure 3. Design before mutation



span bridges with and without support require only two basic types of elements—the angled section element (to be used as a span element only) and a simple rectangle element, which can be used as both a spanning and supporting element. All elements have an assumed constant. The object-based representation can take advantage of the principles of object-oriented programming such as inheritance—that is, if we want to add a new kind of element, say curved span section, we could easily do so by extending the basic properties of element and adding the extra properties required for a curved section.

During initial design activity, it may not be desirable or possible to strictly parameterise a design. Designs make the transition from abstract concepts to well-defined specifications. The above object-based representation can represent designs at all levels, including at the abstract level by using high-level objects/elements and at the well-defined level as a set of specifications. Functionality can be modified based upon changing requirements. Thus objects offer a straightforward way to cover design activity at all levels. To produce the same in a simple string-based chromosome would require additional checks to ensure consistency of the chromosome is not violated once a bit or real number is added or removed, or its functionality modified. The overall system would be overly complex and difficult to manipulate.

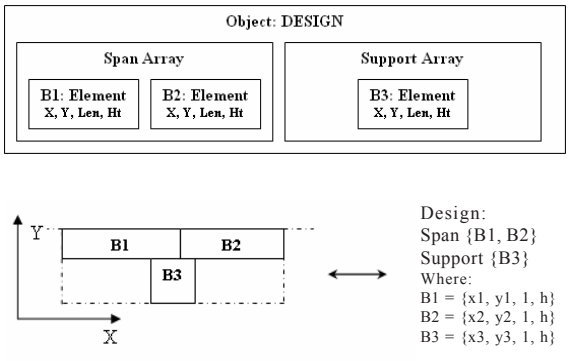
Mutation

Let us assume the chromosome to be mutated represents the design in Figure 3.

This is a basic beam bridge simply supported at either end with a single intermediate support (B3) and two span elements (B1, B2). L is the span and H is the maximum height of the bridge (see Figure 4).

The mutation is rule based, and a rule is selected randomly. There are separate rules for the two arrays of elements. The supports can only move left or right. Their height is based

Figure 4. Structure of a design chromosome



upon the thickness of the spanning element they support. Hence there are only four rules for supports: two rules for left and right movement, and two for increasing and decreasing width. The depth of each span element can vary, but they must have a level upper surface and must be continuous with no overlap or space between them. Thus for a simple element in a span, there are just two rules, namely to increase or decrease the span depth. Now, for example, if the selected rule for support (B3) is to move it left by a constant distance (say two units) and for span to decrease thickness of B2 support by constant units (say two units again), then the B3 object in the support array will have its X value attribute decreased by two units and the B2 object in the span array will have its height value attribute decreased by two units. The height attribute of the support will be automatically adjusted to make it continuous and remove any overlap at its new position. The mutated design is shown in Figure 5.

Introduction of Agency

Initial testing of the representation involved freeform assembly and evolution of simple bridge structures using GA, EP, and agent-based approaches. Evolving feasible structures proved a difficult and lengthy process, whereas rule-based agent assembly was straightforward and rapid. Thus a combined approach has been developed where agents create the initial population of structures and provide a continuous 'repair' capability, and an evolutionary system performs search, exploration, and optimisation across the space of possible structures.

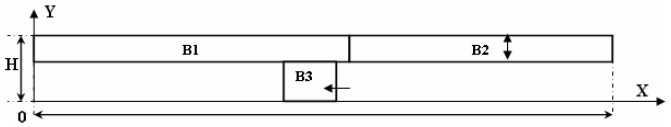
The construction and repair agents (CARAs) have the task of assembling structures with varying size and shape regarding span and supports. Agents are given specifications relating to restrictions on placement of supports and types of span section. Thus CARAs are informed of the design environment and will create initial population designs within it. The evolutionary process then takes care of the SEO process, with the CARAs keeping a check on potentially disruptive processes and repairing the structure, where necessary, to ensure feasibility.

Simple Structural Criteria

The CARAs can currently create three kinds of bridges:

- simple beam bridges without supports (Type 1a);

Figure 5. Design after mutation



- simple beam bridges with supports (Type 1b); and
- simple beam bridges with sloping span sections and supports (Type 2).

An initial population can consist of a mixture of these designs which are then evaluated against structural and aesthetic criteria. In terms of structure, solutions are assessed via simple length/depth ratios and minimising material. Column design is subjected to buckling criteria.

Fitness Evaluation for Type 1A

Type 1a is treated as a simple beam deflection problem under uniform distributed loading. A simple heuristic relating to an ideal length-to-height ratio for a span element of 20:1 is utilised—that is, the closer a span section is to the ideal ratio (R), the better its fitness.

L_i and H_i are the length and height of the *i*th span element.

$$F_i = \left| R - (\frac{l_i}{h_i}) \right| \quad (1)$$

$$Stability = \frac{1}{(1+\Sigma F_i)} \quad (2)$$

To ensure the overall integrity of the structure, the above equations are used. It is evident that the closer the dimensions of the span elements are to the ideal ratio (*R*), the lower the value of F_i will be. At the minimum, all F_is are equal to zero and thus stability is equal to one.

Fitness Evaluation for Type 1b and Type 2

In Type 1b the buckling in the columns due to the weight of the loaded beam is also considered using:

$$P' = \frac{\pi^2 EI}{H^2} \quad (3)$$

where: P' = maximum possible load, E = modulus of elasticity, I = moment of inertia, and H = column height. The load on column is determined from the length of the beam between the end supports, calculating the loading and distributing this loading across intermediate supports. A column satisfying the buckling criteria can either increase or decrease in thickness. Otherwise it can only increase in thickness.

Example

A basic EP approach is used with a population size of 100 solutions. Tournament selection is utilised with a tournament size of 10, and the system is run for 100 generations. A few members from the initial population are shown in Figure 6.

The initial population consists of three different kinds of designs: a simply supported

Figure 6. Sample of mixed initial population of bridge shapes

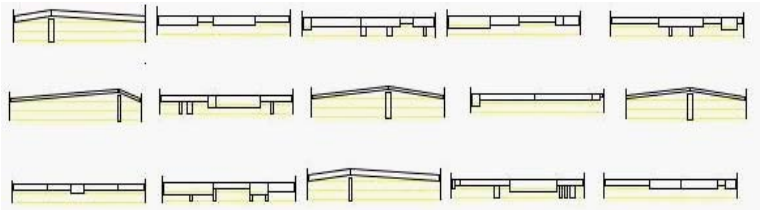
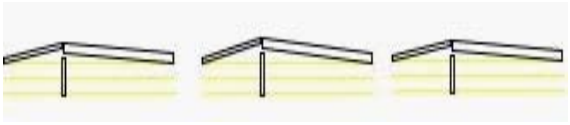


Figure 7. Optimised bridges



span, a span with columns, and an angled span bridge. After 100 generations the optimal designs shown in Figure 7 are achieved. The angled span bridges turn out to be most efficient in terms of structural criteria (i.e., stability and material usage). The other two design types have been evolved out of the population.

Aesthetics and User Evaluation

It is very difficult to integrate subjective aesthetic criteria with machine-based design unless user interaction plays a significant role. Moore et al. (1996a, 1996b) have published significant work in this area which has contributed to a New South Wales RTA set of guidelines on bridge aesthetics (Road and Traffic Authority, 2004). However, aesthetics evaluation can only be partially quantified by generic guidelines and rules. While aesthetically pleasing shapes can be explicitly specified to some extent, complete aesthetic evaluation must also involve the designer—that is, valuation must involve both rule-based and subjective factors. In the present system the following aesthetics have been hard coded:

- symmetry of support placement (A1),
- slenderness ratio (A2),
- uniformity in thickness of supports (A3), and
- uniformity in thickness of span sections (A4).

Many other such rules are currently being evaluated and included in the work as the design representation is developed to support detailed aesthetic evaluation. Each aesthetic rule is evaluated by a separate *aesthetic agent*. The *rule-based aesthetic fitness* is calculated as:

$$Aesthetic\_Fitness = \sum_{i=1}^{4} w_i A_i \qquad (4)$$

where w_i are weights for each of the aesthetic rules (A_i = A1 to A4) which can also be modified online.

In addition, *user-assigned aesthetic fitness* (Ufit) is the fitness given to a design directly by the user on a scale of 0 to 10 (10 being the best). The user can also mark solutions for preservation into the next generation. Overall user evaluation operates thus:

1. User stipulates the frequency of user interaction (e.g., once every 10 generations).
2. User aesthetically evaluates a preset number of population members from the initial population (usually the top 10 members, i.e., those with highest fitness regarding stability, material usage, and explicitly defined aesthetic criteria).
3. The EP system runs.
4. Population members are aesthetically evaluated by the user every ***n*** generations.
5. Repeat steps 3 and 4 until user terminates the evolutionary process.

The overall fitness function now includes *aesthetic fitness* and *user-assigned aesthetic fitness*. Furthermore, weights have been added (w1 to w4) to each of the objectives which the user can modify online to influence evolutionary direction:

$$Fitness = (w1 * Stability) + \left(\frac{w2}{Material\_Usage}\right) + (w3 * Aesthetic\_Fitness) + (w4 * Ufit) \qquad (5)$$

Figure 8. Aesthetically pleasing cross-sections

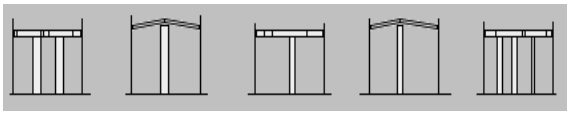


Figure 8 shows aesthetically pleasing cross-sections after 30 generations with user evaluation every 10 generations. The aesthetic objectives (A1 to A4) are clearly reflected in them. The span elements are of the same size. The supports are of nearly uniform thickness, and their placement is also symmetric.

Furthermore, due to user interaction, the optimised shapes are not limited to angled sections but take on a variety of different aesthetically pleasing shapes that not only satisfy the explicitly defined aesthetic guidelines (A1 to A4) but also the implicit aesthetics of the user (Ufit).

Incorporating Learning

Current work involves the introduction of supervised learning taking user evaluation into account. Since there is a natural classification in the designs (i.e., angled spans, supported beams, and unsupported beams), learning is attempted at two levels. The first level determines user preference for one of the three types of bridge design. This is achieved by evaluating the relative difference between user-assigned fitness (or rank) for each type of design. The second level assesses what kind of features the user finds pleasing in the different designs. Figures 7 and 8 indicate that for the angled spans, there are two features that catch the eye immediately. These are the peak of the bridge (that is the location of the rise point) and the thickness of the span sections. Such features are converted into fuzzy variables to create an aesthetic model of the particular bridge type. For the angled section the following fuzzy variables are used to specify the aesthetic model:

1. **Peak:** Left, Central, Right
2. **Difference in Span Thickness:** Left Thicker, Equal Thickness, Right Thicker
3. **Average Thickness:** Low, Medium, High
4. **Column Thickness:** Low, Medium, High
5. **User-Assigned Fitness (Ufit):** Low, Medium, High

Similar models can be created for supported beam spans and unsupported beam spans. Based on this model a fuzzy rule generator has been implemented. Initial results have been encouraging. The intention is that as search progresses there will be a gradual lessening of the degree of user interaction allied with an increasing degree of autonomous machine-based solution evaluation involving both aesthetic and structural criteria. This addresses the problem of user fatigue.

EVOLVING THE PROBLEM SPACE THROUGH INTERACTIVE EVOLUTIONARY PROCESSES

The second illustration of the utilisation of user-centric evolutionary computing relates to a more implicit form of interaction. This is aimed primarily at the extraction of high-quality information and the succinct presentation of such information to the designer/decision maker in such a manner that supports a better understanding of complex relationships between variables, multiple objectives, and constraints during conceptual design. This complements the initial IED concept by further attempting to meld experiential knowledge and intuition with powerful machine-based search, exploration, and information processing.

Machine-based problem representations support exploration through the evaluation of solu-

tions against seemingly relevant criteria. Although initial representations may be relatively basic, and confidence in model output will be low, such representations can provide essential problem insight despite apparent shortfalls. Identified high-performance solutions based upon quantitative criteria followed by qualitative human evaluation may provide an indication of concept viability and model fidelity. An iterative user/machine-based process can commence where gradual improvements in understanding contribute to the development of better representations, a growing knowledge base, and the establishment of computational models that support rigorous analysis—that is, a process emerges that supports the development of representation through knowledge discovery.

An initial variable parameter set may be selected with later addition or removal of variables as the sensitivity of the problem to various aspects becomes apparent. Constraints may be treated in the same way with the added option of softening them to allow exploration of non-feasible regions. Included objectives may change as significant payback becomes apparent through a reordering of objective preferences. Some non-conflicting objectives may merge, whilst difficulties relating to others may require serious re-thinking with regard to problem formulation. The initial design space is therefore a moving feast rich in information (Parmee, 2002).

The visualisation of variable and objective space from cluster-oriented genetic algorithm (COGA) output provides a variety of perspectives illustrating complex relationships (Parmee & Abraham, 2004). This information is further defined by data mining, processing, and visualisation techniques. The intention is to support implicit learning and reduce complexity by supporting the designer development of a quantitative and intuitional understanding of the problem. This leads to the iterative model development described above.

COGAs and the MiniCAPs Model

Cluster-oriented genetic algorithms provide a means to identify high-performance (HP) regions of complex conceptual design spaces and enable the extraction of information from such regions (Parmee, 1996). COGAs identify HP regions through the online adaptive filtering of solutions generated by a genetic algorithm. COGA can be utilised to generate design information relating to single and multi-objective domains (Parmee & Bonham, 1999). The technique has been well documented (see http://www.ad-comtech.co.uk/Parmee-Publications.htm for relevant papers).

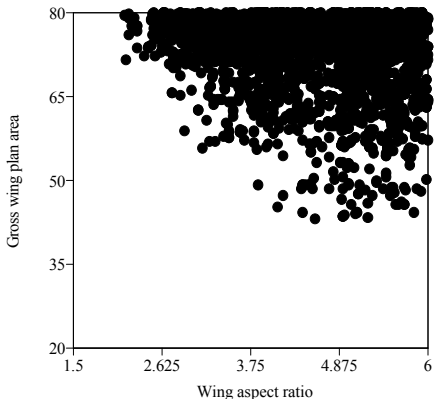
The research utilises the BAE Systems' MiniCAPs model, a simplified version of a suite of preliminary design models for the early stages of military aircraft airframe design and initially developed for research relating to the development of the IED concept. The model comprises nine continuous input variables and 12 continuous output parameters relating to criteria such as performance, wing geometry, propulsion, fuel capacity, structural integrity, and so forth. Input variables are:

1. Climb Mach Number (CLMN)
2. Cruise Height (CH)
3. Cruise Mach Number (CRMN)
4. Gross Wing Plan Area (GWP)
5. Wing Aspect Ratio (WAR)
6. Wing Taper Ratio (WTR)
7. Wing Lead Edge Sweep (WLES)
8. Wing T/C Ratio (WTCR)
9. By Pass Ratio (BPR)

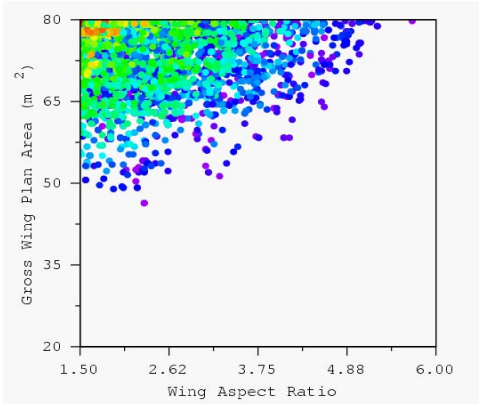
Identifying High-Performance Regions Relating to Differing Objectives

Figures 9(a), (b), and (c) show HP regions comprising COGA-generated solutions relating

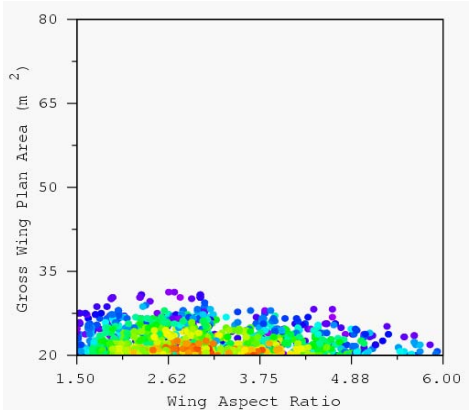
Figure 9. COGA-generated high-performance regions relating to three differing objectives: (a) FR—Ferry Range; (b) ATR1—Attained Turn Rate; (c) SEP1—Specific Excess Power. All projected onto the GWPA (Gross Wing Plan Area)/WAR (Wing Aspect Ratio) variable hyperplane



(a)



(b)



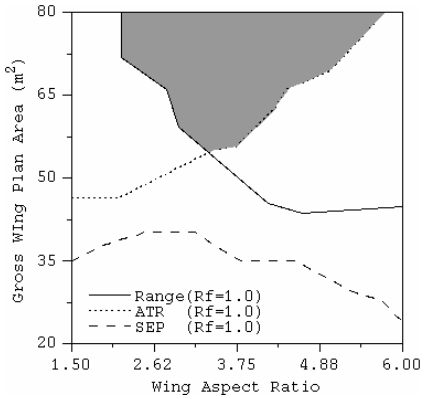
(c)

N.B. Colour versions of figures can be found at: http://www.ad-comtech.co.uk/cogaplots.htm

to three of the 12 MiniCAPS objectives (Ferry Range—FR, Attained Turn Rate—ATR1, and Specific Excess Power—SEP1) projected onto a variable hyperplane relating to two of the nine variables utilised in the search process. This projection allows the designer to visualise the

HP regions, identify their bounds, and subsequently reduce the variable ranges as described in previously referenced papers. These papers also introduce the projection of these differing objective HP regions onto the same variable hyperplane as shown in Figure 10 from which the degree of objective conflict immediately becomes apparent to the designer. The emergence of a mutually inclusive region of HP solutions relating to the ATR1 and FR objectives indicates a low degree of conflict, whereas the HP region relating to SEP1 is remote (in

Figure 10. All HP regions projected onto the GWPA/WAR variable hyperplane



variable space) to both the ATR1 and FR regions, indicating a higher degree of conflict.

There is much information contained in the HP regions relating to appropriate variable ranges for single objectives, degree of conflict between multiple objectives, and the emergence and definition of mutually inclusive (common) HP regions. This graphical representation provides an excellent spatial indication of the degree of objective conflict. However, searching through all possible two-dimensional variable hyperplanes to visualise such information is not a feasible approach. Recent research has resulted in single graphical representations that can present all variable and objective data whilst providing links to other visual perspectives. The *parallel coordinate box plot* (PCBP) representation shown in Figure 11 is one such graphic that provides a central repository containing much single and multiple-objective solution information.

Parallel Coordinate Box Plots

Parallel coordinate plots (Inselberg, 1985) appeared to offer potential in terms of providing a single graphic illustrating complex relationships between variable and objective space. Parallel coordinate representation displays each variable dimension vertically parallel to each other. Points corresponding to a solution's value of that variable can then be plotted on each vertical variable axis. It is thus possible to show the distribution of solutions in all variable dimensions and the correlation between different dimensions. The disadvantage of the technique when attempting to include multiple objectives is that the density of the information presented hinders perception (Parmee & Abraham, 2004, 2005). To overcome the 'data density' problem, three modifications to the standard parallel coordinate representation have been included:

1. additional vertical axes for each variable so that each objective can be represented,
2. an indication of the degree of HP region solution cover across each variable range, and
3. the introduction of box plots to indicate skewness of solutions across each variable range.

This PCBP provides a much clearer graphic (see Figure 11). The vertical axis of each variable is scaled between the minimum and

Figure 11. Parallel box plot of solution distribution of each objective across each variable dimension

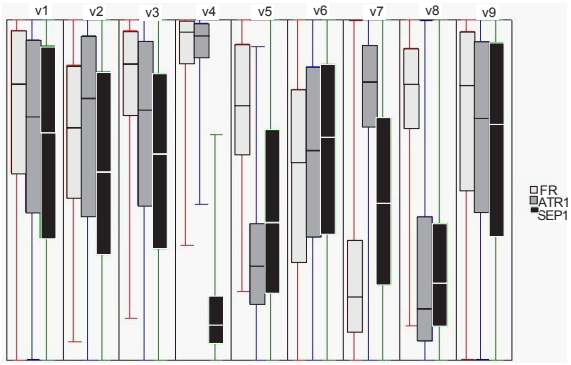
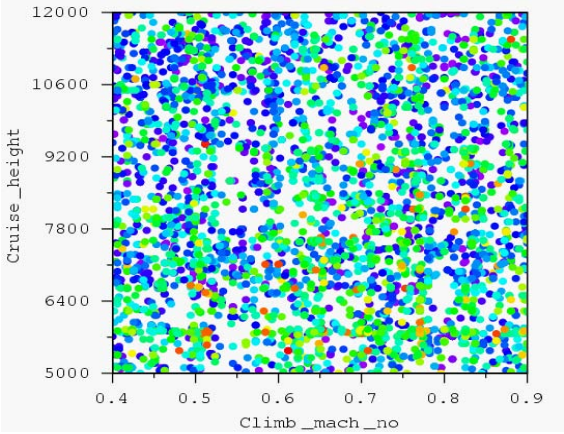


Figure 12. Projection of results onto variable 1/variable 2 hyperplane for Attained Turn Rate (ATR1) objective



maximum value of the variable in the HP region solutions of each objective. The length of the axis represents the normalised ranges of variable values present in an HP region. Where an HP solution set does not fully extend across the variable range, the axis is terminated by a whisker at the maximum or minimum value of the variable. The colour-coded box plots relate to each objective (i.e., SEP1, ATR1, and FR). The median is marked within the box, and the box extends between the lower and upper quartile values within the variable set. The PCBP clearly visualises the skewness of solution distribution relating to each objective in each variable dimension which provides an indication of the degree of conflict between objectives.

For instance, it is apparent that all three objective boxes overlap in the case of variables 1, 2, 3, 6, and 9. However, significant differences in the distribution of the boxes are evident in terms of at least one objective where variables 4, 5, 7, and 8 are concerned. Variables 4 and 5 are Gross Wing Plan Area and Wing Aspect Ratio. The conflict between SEP1 and FR/ATR1 evident in Figure 10 is strongly reflected in the HP solution distribution indicated by the whisker truncation of variable 4 in Figure 11 and in the box plots of that variable. In terms of variable 5, the whisker terminations relating to ATR1 and FR in Figure 11 reflect the extent of the solution distribution across their HP regions in Figure 10. The box plots also reflect the relative distribution of HP solutions of all objectives along that variable plane as illustrated in Figure 10. Figure 12 shows a projection of the ATR1 HP region onto the Cruise Height (variable 1) and Climb Mach No (variable 2) hyperplane. The relatively uniform distribution of HP solutions across the hyperplane is reflected in the appropriate variable plots of Figure 11.

The PCBP represents a single graphic from which the designer can perceive which variables are causing high degrees of objective conflict. To get an alternative, very clear perspective of these conflicts, any two of these variables can be selected to also view the relevant graphics similar to Figure 10. Further reinforcement can be obtained from the perspectives explored in the following section relating to projections upon objective space. Improved understanding can lead to developments of the computational representation and to appropriate setting of objective preferences.

Projection of COGA Output on to Objective Space

The HP region solutions for ATR1 and FR can be projected onto objective space as shown in Figure 13. A relationship between the HP region solutions and a Pareto-frontier emerges along the outer edge of the plot (Parmee & Abraham, 2004) despite the fact that the working principle of COGA is very different to that of evolutionary multi-objective algorithms (Deb,

Figure 13. Distribution of FR and ATR1 solutions in objective space

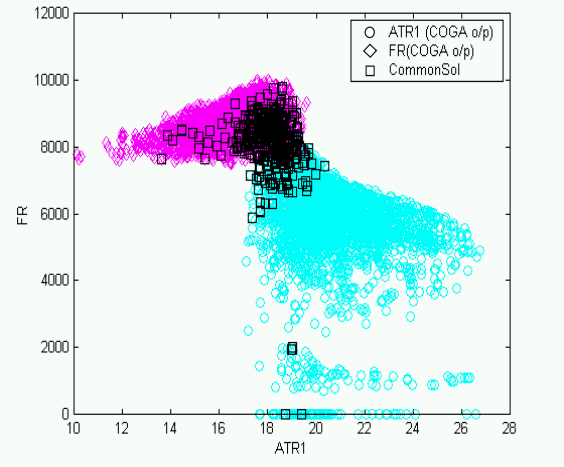
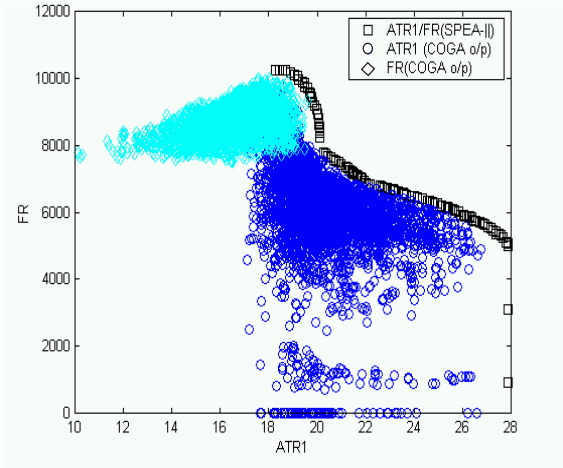


Figure 14. Distribution of ATR1 and FR solutions against SPEA-II Pareto front



2001), which tend to use a non-dominance approach.

For comparative purposes, Figure 14 illustrates the distribution of COGA output and SPEA-II (Zitzler et al., 2002) Pareto-front output in objective space. Using a standard multi-objective GA (MOGA), it is possible to obtain solutions lying along the Pareto-front, but difficult to explore the relationship between variable and the objective space. However, it is likely that the designer is also interested in solutions that lie around particular sections of the Pareto-front.

The COGA approach therefore provides a good visual indication of the degree of conflict between objectives, an opportunity to explore varying objective preferences and view their effect upon HP region bounds, and the ability to generate an approximate Pareto-front relating to the objectives under investigation plus solutions around the Pareto-front. This is in addition to the utility of COGA in single-objective space as described in previous referenced papers. All of this utility directly complements the original IEDS concept regarding information extraction, processing, and presentation.

SUMMARY AND CONCLUSION

The aesthetics work reveals a significant potential in terms of the development of systems that include criteria ranging from purely quantitative through to purely subjective. Ultimately the system will be required to give a comparative indication in terms of aesthetically pleasing design and likely cost whilst indicating structural feasibility.

The introduction of such an interactive process also poses many questions such as:

- How many designs from each population should be presented to the user?
- How should these be selected?
- How many evaluations can a user be expected to perform before becoming fatigued?

These questions have been repeatedly posed, but seldom successfully addressed within the interactive evolutionary computing (IEC) community. Our continuing research is addressing these issues and, in particular, the user-fatigue aspect.

The integration of user preference and user-varied objective weights supports the transfer of subjective evaluation from the user to a design/decision-making system. In order to address the third question above, a machine learning system is required which learns the preferences of the user during the interactive process. User fatigue is perhaps the major stumbling block in the development of successful systems. Much work is required to develop an appropriate, fully functioning, machine-learning sub-system. The fuzzy rule-based learning system is not ideal, and current work is investigating a case-based approach.

The developing system should be seen as a generic framework for the integration of user-evaluation with any preliminary design/decision-making domain. The CARA-EP representation concept should be portable across many problem domains. Any system must significantly decrease the load on the user as early as possible in the evolutionary process. A multi-agent-based learning environment is therefore under investigation that gradually replaces the subjective criteria evaluation.

It is apparent from previous research and the research presented here that COGA-generated data can provide visual representations in variable space of the degree of conflict between objectives and excellent spatial indications of the distribution of high-performance solution regions relating to a number of objectives. It is also apparent that the COGA HP solution sets, when projected onto objective space, provide the designer with an opportunity to explore a wealth of HP solutions that offer varying degrees of objective compromise and a variety of design characteristics. The non-dominance sorting of these solutions also provides an approximate Pareto-frontier illustrating succinct available trade-offs. The direct mapping of solutions between objective and variable space facilitates an understanding of the relative utility of solutions in terms of preferred variable ranges and particular design characteristics.

The PCBP of Figure 11 offers a first point of call for the designer to get an overview of the varied information available from COGA output. The intention is that the COGA graphical perspectives will be available through simple menu/clicking operations from the central PCBP image. These differing perspectives are seen as essential aids to understanding overall complexities relating to the two dependant design spaces (variable vs. objective space).

There is a wealth of information available from COGA output relating to single objective solutions that is also inherent within the multi-objective output. Hence the utility of the approach should be assessed across both areas. The information available from single-objective HP regions has been fully discussed in previous referenced papers.

User-centric techniques described in the chapter and variations of them are also currently being applied in the conceptual design of submersible vehicles (Parmee & Abraham, 2005), pharmaceutical drug design and discovery (Sharma & Parmee, 2005), and conceptual software design (Simons & Parmee, 2004). Details of this associated work can be found on the ACDDM Web site at www.ad-comtech.co.uk/ACDDM_Group.htm.

ACKNOWLEDGMENT

The authors wish to acknowledge the contribution to the bridge aesthetics work of Professor John Miles of the Institute of Machines and Structures at the University of Cardiff.

REFERENCES

Bentley, P. J. (2000, April 26-28). Exploring component-based representations—The secret

of creativity by evolution? In I. C. Parmee (Ed.), *Proceedings of the 4th International Conference on Adaptive Computing in Design and Manufacture* (ACDM 2000) (pp. 161-172). University of Plymouth, UK.

Deb, K. (2001). *Multi-objective optimisation using evolutionary algorithms.* New York: John Wiley & Sons.

Carnahan, B., & Dorris, N. (2004). Identifying relevant symbol design criteria using interactive evolutionary computation. *Proceedings of the Genetic and Evolutionary Computing Conference (GECCO),* Seattle, USA (pp. 265-273).

Cramer, N. L. (1985). A representation for the adaptive generation of simple sequential programs. *Proceedings of the International Conference on Genetic Algorithms and Their Application,* Pittsburgh (pp.183-187).

Fogel, D. B. (1988). An evolutionary approach to the travelling salesman problem. *Biological Cybernetics, 60*(2), 139-144.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* Reading, MA: Addison-Wesley.

Herdy, M. (1997). Evolutionary optimisation based on subjective selection—evolving blends of coffee. *Proceedings of the 5th European Congress on Intelligent Techniques and Soft Computing (EUFIT'97),* Aachen, Germany (pp. 640-644).

Inoue, T., Furuhashi, T., & Fujii, M. (1999). Development of nurse scheduling support system using interactive evolutionary algorithms. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* (SMC'99) (pp. 533-537).

Inselberg, A. (1985). The plane with parallel coordinates. *The Visual Computer, 1,* 69-91.

Levine, D., Facello, M., & Hallstrom, P. (1997). Stalk: An interactive system for virtual molecular docking. *IEEE Computer Science Engineering Magazine, 4*(2), 55-65.

Machwe, A. Parmee, I. C., & Miles, J. C. (2005). Overcoming representation issues when including aesthetic criteria in evolutionary design. *Proceedings of the ASCE International Conference in Civil Engineering,* Mexico [CD-ROM].

Moore, C. J., Miles, J. C., & Evans, S. J. N. (1996a). Establishing a knowledge base for bridge aesthetics. *Structural Engineering Review, 8*(2/3), 247-258.

Moore, C. J., Miles, J. C., & Evans, S. J. (1996b). Innovative computational support in bridge aesthetics. *Transport Research Record, 1549,* 1-11.

Parmee, I. C. (1996). The maintenance of search diversity for effective design space decomposition using cluster-oriented genetic algorithms (COGAs) and multi-agent strategies (GAANT). *Proceedings of the 2nd International Conference on Adaptive Computing in Engineering Design and Control,* Plymouth, UK (pp. 128-138).

Parmee, I. C., & Bonham, C. R. (1999). Towards the support of innovative conceptual design through interactive designer/evolutionary computing strategies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing Journal, 14,* 3-16.

Parmee, I., Watson, A., Cvetkovic, D., & Bonham, C. R. (2000). Multi-objective satisfaction within an interactive evolutionary design environment. *Journal of Evolutionary Computation,* 8(2), 197-222.

Parmee, I. C. (2002). Improving problem definition through interactive evolutionary computation. *Journal of Artificial Intelligence in*

Engineering Design, Analysis and Manufacture (Special Issue: Human-Computer Interaction in Engineering Contexts), 16(3), 185-202.

Parmee, I. C., & Abraham, J. R. (2005). Interactive evolutionary design. In Y. Jin (Ed.), *Knowledge incorporation in evolutionary computation* (pp. 435-458). Berlin: Springer-Verlag.

Parmee, I. C., & Abraham, J. R. (2004). Supporting implicit learning via the visualisation of COGA multi-objective data. *Proceedings of the IEEE Congress on Evolutionary Computation,* Portland (pp. 395-402).

Rosenman, M. A. (1997). An exploration into evolutionary models for non-routine design. *Artificial Intelligence in Engineering, 11*(3), 287-293.

Rosenman, M. A. (1996). A growth model for form generation using a hierarchical evolutionary approach. *Microcomputers in Civil Engineering (Special Issue on Evolutionary Systems in Design), 11,* 161-172.

Saunders, R. (2001). *Curious design agents and artificial creativity.* PhD Thesis (Electronic edition), Faculty of Architecture, University of Sydney, Australia.

Road and Traffic Authority. (2004). *New South Wales (NSW) report: Bridge aesthetics: Design guidelines to improve the appearance of bridges in New South Wales.* Report No. RTA/Pub 04.003.

Semet, Y., Lutton, E., Biojout, R., Jamont, Y., & Collet, P. (2003). *Artificial ants colonies & e-learning: An optimisation of pedagogical paths*. Retrieved from http://www.ad-comtech.co.uk/Workshops.htm/

Simons, C. L., & Parmee, I. C. (2004). 35 years on: To what extent has software engineering design achieved its goals? *IEEE Software, 150*(6), 337-350.

Sims, K. (1991). Artificial evolution for computer graphics. *Computer Graphics, ACM SIGGRAPH Proceedings* (Vol. 25, pp. 319-328).

Sharma, B., & Parmee, I. C. (2005). Drug discovery: Exploring the utility of cluster-oriented genetic algorithms in virtual library design. *Proceedings of the IEEE Congress on Evolutionary Computation,* Edinburgh, UK (pp. 668-675).

Takagi, H. (1998). Interactive evolutionary computation: Cooperation of computational intelligence and human KANSEI. *Proceedings of 5th International Conference on Soft Computing and Information/Intelligent Systems,* Japan (pp. 41-50).

Chapter XXVIII
Genetic Algorithms for Organizational Design and Inspired by Organizational Theory

Tian-Li Yu
University of Illinois at Urbana-Champaign, USA

Ali A. Yassine
University of Illinois at Urbana-Champaign, USA

David E. Goldberg
University of Illinois at Urbana-Champaign, USA

ABSTRACT

Modularity is widely used in system analysis and design such as complex engineering products and their organization, and modularity is also the key to solving optimization problems efficiently via problem decomposition. We first discover modularity in a system and then leverage this knowledge to improve the performance of the system. In this chapter, we tackle both problems with the alliance of organizational theory and evolutionary computation. First, we cluster the dependency structure matrix (DSM) of a system using a simple genetic algorithm (GA) and an information theoretic-based metric. Then we design a better GA through the decomposition of the optimization problem using the proposed DSM clustering method.

INTRODUCTION

Modularity is ubiquitous (Schilling, 2002). The concept can be found in and applied to many disciplines: biology (Hartwell, Hopfield, Leibler, & Murray, 1999; Andrews, 1998), psychology (Fodor, 1996), social networks (Newman & Girvan, 2004; Guimerà, Danon, Diaz-Guilera, Giralt, & Arenas, 2003), engineering design (Ishii & Yang, 2003; Fixson, 2003), engineering

optimization (Altus, Kroo, & Gage, 1996), and software development (Sullivan, Griswold, Cai, & Hallen, 2001), to name a few. Such modular structure in many of these natural and man-made complex systems is believed to improve the functionality, performance, and robustness of these systems (Alon, 2003; Baldwin & Clark, 2000). It allows increased product or organizational variety (Ulrich & Eppinger, 2000), increased rates of technological or social innovation (Baldwin & Clark, 2000), increased opportunities for market dominance through interface capture (Moore, 1999), and increased specialization at firm level which in turn may allow more flexible response to environmental change (Sanchez & Mahoney, 1996).

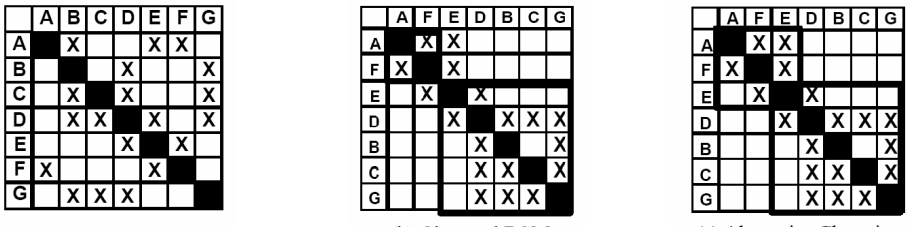
Therefore, revealing the modular structure of complex systems is crucial for developing performance enhancement techniques. In other words, we first have to discover modularity in a system, and then leverage this knowledge to improve the performance of the system. In this chapter, we first propose a genetic algorithm-based method to discover the modular structure of a decomposed system (e.g., a product or organization), then we propose a new, improved genetic algorithm (GA) based on the modular arrangement of the system.

The proposed clustering method is developed based on a matrix representation of a graph, called the dependency structure matrix or DSM (Yassine & Braha, 2003); the minimum description length (MDL) principle (Rissanen, 1978, 1999; Barron, Rissenen, & Yu, 1998; Lutz, 2002); and a simple genetic algorithm (Goldberg, 1989). The method is capable of partitioning the product (or organizational) architecture into an *optimal* set of modules (or teams) and can be fine-tuned to mimic clustering arrangements proposed by human experts. Our proposed clustering algorithm is an improvement over other existing algorithms for two reasons: (1) existing algorithms are manual, very dependent on human expertise, and consequently hard to automate or replicate (McCord & Eppinger, 1993; Pimmler & Eppinger, 1994; Stone, Wood, & Crawford, 2000; Gonzalez-Zugasti, Otto, & Baker, 2000); and (2) existing algorithms use simple mathematical constructs to discriminate between modules, and consequently these algorithms collapse when confronted with complex product architectures (Fernandez, 1998; Thebeau, 2001; Whitfield, Smith, & Duffy, 2002).

It is interesting that the clustering techniques can help us better design a GA that decomposes the optimization problem by recognizing the modularity between decision variables. Holland (1975) has suggested that operators learning linkage information to recombine alleles might be necessary for GA success. Many such methods have now been developed to solve this including perturbation (Goldberg, Korb, & Deb, 1989; Goldberg, Deb, Kargupta, & Harik, 1993; Kargupta, 1996; Munetomo & Goldberg, 1999), linkage learning schemes (Harik & Goldberg, 1996; Smith, 2002), and model-building schemes (Bosman & Thierens, 1999; Harik, 1999; Pelikan, Goldberg, & Cantú-Paz, 1999). The genetic algorithm developed in this chapter adds to the literature of competent GAs with a method inspired by organizational theory. In particular, the proposed *dependency structure matrix genetic algorithm* (DSMGA) is able to identify building blocks (BBs) with the help of a DSM and accomplish BB-wise crossover. As the experimental results suggest, compared to a simple genetic algorithm, the DSMGA is able to maintain a reliable solution quality under tight, loose, and random linkage with the same amount of function evaluations.

The rest of the chapter proceeds as follows. The next section provides a quick overview of the DSM method. We then propose a metric to evaluate different architectural arrangements

Figure 1. DSM clustering examples



(a) Original DSM *(b) Clustered DSM* *(c) Alternative clustering*

based on the MDL principle, describe the MDL-DSM clustering methods in detail, and show several illustrative examples. Next, we demonstrate this new clustering method using a DSM for a 10 MWe gas turbine, and we utilize this DSM clustering method to design a better genetic algorithm—DSMGA. Finally, we conclude the chapter.

THE DEPENDENCY STRUCTURE MATRIX (DSM) METHOD

A DSM is a matrix representation of a graph (Steward, 1981). The nodes of the graph (which represent components of a product or system) correspond to the column and row headings in the matrix (Eppinger, Whitney, Smith, & Gebala, 1994; Yassine & Braha, 2003). The arrows (which represent relationships between components) correspond to the *X* marks inside the matrix. For example, if there is an arrow from node C to node A, then an *X* mark is placed in row A and column C. Diagonal elements have no significance and are normally blacked out or used to store some element-specific attribute(s). Alternatively, number "one" can be placed instead of an *X* and "zero" instead of a blank. This makes the DSM a binary matrix with entries $d_{ij} = 0$ or 1.

Once the DSM for a product is constructed, it can be analyzed for the identification of modules—a process referred to as clustering. The goal of DSM clustering is to find subsets of DSM elements (i.e., clusters or modules) that are mutually exclusive or minimally interacting. In other words, clusters contain most, if not all, of the interactions (i.e., DSM marks) internally, and the interactions or links between separate clusters are eliminated or minimized (Fernandez, 1998). As an example, consider the DSM in Figure 1. As can be seen in Figure 1(b), the original DSM was rearranged (by simply swapping the position of rows and columns) to contain most of the interactions within two separate blocks or modules: *AF* and *EDBCG.* However, three interactions are still outside any block. An alterative arrangement is suggested in Figure 1(c). This arrangement suggests the forming of two overlapping modules (i.e., *AFE* and *EDBCG*).

The DSM representation of a system/product architecture has proved useful because of its visual appeal and simplicity, and numerous researchers have used it to propose architectural improvements by simple manipulation of the order of rows and columns in the matrix (McCord & Eppinger, 1993; Pimmler & Eppinger, 1994). In an attempt to automate this manual process of DSM inspection and manipulation, Fernandez (1998) used a DSM model with simulated annealing search techniques in order to find "good" DSM clustering arrangements. In his approach, each element is placed

in an individual set and bids evaluated from all the other sets (clusters). If any cluster is able to make a bid that is better than the current base case, then the element is moved inside the cluster. The objective function is therefore a trade-off between the costs of being inside a cluster and the overall system benefit. Sharman and Yassine (2004) attempted using the clustering algorithm described in Fernandez (1998) on an industrial gas turbine. However, he showed that this algorithm is incapable of predicting the formation of "good" clustering arrangements for complex product architectures due to the oversimplification of the objective function utilized and the frequent susceptibility of the search algorithm that used to be trapped in local optimal solutions. In a similar venue, Whitfield et al. (2002) used genetic algorithms to form product modules. Their algorithm is also built upon the same concepts introduced by Fernandez and as such suffers from similar problems.

The problem with applying automated clustering algorithms to complex DSMs is that they find it hard to extract the relevant information from the data and then to convey it to the user. This is most noticeable in the poor handling of overlapping clusters, bus modules, and three-dimensional structures (Sharman & Yassine, 2004).

MINIMUM DESCRIPTION LENGTH-BASED METRIC

The lack of an efficient clustering method particularly suited for analyzing product and organizational architectures motivated us to seek a better clustering metric based on information theoretic measures. Sharman and Yassine (2004) outlined the requirements necessary for the development of a new clustering metric and its corresponding algorithm:

1. The algorithm should be able to suggest the optimal number of clusters.
2. The algorithm should be able to detect the existence of bus modules.
3. The algorithm should be able to detect overlapping clusters and three-dimensional structures.

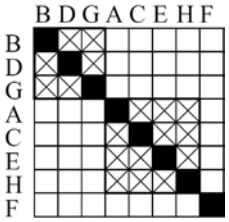
While the above two requirements can be addressed by the appropriate choice of a clustering metric, as will be discussed later in this chapter, the third requirement is directly related to the proposed search strategy and the encoding.

Model Description

Suppose that we have a model which describes a given product structure or a data set. Usually, the model does not completely describe the given data; otherwise, the model would become too complicated. Therefore, the description length needed to describe the whole given data consists of two parts: the *model description* and the *mismatched data description.* This scheme may be easier to understand in light of the following sender-receiver example.

Assume that a sender has a given data set which is needed by the receiver. Given a model that approximately (i.e., not exactly) describes the given data set, the sender first sends the model (i.e., model description) to the receiver. To ensure that the receiver gets exactly the same data set, the sender is also required to send the data which are mis-described (i.e., mismatched data description) by the model sent earlier. If the model is too simple, the model description is short; but many data mismatches exist, and the mismatched data description becomes longer. On the other hand, a complicated model reduces mismatched data, but the model description is longer.

Figure 2. A clustering arrangement of a DSM



Length	log n_n	3 log n_n	log n_n	4 log n_n
Description	3	*B,D,G*	4	*A,C,E,H*

The minimum description length (MDL) principle (Rissanen, 1978, 1999; Barron et al., 1998; Lutz, 2002) satisfies our needs for dealing with the above tradeoff. The MDL can be interpreted as follows: *among all possible models, choose the model that uses the minimal length for describing a given data set (that is, model description length plus mismatched data description length).* There are two key points that should be noted when MDL is used: (1) the encoding should be uniquely decodable, and (2) the length of encoding should reflect the complexity. For example, the encoding of a complicated model should be longer than that of a simple model. Next, we define the MDL clustering metric in detail.

Model Encoding

The way we encode the model is straightforward. The description of each cluster starts with a number that is sequentially assigned to each cluster, and then this is followed by a sequence of nodes in the cluster. Figure 2 shows a DSM clustering arrangement and the corresponding model description. It is easily seen that the length of this model description is as follows:

$$\sum_{i=1}^{n_c}\left(\log n_n + cl_i \cdot \log n_n\right), \quad (1)$$

where n_c is the number of clusters in the DSM, n_n is the number of nodes, cl_i is the number of nodes in the i^{th} cluster, and the logarithm base is 2. In the example of Figure 2, $n_c = 2$ clusters, n_n=8 nodes, $cl_1 = 3$, and $cl_2 = 4$. The table in the figure reads as follows: "*cluster 1 has 3 nodes: B, D, and G; cluster 2 has 4 nodes: A, C, E, and H.*"

In Figure 2, If n_n and n_c are known, it is not difficult to see that the above model description is uniquely decodable. When n_n is given, and assuming $n_c \le n_n$, then $\log n_n$ bits are needed to describe n_c. The $\log n_n$ bits are fixed for all models, and therefore they are omitted without loss of accuracy.

Mismatched Data Description

Based on the model, we first construct another matrix (call it *DSM'*), where each entry d'_{ij} is "1" *if and only if:* (1) some cluster contains *both* node *i* and node *j* simultaneously, or (2) the bus contains *either* node *i* or node *j*. Then, we compare d'_{ij} with the given d_{ij}. For every mismatched entry, where $d'_{ij} \neq d_{ij}$, we need a description to indicate where the mismatch occurred (*i* and *j*) and one additional bit to indicate whether the mismatch is zero-to-one or one-to-zero. Define the following two mismatch sets: $S_1 = \{(i,j) \mid d_{ij} = 0, d_{ij}' = 1\}$ and $S_2 = \{(i,j) \mid d_{ij} = 1, d_{ij}' = 0\}$. We call the mismatch that contributes to S_1 the *type 1 mismatch* and the mismatch that contributes to S_2 the *type 2 mismatch*. The mismatched data description length is given by:

$$\sum_{(i,j)\in S_1}\left(\log n_n + \log n_n + 1\right) + \sum_{(i,j)\in S_2}\left(\log n_n + \log n_n + 1\right) \quad (2)$$

The first $\log n_n$ in the bracket indicates *i*, the second one indicates *j*, and the additional one bit indicates the type of mismatch.

MDL Clustering Metric

The MDL clustering metric is given by the weighted summation of the model description length and mismatched data description given above. With some arithmetic manipulations, the metric can be written as follows:

$$f_{DSM}(M) = (1-\alpha-\beta)\cdot\left(n_c \log n_n + \log n_n \sum_{i=1}^{n_c} cl_i\right), \\ +\alpha\cdot[\,|S_1|(2\log n_n+1)]+\beta\cdot[\,|S_2|(2\log n_n+1)] \quad (3)$$

where α and β are weights between 0 and 1. Here we use weighting in order to match the preference of human experts. This becomes more evident in the case study. The α and β setting is not a simple task. A naïve setting is $\alpha = \beta = 1/3$. Later in the chapter, we adjust α and β to mimic the behavior of a manual clustering arrangement.

Finally, the objective is to find a model *M* that minimizes f_{DSM}. In other words, f_{DSM} is the length (in bits, when the logarithm is taken in base 2) that model *M* needs to describe the given data.

From Binary DSM to Weighted DSM

Most real-world DSMs are real valued or contain information of different levels of interaction strength. Therefore, it is necessary to extend our algorithm to be capable of clustering weighted DSMs. If we normalize a weighted DSM so that every entry in the DSM is between 0 and 1, the value of each entry can be considered as the probability of communication. This is consistent with binary DSMs. In a binary DSM, $d_{ij} = 1$ can be thought of as that node i communicates with node j with probability 1. The same interpretation is also valid for $d_{ij} = 0$. Based on the interpretation, we can modify the MDL scoring metric as follows. First, the entries d_{ij} in the DSM is normalized to $p_{ij} = (d_{ij} - d_{\min})/(d_{\max} - d_{\min})$, where $d_{\max} = \max_{i,j} d_{ij}$ and $d_{\min} = \min_{i,j}$. The formula of the description length of model complexity remains the same. The mismatch sets for type 1 and type 2 are modified as $S_1 = \sum_{d_{ij}'=1}(1-p_{ij})$ and $S_2 = \sum_{d_{ij}'=0} p_{ij}$, respectively. The modification is so because entry (i, j) has a probability $(1-p_{ij})$ to be a type 1 mismatch if it is inside a cluster, and a probability p_{ij} to be a type 2 mismatch if it is outside clusters.

THE PROPOSED CLUSTERING ALGORITHM

To use a GA with the MDL clustering metric, an encoding method that encodes a clustering arrangement into a chromosome is needed. As indicated above, the encoding should deal with overlapping clusters and three-dimensional structures. As long as the encoding allows that a node belongs to several different clusters, the GA is able to detect overlapping clusters and three-dimensional structures.

The chromosome is a binary string of $(n_c \cdot n_n)$ bits, where n_c is predefined maximal number of clusters and n_n is the number of nodes. The $(x+n_n \cdot y)$-th bit represents that node $(x+1)$ belongs to cluster $(y+1)$. The last cluster is treated as a bus. For example, in Figure 2, $n_n = 8$, and given n_c is 3, then the model can be described by the following chromosome shown in Figure 3. When manipulated, the chromosome is transformed into a binary string which is a concatenation of all rows.

With the above encoding scheme, now it is sufficient to apply the proposed MDL clustering metric to a GA to create a DSM clustering

Figure 3. A chromosome that represents the model shown in the lower part of Figure 7; the binary string manipulated is 010100101010100100000000

Node	A	B	C	D	E	F	G	H
Cluster 1	0	1	0	1	0	0	1	0
Cluster 2	1	0	1	0	1	0	0	1
Bus	0	0	0	0	0	0	0	0

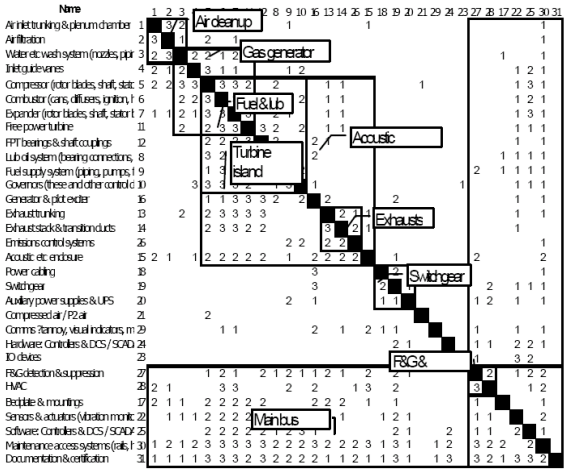
algorithm. At the beginning, a population of encodings of DSM clustering arrangements is randomly initialized. The MDL clustering metric is then applied to each DSM clustering arrangement, and the description length of each DSM clustering arrangement is obtained. With the evaluations, a GA with (λ+ μ) selection, uniform crossover, and simple mutation searches the DSM clustering arrangement with a minimal description length.

The parameters α and β in equation 3 can be tuned by the use of Widrow-Hoff iteration (or Delta rule) (Widrow & Hoff, 1960), so that the GA results would have similar ratios for the description length as human experts. For more details, refer to Yu, Yassine, and Goldberg (2004).

CASE STUDIES

In this section, the proposed DSM clustering algorithm is tested on a real-world DSM for a 10 MWe gas turbine (Sharman & Yassine,

Figure 4. Manual clustering of the gas turbine with named clusters (the numbers 1, 2, and 3 in the figure represent varying dependency strengths)

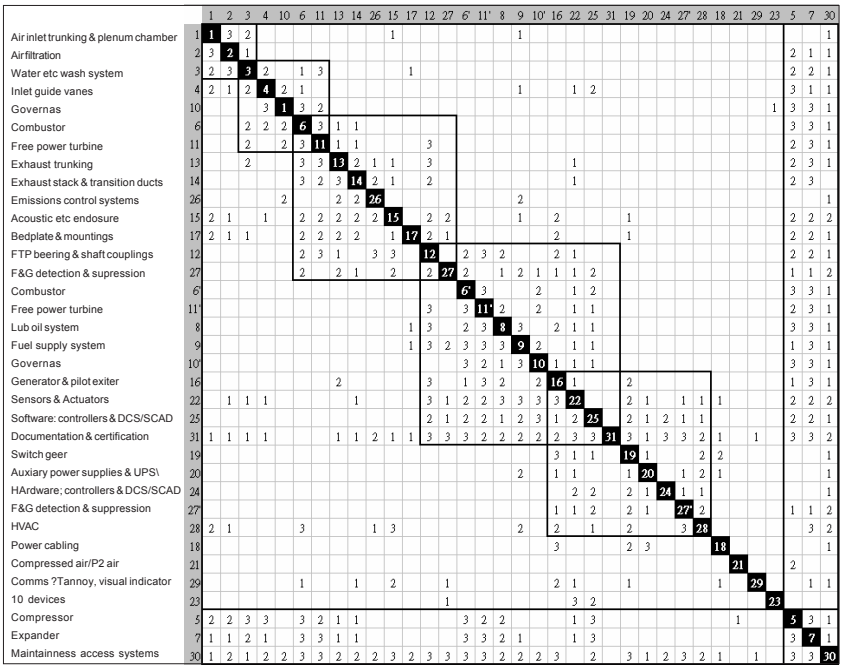


2004). The DSM for a generic 10 MWe gas turbine-driven electrical generator set was constructed by decomposing it into 31 sub-systems. The sub-systems initially were listed randomly in the DSM and then tick marks denoting material relationship from one sub-system to another were inserted. Figure 4 shows an attempt at manually clustering the DSM (Sharman & Yassine, 2004). This took a few manual changes to the order of elements in the initial DSM, which revealed the clusters marked in the figure. After inspection of the clusters, they were given names to identify them. Readers are referred to Yu et al. (2004) for more real-world DSM case studies.

Figure 5. Results of the Widrow-Hoff iterations—the first row is the objective ratios of description lengths; the second row shows the resulted ratios and the weights that produce such ratios.

	w_1	w_2	w_3	**Description length ratios**		
Objective				0.0784	0.8116	0.1102
Results	0.4533	0.1228	0.4239	0.0729	0.8154	0.1117

Figure 6. Clustering arrangement by the proposed GA for the gas turbine DSM



Automated Clustering Using the Proposed MDL-GA

By inspecting several expert-clustered DSMs in Yu et al. (2004), the average ratios of the description lengths of the model, type 1 mismatch, and type 2 mismatch are set as the average of 0.0784: 0.8116:0.1102. The maximal number of clusters is set to half of the number of nodes. Hence the length of the chromosomes for the DSM for the 10 MWe gas turbine is 15×31=465. Crossover probability is set to one, and mutation probability is set to one over the chromosome length. The GA is terminated if there is no improvement in 50 generations. A set of experiments showed that (4250+4250) selection produces satisfactory results. Since the number of Widrow-Hoff iterations is limited to 10, after 10 iterations the best run is chosen according to the minimal sum of squared errors. The objective ratios and the experimental ratios obtained from the 10 Widrow-Hoff iterations are shown in Figure 5. Finally, applying the proposed MDL-GA clustering algorithm resulted in Figure 6.

Discussion of Results

Figures 7 and 8 show the clustering arrangements, mismatches, and description length performed by human experts manually and the proposed MDL-GA algorithm. According to

Figure 7. Comparison of the clustering arrangements given by human experts and the proposed GA

	n_c	cl_i	$\|S_1\|$	$\|S_2\|$
Manual	8	2, 3, 3, 3, 6, 7, 8, 13	342.33	17.67
GA	6	3, 3, 5, 9, 9, 11	233.67	32.00

Figure 8. Description length and mismatches of the DSM clustering arrangement done by human experts vs. GAs

Description Length	Model	Type 1 Mismatch	Type 2 Mismatch	Total Weighted Description Length
Manual	262.57	3897.93	192.71	3205.38
GA	227.89	2548.93	349.07	2125.05

Figure 8, the proposed MDL-GA clustered the gas turbine DSM with a less complex model and fewer mismatches, and the MDL-GA gave a shorter total description length. If the MDL-GA clusters the DSMs using the specific weights tuned for the individual case, then it will find a better arrangement (i.e., less complex models and fewer mismatches), as depicted by the results of Figure 9. In this case, the MDL-GA gave shorter description lengths in all three categories: model, type 1 mismatch, and type 2 mismatch.

Furthermore, according to Figure 8 we can see that the magnitude of type 1 mismatch is always larger than type 2 mismatch. In other words, human experts tend to endure type 1 mismatch (i.e., inside clusters) more than type 2 mismatch (i.e., between clusters). This observation is intuitive if we consider that system engineers tend to pay more attention to minimizing interactions between clusters (i.e., product modules or design teams) than interaction patterns inside a cluster. Therefore, an automated algorithm that would mimic human clustering would be biased in the same direction.

Figure 9. Comparison of the clustering arrangements given by human experts and the proposed GA using the weights according to the respective human clustering arrangement; it is worth noting that the results given by the GA dominate in all three categories.

Description Length	Model	Type 1 Mismatch	Type 2 Mismatch
Manual	262.57	3897.93	192.71
GA	208.72	3421.60	174.53

These results can be interpreted by two points of view. If the MDL is a more appropriate criterion for the clustering problem, then the GA provides better solutions than humans. On the other hand, if human clustering is more appropriate due to considering several subtle constraints that were not observed by the GA, then the problem is how to "tune" the MDL-metric to mimic human experts' preference. As an initial attempt, we have accomplished that by tuning the weights (w_1, w_2, w_3) according to the method described in this chapter. However, our point of view is that the MDL-GA need not provide better results than human expert clustering, but must have the ability to devise an automated algorithm capable of producing competitive clustering arrangements aligned with human expertise. The proposed method provides a consistent, systematic, and automatic way to cluster DSMs, and the clustering results can be either used directly, or used as an initial clustering arrangement for human experts to tune.

DEPENDENCY STRCTURE MATRIX GENETIC ALGORITHM

The modularity among people in an organization suggests an optimal teaming and hence minimizes the coordinate cost in the organization. The modularity among product components of a product line suggests an optimal parallelism and hence maximizes the benefit for the product line. Similarly, the modularity among decision variables of an optimization problem suggests an optimal problem decomposition and hence ensures the optimization problem can be solved *quickly, reliably,* and *accurately* (Goldberg, 2002).

In this section, we develop a technique for extracting building block information by the DSM clustering technique presented in previous sections called a dependency structure matrix genetic algorithm (Yu, Goldberg, Yassine, & Chen, 2003). DSMGA is able to properly decompose the optimization problem with the help of DSM clustering and then solves the problem by BB-wise crossover.

Description of DSMGA

This section constructs DSMGA. The DSMGA consists of three main tasks: constructing a DSM representing interactions between pairs of genes, clustering the DSM to obtain BB information, and perform BB-wise crossover by exploiting the BB information. This section describes the framework of the DSMGA and how to accomplish the three tasks mentioned above.

The Framework of the DSMGA

There are two levels of evolutionary algorithms in the DSMGA: one is to solve the given problem, called the *primary GA,* and other one is to solve the building block identification problem, called the *auxiliary GA*. The basic idea of the DSMGA is to use the auxiliary GA to identify BBs, then the primary GA solves the optimization problem by utilizing the BB information.

Basically, the auxiliary GA receives a DSM which represents the dependencies between genes from the primary GA. After solving the DSM cluster problem, the auxiliary GA returns the BB information back to the primary GA, and the primary GA utilizes the BB information to accomplish BB-wise crossover. More details about the DSM construction and the auxiliary GA can be found in following subsections. Additionally, unlike a meta-GA (Mercer & Sampson, 1978), the auxiliary GA in the DSMGA does not consume function evaluations of the given problem. Instead, the auxiliary GA uses only the MDL clustering metric, f_{DSM} (equation 3), which is much less computationally expensive than many real-world objective functions.

DSM Construction

The way that the DSMGA detects the dependency of gene *i* and gene *j* is similar to LINC and LIMD (Munetomo & Goldberg, 1999). Define $f(a_i = x, a_j = y)$ as the fitness value of the schema where the *i*-th gene is *x*, the *j*-th gene is *y*, and the rest are * (wild card). For example, for *i*=2 and *j*=5 in a 5-bit problem, $f(a_i = 0, a_j = 1) = f(*0**1)$. If the *i*-th gene and the *j*-th gene are independent (linear), $f(a_i = 0, a_j = 1) - f(a_i = 0, a_j = 0)$ and $f(a_i = 1, a_j = 1) - f(a_i = 0, a_j = 0)$ should be the same. Therefore, the interaction (nonlinearity) between the *i*-th gene and the *j*-th gene is defined as:

$$\Delta_{ij} = | f(a_i = 0, a_j = 1) - f(a_i = 0, a_j = 0) - f(a_i = 1, a_j = 1) + f(a_i = 1, a_j = 0) | \quad (4)$$

However, the fitness value of schemata cannot be computed unless every possible combination is visited. Now the task is to approxi-

mate Δ_{ij} with s_{ij}, which is computed based on the individuals seen so far. First, define the sampled fitness of a schema in the population of t-th generation as

$$f(a_i = x, a_j = y)^t = \frac{1}{n_a} \sum_{a \in P_t, a_i = x, a_j = y} f(a),$$

where t is the generation, P_t is the population of the t-th generation, f is the fitness function, a is an individual where its i-th gene is x and j-th gene is y, and n_a is the number of such a in the population. We call $f(a_i = x, a_j = y)^t$ *undefined* if n_a is zero. The information of interactions gathered from the population is

$$s_{ij}^t = | f(a_i = 0, a_j = 1)^t - f(a_i = 0, a_j = 0)^t - f(a_i = 1, a_j = 1)^t + f(a_i = 1, a_j = 0)^t |$$

s_{ij}^t is *undefined* if *any* of the $f(a_i, a_j)^t$ are *undefined*.

To utilize all individuals of all populations seen so far, s_{ij}^t is then averaged over generations. Defining a set $D = \{s_{ij}^t \mid s_{ij}^t \text{ is defined.}\}$, the average of s_{ij}^t is expressed as

$$s_{ij}^t = \frac{1}{|D|} \sum_{t=1, s_{ij}^t \in D}^{t=T} s_{ij}^t,$$

where T is the current generation. With a threshold θ, s_{ij}^t is then transferred into 0-1 domain, and a DSM is constructed:

$$d_{ij} = \begin{cases} 0, \text{if } s_{ij} \le \theta. \\ 1, \text{if } s_{ij} > \theta. \end{cases} \quad (5)$$

The threshold is calculated by the k-mean algorithm (Hartigen & Wong, 1979) by setting k=2. The threshold can also be set according to some prior knowledge if available.

BB-Wise Crossover

After the BB information is obtained from the auxiliary GA, BB-wise crossover can be achieved in the primary GA level. The BB-wise crossover is like an ordinary allele-wise crossover, but instead of mixing genes, the BB-wise crossover mixes BBs and will not disrupt BBs. The order of BBs is not significant because BBs have no (or little) interaction with each other by definition. For more details about BB-wise crossover, refer to Thierens and Goldberg (1994).

Empirical Results

The experiments were done by using a simple GA (SGA) as a test platform. The DSMGA tested here is a simple GA with the DSM construction and an auxiliary GA with the MDL clustering metric. The test function is a 30-bit MaxTrap problem composed of 10 three-bit trap functions. The three-bit trap is given by:

$$f_{trap^3} = \begin{cases} 0.9, & \text{if } u = 0 \\ 0.45, & \text{if } u = 1 \\ 0, & \text{if } u = 2, \\ 1.0, & \text{if } u = 3 \end{cases} \quad (6)$$

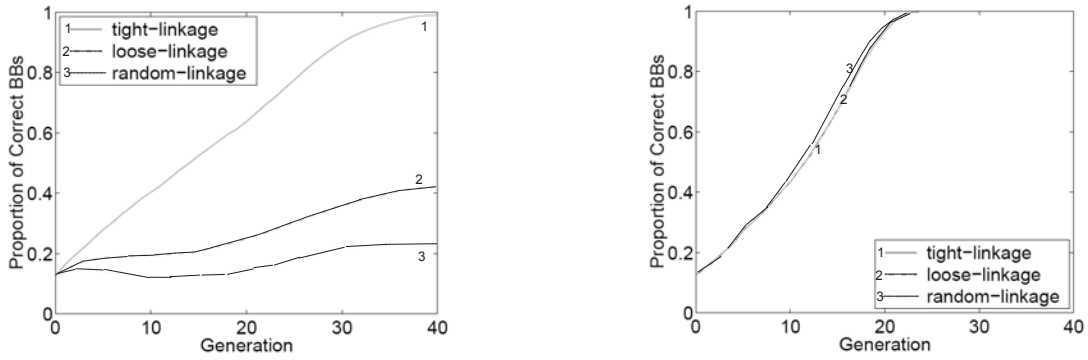
where u is the number of 1s among the three bits.

Three linkage cases were tested: tight linkage, loose linkage, and random linkage. Define $U(x)$ as a counting function that counts the number of 1s in x. In the tight linkage test, genes are arranged as

$$fitness = f_{trap^3}(U(x_1, x_2, x_3)) + f_{trap^3}(U(x_4, x_5, x_6)) + f_{trap^3}(U(x_7, x_8, x_9)) + \cdots$$

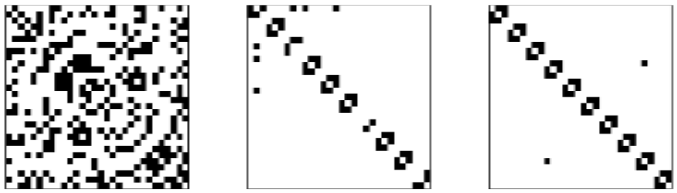
On the other hand, in the loose linkage test case, genes are arranged as

Figure 10. (a) The performance of the SGA with two-point crossover; (b) The performance of the DSMGA with BB-wise two-point crossover



(a) *(b)*

Figure 11. The DSMs created by the DSMGA in the tight linkage test: from the left to the right, the DSMs are created at generation 0, 5, and 10, respectively; the perfect result should be 10 three-bit clusters on the diagonal



$$fitness = f_{trap^3}(U(x_1, x_{11}, x_{21})) + f_{trap^3}(U(x_2, x_{12}, x_{22})) + f_{trap^3}(U(x_3, x_{13}, x_{23})) + \cdots$$

In the random linkage test case, genes are simply arranged randomly. For more details about this test scheme, see Goldberg et al. (1989).

Given the failure rate to be 1/10, the population size is set as 182 by the gambler's ruin model (Harik, Cantú-Paz, Goldberg, & Miller, 1997; Miller, 1997). In the primary GA, binary tournament selection was adopted, and no mutation was used. In the auxiliary GA, the maximal number of cluster n_c is 10 (equal to m), and the mutation probability p_m was set to be 1/30. A ($\lambda+\mu$) selection was adopted, where λ=5 and λ=500.

Figure 10(a) shows the performance of the SGA using two-point crossover. The SGA worked only for the tight linkage case. For loose and random linkage cases, SGA did not work because of BB disruption. Correspondingly, Figure 10(b) illustrates the performance of the DSMGA using BB-wise two-point crossover. The DSMGA converged for all three tests. Even in the tight linkage test, the DSMGA (converged at the 22nd generation) outperformed the SGA (converged at the 40th generation) because DSMGA disrupted fewer BBs. This

argument can be verified by Figure 11, which shows the DSM created by the DSMGA for the tight linkage case. The prefect result is 10 three-bit clusters located on the diagonal. As the figure indicates, at the fifth generation, the DSMGA was able to identify eight BBs; the DSMGA has successfully identified all BBs at the tenth generation.

CONCLUSION

The chapter started with a delicious circle. We sought to discover modularity with the aid of a genetic algorithm, and at the same time we sought to improve genetic algorithms through the discovery of modularity. Although circular reasoning sometimes leads to logical inconsistency, here a stepwise process led to success on both counts. In this chapter, we first presented the concept of DSMs. Then we introduced the MDL concept and used it as a metric for the proposed clustering objective function. The MDL-based metric was then used with a simple GA to cluster weighted graphs or their corresponding DSMs. We applied the MDL-GA to a real-world problem—a DSM of a 10 MWe gas turbine. The results were compared to manual clustering to show the promise of automated clustering using GAs, and the parameters can be tuned to mimic expert clustering arrangements. Finally, the DSM clustering technique was utilized to develop a competent GA—DSMGA. A series of experiments showed that DSMGA is capable of properly decomposing the optimization problem and is much more powerful than a simple GA.

The DSM is a powerful tool for representing and visualizing product architectures. This representation allows for the analysis and development of modular products by clustering the DSM. Clustering algorithms are scarce and inefficient, especially when confronted with complex product architectures as in the reported case studies. The MDL-GA clustering algorithm presented in this chapter was capable of identifying complex clustering arrangements and overcoming many of the difficulties observed in previous clustering tools. The proposed DSM clustering method has the following unique features: (1) it accounts for bus modules, (2) it allows overlapping modules, (3) it is specifically designed to overcome DSM manual/human clustering problems, (4) it has a unique information theoretic clustering measure, (5) it has the tuning capability to mimic human experts' clustering, and (6) it allows tuning of GA parameters for specific types of products by human experts (i.e., different parameters/weights for different products), then the tuned algorithm (i.e., particular weights) can be used repeatedly by others (e.g., non-experts) for similar products or future generations of the same product. As future work, several possible steps can be taken to further refine the proposed method. These efforts include an extension to multi-objective clustering, where the values of entries in the DSM represent different types of dependencies between two nodes (Schloegel, Karypis, & Kumar, 1999). Along similar lines, a more explicit representation of domain-specific expert knowledge may allow for better tuning of the weights of the model description, and more experiments (i.e., case studies) might be needed to find the real preference of human experts (Nascimento & Eades, 2001). Furthermore, the proposed method is capable of identifying buses and overlapped clusters, but other predominant architectural features may also need to be identified and incorporated into the MDL clustering metric. One such example is the concept of a mini-bus or a floating bus discussed in Sharman and Yassine (2004).

Interestingly, the MDL-DSM combination also led us to investigate the dual problem of

using DSM clustering to design a more effective genetic algorithm—DSMGA (Yu et al., 2003). DSMGA utilizes the DSM clustering technique to identify BBs. An MDL clustering metric used by the auxiliary GA was constructed for the DSM clustering problem. Empirical results have shown that using the BB information obtained by the DSM clustering technique can help the convergence of GAs on tight, loose, and random linkage problems. Just as the DSM clustering results can be reused in organization, the BB information obtained from DSM clustering can also be used in similar optimization problems. In other words, DSMGA creates a specific crossover operator for the given problem. For more information of such reusability, refer to Yu and Goldberg (2004). Furthermore, DSMGA can be readily extended to solve hierarchical problems (Pelikan & Goldberg, 2001), and can also be combined with principled efficiency-enhancement techniques such as fitness inheritance (Sastry, Pelikan, & Goldberg, 2004) and time continuation (Lima, Sastry, Goldberg, & Lobo, 2005).

There are many examples in this volume of nature-inspired computing coming to the aid of management and commerce, and this chapter is certainly to be counted among their number. The use of a genetic algorithm and an MDL metric to improve our understanding of organizational and product design is no mean feat; however, this chapter took the additional step of using ideas from genetic algorithms and organizational theory to improve the toolkit of nature-inspired computation itself.

We believe that this two-way street can be profitably exploited more often than it currently is because natural computation and organizations both are so intricately tied to successful adaptation and innovation. We invite other researchers to join this quest for both metaphorical and mechanical inspiration going forward.

REFERENCES

Andrews, J. (1998). Bacteria as modular organisms. *Annual Review of Microbiology, 52,* 105-126.

Alon, U. (2003). Biological networks: The tinkerer as engineer. *Science, 301,* 1866-1867.

Altus, S., Kroo, I., & Gage, P. (1996). A genetic algorithm for scheduling and decomposition of multidisciplinary design problems. *Transactions of the ASME, 118,* 486-489.

Baldwin, C., & Clark, K. (2000). *Design rules: The power of modularity.* Cambridge, MA: MIT Press.

Barron, A., Rissenen, J., & Yu, B. (1998). The MDL principle in coding and modeling. *IEEE Transactions on Information Theory, 44*(6), 2743-2760.

Bosman, P., & Thierens, D. (1999). Linkage information processing in distribution estimation algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference 1999* (Vol. 1, pp. 60-67).

Eppinger, S. D., Whitney, D. E., Smith, R., & Gebala, D. (1994). A model-based method for organizing tasks in product development. *Research in Engineering Design, 6*(1), 1-13.

Fernandez, C. (1998). *Integration analysis of product architecture to support effective team co-location.* Master's thesis, Massachusetts Institute of Technology, USA.

Fixson, S. (2003). *The multiple faces of modularity—a literature analysis of a product concept for assembled hardware products.* Technical Report 03-05, Industrial & Operations Engineering, University of Michigan, USA.

Fodor, J. (1996). *The modularity of mind.* Cambridge, MA: MIT Press.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* New York: Addison-Wesley.

Goldberg, D. E., Korb, B., & Deb, K. (1989). Messy genetic algorithms: motivation, analysis, and first results. *Complex Systems, 3,* 493-530.

Goldberg, D. E., Deb, K., Kargupta, H., & Harik, G. (1993). Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. *Proceedings of the 5th International Conference on Genetic Algorithms,* (pp. 56-64).

Goldberg, D. E. (2002). *The design of innovation: Lessons from and for competent genetic algorithms.* Boston: Kluwer Academic.

Gonzalez-Zugasti, J., Otto, K. & Baker, J. (2000). A method for architecting product platforms. *Research in Engineering Design, 12,* 61-72.

Guimerà, R., Danon, L., Diaz-Guilera, A., Giralt, F., & Arenas, A. (2003). Self-similar community structure in a network of human interactions. *Physical Review E, 68,* 065103(R).

Harik, G., & Goldberg, D. E. (1996). Learning linkage. *Foundations of Genetic Algorithms, 4,* 247-262.

Harik, G., Cantú-Paz, E., Goldberg, D. E., & Miller, B. L. (1997). The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation* (pp. 7-12).

Harik, G. (1999). *Linkage learning via probabilistic modeling in the ECGA.* IlliGAL Report No. 99010, University of Illinois at Urbana-Champaign, USA.

Hartigen, J. A., & Wong, M. A. (1979). A k-means clustering algorithm. *Applied Statistics, 28,* 10-108.

Hartwell, L., Hopfield, J., Leibler, S., & Murray, A. (1999). From molecular to modular cell biology. *Nature, 402,* C47-C52.

Holland, J. H. (1975). *Adaptation in natural and artificial systems.* Ann Arbor: University of Michigan Press.

Ishii, K., & Yang, T. (2003). Modularity: International industry benchmarking and research roadmap. *Proceedings of the ASME Design Engineering Technical Conference 2003* (DETC2003/DFM-48132).

Kargupta, H. (1996). The performance of the gene expression messy genetic algorithm on real test functions. *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation* (pp. 631-636).

Lima, C. F., Sastry, K., Goldberg, D. E., & Lobo, F. G. (2005). Combining competent crossover and mutation operators: A probabilistic model building approach. *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO 2005) (pp. 735-742).

Lutz, R. (2002). Recovering high-level structure of software systems using a minimum description length principle. *Proceedings of AICS-02* (pp. 61-69). Berlin: Springer-Verlag (LNAI 2464).

McCord, K., & Eppinger, S. (1993). *Managing the integration problem in concurrent engineering.* Working Paper 3594, MIT Sloan School of Management, USA.

Mercer, R. E., & Sampson, J. R. (1978). adaptive search using a reproductive meta-plan. *Kybernetes, 7,* 215-228.

Miller, B. L. (1997). *Noise, sampling, and efficient genetic algorithms.* Doctoral dissertation, University of Illinois at Urbana-Champaign, USA.

Moore, G. (1999). *Crossing the chasm: Marketing and selling high-tech products to mainstream customers*. New York: HarperBusiness.

Munetomo, M., & Goldberg, D. E. (1999). Identifying linkage groups by nonlinearity/non-monotonicity detection. *Proceedings of the Genetic and Evolutionary Computation Conference 1999* (Vol. 1, pp. 433-440).

Nascimento, H. A. D., & Eades, P. (2001). Interactive graph clustering based upon user hints. *Proceedings of the 2nd International Workshop on Soft Computing Applied to Software Engineering,* Enschede, The Netherlands.

Newman, M., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E, 69.*

Pelikan, M., & Goldberg, D. E. (2001). Escaping hierarchical traps with competent genetic algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-2001) (pp. 511-518).

Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. (1999). BOA: The Bayesian Optimization Algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-99) (pp. 525-532).

Pimmler, T., & Eppinger, S.D. (1994). Integration analysis of product decompositions. *Proceedings of the ASME Design Theory and Methodology* (DTM '94) (pp. 343-351).

Rissanen, J. (1978). Modeling by shortest data description. *Automatica, 14,* 465-471.

Rissanen, J. (1999). Hypothesis selection and testing by the MDL principle. *Computer Journal, 42,* 260-269.

Sanchez, R., & Mahoney, J. (1996). Modularity, flexibility, and knowledge management in product and organization design. *Strategic Management Journal, 17,* 63-76.

Sastry, K., Pelikan, M., & Goldberg, D. E. (2004). Efficiency enhancement of genetic algorithms via building-block-wise fitness estimation. *Proceedings of the IEEE Conference on Evolutionary Computation* (Vol. 1, pp. 19-23).

Schilling, M. A. (2002). Modularity in multiple disciplines. In R. Garud, R. Langlois, & A. Kumaraswamy (Eds.), *Managing in the modular age: Architectures, networks and organizations* (pp. 203-214). Oxford, UK: Blackwell.

Schloegel, K., Karypis, G., & Kumar, V. (1999, August/September). A new algorithm for multi-objective graph partitioning. *Proceedings of the European Conference on Parallel Processing,* Toulouse, France (pp. 322-331).

Sharman, D., & Yassine, A. (2004). Characterizing complex product architectures. *Systems Engineering Journal, 7*(1), 35-60.

Smith, J. (2002). On appropriate adaptation levels for the learning of gene linkage. *Journal of Genetic Programming and Evolvable Machines, 3,* 129-155.

Steward, D. V. (1981). The design structure system: A method for managing the design of complex systems. *IEEE Transactions on Engineering Management, 28,* 77-74.

Stone, R., Wood, K., & Crawford, R. (2000). A heuristic method for identifying modules for product architectures. *Design Studies, 21*(1), 5-31.

Sullivan, K., Griswold, W., Cai, Y., & Hallen, B. (2001). The structure and value of modularity in software design. *ACM SIGSOFT Software Engineering Notes, 26*(5), 99-108.

Thierens, D., & Goldberg, D. E. (1994). Convergence models of genetic algorithm selection

schemes. *Parallel Problem Solving from Nature, III,* 119-129.

Thebeau, R. (2001). *Knowledge management of system interfaces and interactions for product development processes.* Master thesis, Massachusetts Institute of Technology, USA.

Ulrich, K., & Eppinger, S. (2000). *Product design and development.* New York: McGraw-Hill.

Whitfield, R., Smith, J., & Duffy, A. (2002, July). Identifying component modules. *Proceedings of the 7th International Conference on Artificial Intelligence in Design* (AID'02), Cambridge, UK.

Widrow, B., & Hoff, M. E., (1960). Adaptive switching circuits. *1960 IRE WESCON Convention Record, 4,* 96-104. New York: IRE. Reprinted in Anderson and Rosenfeld, 1988.

Yassine, A., & Braha, D. (2003). Four complex problems in concurrent engineering and the design structure matrix method. *Concurrent Engineering Research & Applications, 11*(3), 165-176.

Yu, T.-L., Yassine, A., & Goldberg, D. E. (2004). An information theoretic method for developing modular architectures using genetic algorithms. *Research in Engineering Design.*

Yu, T.-L., Goldberg, D. E., Yassine, A., & Chen, Y.-P. (2003). Genetic algorithm design inspired by organizational theory: Pilot study of a design structure matrix-driven genetic algorithm. *Proceedings of Artificial Neural Networks in Engineering 2003* (ANNIE 2003) (pp. 327-332).

Yu, T.-L., & Goldberg, D. E. (2004). Dependency structure matrix analysis: Off-line utility of the dependency structure matrix genetic algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference 2004* (GECCO 2004) (pp. 355-366).

Chapter XXIX
Autonomous Systems with Emergent Behavior

Giovanna Di Marzo Serugendo
University of Geneva, Switzerland

ABSTRACT

This chapter presents the notion of autonomous engineered systems working without central control through self-organization and emergent behavior. It argues that future large-scale applications from domains as diverse as networking systems, manufacturing control, or e-government services will benefit from being based on such systems. The goal of this chapter is to highlight engineering issues related to such systems, and to discuss some potential applications.

INTRODUCTION

Devices from personal computers, to handhelds, to printers, to embedded devices are very widely available. Further, today's wireless network infrastructures make it possible for devices to spontaneously interact. In addition, large-scale communication, information, and computation infrastructures such as networks or grids are increasingly being built using numerous heterogeneous and distributed elements, which practically cannot be placed under direct centralized control. These elements exhibit certain degrees of autonomy and of self-organization, such as taking individual decisions and initiatives, interacting which each other locally, and giving rise to an emergent global behavior.

This chapter introduces first the notion of autonomous systems; second it reviews the notions of decentralized control, self-organization, and emergent behavior, and discusses how they relate to each other. Third, this chapter discusses different issues pertaining to the design and development of autonomous systems with emergent behavior. Fourth, it reviews techniques currently being established for building those systems. Finally, it provides several examples of applications.

AUTONOMOUS SYSTEMS

We distinguish different classes of autonomous systems. First, autonomous systems as *distributed embedded devices* consist of physical devices having some onboard intelligence, and standalone and communication capabilities. Such devices comprise intelligent mobile robots, but also intelligent wearable computing, surveillance, or production devices. Second, from a software point of view, *autonomous agents and multi-agent systems* are a notion first established by the distributed artificial intelligence community. Such systems do not have to cope with the same problems faced with devices situated in a physical environment (e.g., low battery). However, agents provide a metaphor for software design which incorporates most of the elements present in embedded devices such as autonomous decision-taking processes, communication with other agents, and social interactions for collaboration, negotiation, transactions or competition purposes (Wooldridge, 2003). Third, more recently an initial focus has been given from the research community on autonomous software entities interacting with each other in a decentralized self-organized way in order to realize a dedicated high-level functionality (interactions for collaboration purposes), or giving rise to an emergent global behavior as a side effect of their local interactions (interactions for competition purposes). This category of applications or entities is referred to as *self-organizing systems* or *systems with emergent behavior* (Di Marzo Serugendo et al., 2004). In some sense, this last category combines the first two views where autonomous software populates autonomous devices. Fundamental points of these different views of autonomous systems are: the social interactions arising among the different elements, and the need for adaptation to unforeseen (at design time) situations encountered in dynamic environments.

There is currently a growing interest in autonomous applications able to self-manage, not only from academic research but also from the industry. *Ambient intelligence* envisions seamless delivery of services and applications, based on ubiquitous computing and communication. Invisible intelligent technology will be made available in clothes, walls, or cars; and people can freely use it for virtual shopping, social learning, micro-payment using e-purses, electronic visas, or traffic guidance system (Ducatel et al., 2001). Ambient intelligence requires low-cost and low-power designs for computation running in embedded devices or chips, as well as self-testing and self-organizing software components for robustness and dependability. Based on the human nervous system metaphor, IBM's *Autonomic Computing* initiative considers systems that manage themselves transparently with respect to the applications. Such systems will be able to self-configure, self-optimize, self-repair, and protect themselves against malicious attacks (Kephart & Chess, 2003). Recent interest by Microsoft, as part of the *Dynamic Systems Initiative,* indicates as well the importance of self-organization for managing distributed resources.

Autonomous Computation Entities vs. Autonomous Systems

As follows from the discussion above, autonomous systems are composed of one or, more generally, of several autonomous computation entities interacting together. These autonomous computation entities are either embedded into physical, possibly mobile, devices (e.g., in ambient intelligence applications) or part of a given environment that supports their execution and interactions (e.g., multi-agent systems).

DECENTRALIZED CONTROL, SELF-ORGANIZATION, AND EMERGENT BEHAVIOR

Decentralized control is intimately linked with the notion of emergent phenomena, since some result is expected from a system even if it works with decentralized control. Self-organization may occur with or without central control; it is related to whether or not the system takes itself the measures to cope with the environmental changes. Even though artificial systems will certainly have at the same time these three characteristics—decentralized control, self-organization, and emergent phenomena—it is important to distinguish each of them. The purpose of this section is to briefly clarify these concepts, and to establish the links and differences among these three notions.

Decentralized Control

It is important to distinguish between two kinds of artificial systems working with decentralized control: (a) systems built as a large set of autonomous components, pertaining to the same system and providing as a whole expected properties or functions—otherwise stated, we want to build an application with a well-specified functionality, but for complexity reasons, this application is decentralized and made of a large number of autonomous components; and (b) systems composed of a large set of autonomous components, spontaneously interacting with each other for possibly independent or competing reasons. In both cases, autonomous components may be heterogeneous and dynamically joining and leaving the system.

Even though in both cases most of the issues and discussions are similar, the fundamental difference lies in the engineering process that is behind the building of the system. In the first case, the whole system is designed with emergent functionality in mind. Simply stated, a given collaborative team develops the application ensuring that the expected functionality will emerge. In the second case, the different components are produced by different teams, with different purposes in mind, each being concerned with the fact that their component can interoperate with the others. There is no expected global function or properties emerging, even though emergent phenomena will arise in any case from the different local interactions and have a causal effect on the whole system—that is, on the particular behavior of the individual components.

In the first case, the core idea behind building large-scale systems is to have them composed of autonomous individual components working without central control, but still producing as a whole the desired function. Indeed, decentralized control allows: (1) computation and decisions to be distributed among the different components, thus preventing the need for a central powerful computer; (2) the system is more robust since it does not rely on a single node that may fail and crash the whole system; (3) network and CPU resources are better used in the sense that communication does not occur among a dedicated central node and a large number of components, but locally among the whole set of components; and (4) in dynamic systems, where components join and leave the system permanently, decentralized control allows a flexible schema for communication, for example with a neighbor instead of with the central entity.

Self-Organization

There are different definitions of self-organization as observed in the natural world. We will focus here on three of them, essentially to enhance the fact that there are different kinds of self-organizations, and that designers of

such systems must be aware of which kind of self-organization they are considering when building their system (Di Marzo Serugendo, Gleizes, & Karageorgos, 2006).

Stigmergy

The theory of stigmergy, defined by Grassé (1959) in the field of swarms or social insects' behavior, states that coordination and regulation are realized without central control by indirect communication of the insects through their environment. Self-organization results from the behavior (of the insects) arising from inside the system. Otherwise stated, swarms' autonomous components are themselves at the origin of the re-organization of the whole system.

Decrease of Entropy

In the field of thermodynamics, Glansdorff and Prigogine (1971) established that open systems decrease their entropy (disorder) when an external pressure is applied. Self-organization in this case is the result of a pressure applied from the outside. It is interesting to compare self-organization in this case with the swarms' behavior, where the "initiative" of self-organization occurs from within the system.

Autopoiesis

Through biological studies, Varela (1979) established the notion of autopoiesis as the self-maintenance of a system through self-generation of the system's components, as for instance cell reproduction. Self-organization here is still different from the two other examples above. Indeed, autopoiesis applies to closed systems made of autonomous components whose interactions self-maintain the system through generation of system components.

Even though differently stated in the few definitions above, and with a different impact on the way and the reasons why self-organization is produced, we can consider that self-organization is essentially:

The capacity to spontaneously produce a new organization *in case of environmental changes* without external control, provided it is not external.

Indeed, in the case of social insects, environmental changes will cause ants or termites to find new paths of food—that is, change their behavior in order to still be able to feed the colony. In the case of thermodynamics, external pressure changes will cause gas particles to be more or less excited, change their temperature, and so forth, thus reaching a new stable state. Finally, cells or living organisms regenerate the whole system in order to overcome cells' death and to survive in their given environment.

It is interesting to note that new organization of a system may occur with or without central control provided; it is not external.

Emergent Behavior

Literature on *emergence* is abundant and varied ranging from philosophical discussions to operational descriptions. One of the most popular definitions of emergence which captures the essence of the emergent phenomena comes from Holland (1998), who states: "The whole is more than the sum of its parts."

In systems composed of individual autonomous computation entities, we will consider that an emergent phenomenon is essentially (Di Marzo Serugendo et al., 2006):

A structure (pattern, property, or function), not explicitly represented *at the level of the*

individual components (lower level), and which appears *at the level of the system (higher level).*

An additional important point here is that emergent phenomena have a meaning for an *observer* external to the system, but not for the system itself. We distinguish two kinds of emergent phenomena:

- Observed patterns or functions that have *no causal effect* on the system itself. If we consider stones ordered by sea, with time, a kind of classification of the stones occurs. Small, lighter stones are close to the border, while heavy stones are far from it. In this case, this ordering of the stones has no effect at all on the whole system made of the stones and the sea (Castelfranchi, 2001).
- Observed functions that have a *causal effect* on the system. Such functions can be desired or not, but in both cases they have an effect on the system behavior and will cause the individual parts to modify their own behavior.

Artificial systems are composed of a large number of individual components, of autonomous computation entities. During the course of time, a large number of interactions occur among these components, whose ordering, content, and purpose are not necessarily imposed. It becomes then difficult to predict the exact behavior of the system taken as a whole because of the large number of possible nondeterministic ways the system can behave. However, since we have built the system, the individual behavior's components and the local rules governing the system are known; it becomes then *in principle* possible to determine the (emergent) system's behavior. In practice, current techniques or calculations (essentially simulations) are not sufficient and make it almost impossible to determine the result. That is why the result, functions, or properties, are said to be *emergent*.

When Self-Organization Meets Emergence

Due to the fact that in most systems, self-organization, and emergent phenomena are observed simultaneously, there is a natural tendency to consider that self-organization leads to emergent phenomena or that they are intimately linked. As also pointed out by De Wolf and Holvoet (2005), even though not totally wrong, this assumption needs to be clarified.

Self-Organization without Emergent Phenomenon

Self-organization happens without observed emergent phenomenon, essentially when the system works under central control. Indeed, self-organization is the capacity of the system to find a new organization in order to respond to environmental changes. The new organization can be identified under internal central control, and thus the possibly observed new organization is fully deducible from the central entity.

Emergent Phenomenon without Self-Organization

Emergent patterns, such as zebra stripes, have no causal effect on the whole system. There is no reorganization of the stripes or of the cells producing the stripes. Stones ordered by sea do not undertake a self-organization when they are ordered by the sea.

Self-Organization together with Emergent Phenomenon

We consider that in order to have self-organization and emergent phenomenon at the same

time, the considered system should have the following characteristics:

- *Dynamic self-organizing system*: Individual components are "active"; they may have their own objective and carry out their respective tasks.
- The system works with *decentralized control*.
- *Local interactions* occur among the individual components.

Natural or artificial interesting systems usually considered by scientists are those of the last category, where we usually have decentralized control realized under self-organization, leading to emergent behavior.

ISSUES

This section distinguishes five issues related to systems made of autonomous software entities and exhibiting an emergent behavior.

Interactions among Unknown Autonomous Computation Entities

Autonomous software entities interact with their environment and with other generally unknown software entities. Interaction covers both semantic understanding of the functional and non-functional aspects of a peer entity, and interoperability, which encompasses transactions, service delivery, and exchange of information.

Management of Uncertainty

For autonomous entities situated in a dynamic and insecure environment, uncertainty relates to reliability and trustworthiness of both the environment and interacting peer entities. For instance, an autonomous software entity cannot expect to fully rely on the permanent availability of network accesses, capacity, and loads. In addition, a malicious entity can exhibit desirable characteristics, while it has no willingness to realize them; or, even in good faith, an entity can fail to deliver a service because the conditions required for its correct functioning are no longer provided by the environment (software errors or physical failures).

Adaptability to Changing Environment and Changing User Requirements

Autonomous software considered in this chapter are situated in a physical environment mostly composed of wireless devices, for which availability of network access is not fully granted, availability of interacting entities is not permanently granted—devices can freely join or leave an interacting zone or partners, and reduced consumption power conditions may prevent autonomous software residing in wireless devices to perform their computation at their maximum capacity. In addition to changing environment, autonomous software has to adapt its behavior to changing user requirement or under the evolution of business practices. For instance, a personal assistant may change the user's agenda if the user signals some priority activity.

Design and Development

On the one hand, emergent behavior, as observed in nature or among societies, has fundamental properties such as robustness, behavior adaptability, learning through experiences, and complex global behavior arising from simple individual behavior, which software engineers would like to benefit from when building complex and large-scale systems. On the other hand, because of these properties, such systems are difficult to design correctly, and their

behavior, once deployed in a physical environment, is difficult or impossible to predict. This is mostly due to the non-linear nature of the interactions occurring among the different autonomous computation entities forming the autonomous systems—that is, the behavior of the system as a whole is not a linear function of the behavior of the individual autonomous computation entities. At the research level, we are currently witnessing the birth of a brand new software engineering field specifically dedicated to emergent behavior. One of the most delicate points is to ensure that "good" (i.e., expected) properties will actually emerge, while bad (i.e., not expected or not desired) properties will not.

Control of Emergent Behavior

At run-time, control of emergent behavior is an important issue related to artificial self-organizing systems with emergent behavior. Indeed, those systems usually demonstrate adaptability capabilities to changing environmental conditions (due to the ability of the system to reorganize), coupled with emergent phenomena, which by definition is difficult to predict. From an engineering point of view, it becomes crucial to have means, at runtime once the system is deployed and executing in its environment, allowing the control of such systems, such as changing the system's global goal, stopping the system if necessary, and so forth. Solutions for this issue most likely have to be considered at design time already, by for instance incorporating specific features that will be useful for control.

ENGINEERING EMERGENT BEHAVIOUR

This section describes existing design and development techniques (bio-inspired or not), and tools for building autonomous systems with emergent behavior. Bio-inspired techniques usually rely on stigmergy (Bernon, Chevrier, Hilaire, & Marrow, 2006), but we observe other approaches based on capacity fields or on trust-based human behavior (Hassas, Di Marzo Serugendo, Karageorgos, & Castelfranchi, 2006). This section reviews interaction mechanisms among individual autonomous computation entities, middleware computing infrastructure supporting their computations, methodologies and CASE tools for design and development, and formal methods related to self-organization and emergent behavior.

Interaction Mechanisms

When building a self-organizing system or a system with decentralized control, at the lowest level we need to define first the local interactions among the different individual components of the system.

Swarm Intelligence

Swarms, or the stigmergy paradigm, provide a great source of inspiration, especially for fixed and mobile networks systems management such as routing, load balancing, or network security. Ants' behavior has been extensively reproduced in artificial systems through artificial pheromones coordinating the work of mobile robots, or mobile agents. More recently, other swarms' behavior is being considered as well, as for instance spiders (Bourjot, Chevrier, & Thomas, 2003) and bees (Fabrega, Lòpez, & Masana, 2005).

Biology: Cells

Besides swarm behavior, another category of natural mechanisms reproduced in artificial systems concerns mammalian immune sys-

tems, which have mostly been used for network intrusion detection (Hofmeyr & Forrest, 2000).

Human Behavior/Trust

Trust-based systems or reputation systems take their inspiration from human behavior. Indeed, uncertainty and partial knowledge are a key characteristic of the natural world. Despite this uncertainty, human beings make choices and decisions, learn by experience, and adapt their behavior. As mentioned above, uncertainty is an issue when building decentralized open systems.

Most artificial trust-based management systems combine higher-order logic with a proof brought by a requester that is checked at runtime. Those systems are essentially based on delegation, and serve to authenticate and give access control to a requester (Weeks, 2001). Usually the requester brings the proof that a trusted third entity asserts that it is trustable or it can be granted access. Those techniques have been designed for static systems, where an untrusted client performs some access control request to some trusted server. Similar systems for open distributed and decentralized environments have also been realized. The PolicyMaker system is a decentralized trust management system (Blaze, Feigenbaum, & Lacy, 1996) based on proof checking of credentials, allowing entities to locally decide whether or not to accept credentials (without relying on a centralized certifying authority). Eigentrust (Kamvar, Schlosser, & Garcia-Molina, 2003) is a trust calculation algorithm that allows calculating a global-emergent reputation from locally maintained trust values. Recently, more dynamic and adaptive schemas have been defined which allow trust to evolve with time as a result of observation, and allow adaptation of the behavior of entities consequently (Cahill et al., 2003).

Artificial Mechanisms

In addition to the digital pheromone, which is the artificial counterpart of the natural pheromone used by the ants, new electronic mechanisms directly adapted to software applications are being developed. The notion of tags, a mechanism from simulation models, is one of them. Tags are markings attached to each entity composing the self-organizing application (Hales & Edmonds, 2003). These markings comprise certain information on the entity, for example functionality and behavior, and are observed by the other entities. In this case the interaction occurs on the basis of the observed tag. This is useful if applied to interacting electronic mobile devices that do not know each other in advance. Whenever they enter the same space, for example a space where they can detect each other and observe the tags, they can decide on whether they can or cannot interact.

Smart tagging systems are already being deployed for carrying or disseminating data in the fields of healthcare, environment, and a user's entertainment.

For instance, in the framework of data dissemination among fixed nodes, Beaufour, Leopold, and Bonnet (2002) propose a delivery mechanism based on the local exchange of data through smart tags carried by mobile users. Mobile users or mobile devices do not directly exchange smart tags; they only disseminate data to fixed nodes when they are physically close to each other. Data information vehicled by smart tags is expressed as triples, indicating the node being the source of the information, the information value, and a time indication corresponding to the information generation. Smart tags maintain, store, and update this information for all visited nodes. A Bluetooth implementation of these smart tags has been realized in the framework of a vending machine (Beaufour et al., 2002).

In smart tagging systems, data remain structurally simple and understandable by human beings, and do not actually serve as a basis for autonomous local decisions.

Middleware Computing Infrastructures

Besides local interaction mechanisms favoring communication and cooperation among the individual components, for an artificial system we may need computing infrastructures, also called middleware, supporting the chosen mechanisms and acting as the artificial environment for the system's component. For instance, such middleware supports the evaporation of the artificial pheromone, or allows mobile agents to perform their execution or to move from one host to another.

These infrastructures are usually *coordination spaces* providing uncoupled interaction mechanisms among autonomous entities, which asynchronously input data into a shared tuple space and may retrieve data provided by other entities.

The TOTA (Tuples On The Air) environment propagates tuples according to a propagation rule, expressing the scope of propagation and possible content change (Mamei & Zambonelli, 2003). Such a model allows, among other things, the electronic capture of the notion of digital pheromone, deposited in the tuple space and retrieved by other agents. The propagation rule removes the pheromone from the data space once the evaporation time has elapsed.

Alternatively, the Co-Fields (coordination fields) model drives agents' behavior as would abstract force fields (Mamei, Zambonelli, & Leonardi, 2002). The environment is represented by fields that vehicle coordination information. Agents and their environment create and spread such fields in the environment. A field is a data structure composed of a value (magnitude of field) and a propagation rule. An agent then moves by following the coordination field, which is the combination of all fields perceived by the agent. The environment updates the field according to the moves of the agents. These moves modify the fields, which in turn modify the agent's behavior. This model allows representing not only complex movements of ants and birds, but also tasks division and succession.

Anthill is a framework for P2P systems development based on agents, evolutionary programming, and derived from the ant colony metaphor. An Anthill distributed system is composed of several interconnected nests (a peer entity). Communication among nests is assured by ants—that is, mobile agents travel among nests to satisfy requests. Ants observe their environment and are able to perform simple computations (Babaoglu, Meling, & Montresor, 2002).

Methodologies and CASE Tools

Finally, at the highest level, from the designer point of view, it is crucial to rely on a development methodology and tools supporting the different phases of development of systems with emergent behavior. Research in this field is at its infancy, and very few results are available.

The Adelfe (Bernon, Gleizes, Peyruqueou, & Picard, 2002) methodology supports designers in making decisions when developing a multi-agent system exhibiting emergent phenomena, and in helping developers in the design of the multi-agent system. It is based on the AMAS (adaptive multi-agent system) theory where self-organization is achieved through cooperation among the agents—that is, agents avoid non-cooperative situations.

Ongoing research in this field seems to favor solutions combining formal or traditional models with simulations in order to be able, on the one hand, to formally define the system, its expected properties, and the behavior of the individual components, and on the other hand (through simulation of these models), to be able to validate or invalidate the design and to predict some emergent phenomena.

Models and Formal Specifications

Mathematical equations, cellular automaton, and neural networks have long been used to understand complex systems, emergent patterns, and the human neuronal activity. More recently, since autonomous software agents naturally play the role of individual autonomous computation entities, multi-agent systems are also being used to model complex systems, and to derive, through simulation, results about emergent phenomena, adaptability characteristics, starting conditions, parameters, and so on. The purposes of multi-agent-based models are of two different natures. On the one hand, the related simulations serve as experiments to better understand a complex system or to (in)validate a given theory—a purpose similar to that pursued with cellular automaton and neural networks models. On the other hand, for artificial systems essentially, agent-based simulations help predict the run-time behavior of a given system, tune the different parameters, and so on. Multi-agent systems are particularly interesting when considering artificial systems with emergent behavior, since those systems involve mobility, social interactions (negotiation, competition, and collaboration), and a high number of entities interacting in a networked environment. The combination of all these features can be hardly modeled through mathematical models, cellular automaton, or neural networks. For the same reasons, a third purpose of the use of multi-agent systems is currently being investigated; it consists of actually building artificial self-organizing applications with autonomous software agents.

In addition to models and simulations, formal reasoning about adaptability characteristics and emergent properties is a research area under consideration in the field of engineering of systems with emergent behavior. As is the case for software engineering related to traditional software, formal specifications allow deriving formal models on which reasoning of different kinds can be performed in order to provide design-time results about the system seen as a whole. We can distinguish different works depending on the use and purpose of the formal specifications. From a very abstract perspective, category theory has proved useful for reasoning about emergent properties arising among interacting components, those properties being expressed through an underlying logic (Fiadeiro, 1996). From a more concrete point of view, recent work has shown interest in emergent properties related to multi-agent systems (Zhu, 2005). In addition to the use of formal specifications and reasoning at design time, we can mention as well the use of formal specifications at run-time and its expected benefits for both designing and controlling (at run-time) emergent behavior (Di Marzo Serugendo, 2005).

APPLICATIONS

This section presents several application domains, current and undergoing realizations, as well as some visionary applications: networking, manufacturing, or cultural applications based on stigmergy and swarm-like systems, and self-managing global computers. Additional descriptions of self-organizing applications can be found in Mano, Bourjot, Lopardo, and Glize (2006).

Networking Systems

Seminal work by Bonabeau, Dorigo, and Théraulaz (1999) describes different types of swarm behavior and explains how to apply it to different applications: ant foraging is useful for routing in communication networks, and consequently for optimization of network-based problems; ant division of labor is useful for task allocation; ants' management of dead bodies is useful for clustering.

T-Man is a generic protocol based on a gossip communication model and serves to solve the topology management problem (Jelasity & Babaoglu, 2005). Each node of the network maintains its local (logical) view of neighbors. A ranking function (e.g., a distance function between nodes) serves to reorganize the set of neighbors (e.g., increasing distance). Through local gossip messages, neighbor nodes exchange or combine their respective views. Gradually, in a bottom-up way, through gossiping and ranking, nodes adapt their list of neighbors, and consequently change and reorganize the network topology. The T-Man protocol is particularly suited for building robust overlay networks supporting P2P systems, especially in the presence of a high proportion of nodes joining and leaving the network.

The SLAC (Selfish Link and behavior Adaptation to produce Cooperation) algorithm (Hales, 2005) favors self-organization of P2P network's nodes into *tribes* (i.e., into specialized groups of nodes). The SLAC algorithm is a selfish re-wiring protocol, where by updating its links with other nodes in order to increase its utility function, a specific node leaves its current tribe and joins a new one. In addition to P2P systems, the SLAC algorithm has many potential applications, for instance to organize collaborative spam/virus filtering in which tribes of trusted peers share meta-information such as virus and spam signatures.

In the field of mobile ad-hoc networks, a self-organized public key management has been defined. The idea is that each node simply carries a subset of the certificates issued by other users. This alleviates the need of centralized certification authorities (Capkun, Buttyan, & Hubaux, 2003).

For intrusion detection and response in computer networks (Foukia, 2005), the immune system serves as a metaphor for detecting intruders, and the stigmergy paradigm is used for responding to the attack. Mobile agents permanently roam the network in order to locate abnormal patterns of recognition. Once an attack is detected, a digital pheromone is released so that the source of attack can be located and a response to the attack can be given. Mobile agents specialized for tracking the source of the attacks are created by the system and act as ants by following the pheromone trail up to the source of the attack.

Manufacturing Control

The stigmergy paradigm serves also for manufacturing control (Karuna et al., 2003). Agents coordinate their behavior through a digital pheromone. In order to fulfill manufacturing orders, they use mobile agents that roam the environment and lay down pheromonal information.

Cultural Heritage

In the field of cultural heritage, a system inspired by bees' behavior has been designed by Fabrega et al. (2005). This system allows independent, non-specialized people to enter information on a given subject. The underlying system then creates new concepts as they are entered into the system by users and correlates together existing concepts. The bees' queen maintains the number and type of bees; it creates new bees whenever a new concept

appears. Different types of bees look for information (nectar), bring that information into cells (honeycomb), validate the information, or look for similarities in honeycombs.

Self-Managing Systems

In order to help human administrators managing large systems, such as distributed operating systems or networks, self-managing systems are being investigated and research efforts are dedicated to these systems.

Expected properties of self-managing systems are to self-configure, self-optimize their parameters, self-repair in case of errors, and to ensure themselves their protection. These characteristics place these systems under the category considered in this chapter. Indeed, such systems work more efficiently without central control; they need to adapt to changes, to reorganize; these systems are dynamic since, for instance, components in error have to leave the system and, being replaced by new ones, updated components have to seamlessly integrate the system (Kephart & Chess, 2003).

However, for self-managing systems considered as autonomous systems with emergent behavior, the situation may be even more complex. Indeed, such systems have three aspects. First, they need to manage themselves; this can be considered a "regular" case of self-organization. Second, in addition to themselves they need to manage any additional resource pertaining to the system. Third, they need to interact with a human administrator; this implies that such a system needs a way to receive global orders from the administrators, and these orders have to be split down into low-level goals or tasks, and conversely, the results or information the self-managing system wants to provide to the human administrator must be coherently packed into a single, meaningful bit of information. Individual components cannot send their respective individual results directly to the administrator.

CONCLUSION

We already observe that technologically advanced societies heavily rely on autonomous devices full of autonomous software (PDAs, mobile phones, portable computers) interacting with each other in a more or less autonomous way. Our vision is that future applications will in fact be composed of autonomous systems organized in a society of devices and software seamlessly interacting together for supporting end users in their everyday life.

We currently observe that artificial systems reproduce natural self-organization principles. They are borrowed from biology, from the social behavior of insects or humans. Different artificial techniques are used for realizing these systems: from indirect interactions, to reinforcement, to adaptive agents, to cooperation, to establishment of dedicated middleware. The interest of self-organization and emergence lies in the natural robustness and adaptation of these systems, and in the relative simplicity of the different components participating to the system. However, it is interesting to notice that, despite any benefit emergence and self-organization can bring to a system, they are not necessarily a good thing. Indeed, in addition to the expected emergent behavior, unexpected emergent behavior will necessarily arise from the different interactions of the system. This behavior will have a causal effect on the system, and especially in the case of self-interested agents, the optimum order (the stable state reached by the reorganization) can actually be bad for individuals or even for everybody. Additionally, current engineering techniques have their limits in terms of control of the emergent behavior, design of the system, and

prediction of the emergent expected or not behavior. Research in this field is still beginning, and much work is needed before any commercial application is widely available for the public.

ACKNOWLEDGMENT

This work is supported by Swiss NSF grant 200020-105476/1.

REFERENCES

Babaoglu, O., Meling, H., & Montresor, A. (2002). Anthill: A framework for the development of agent-based peer-to-peer systems. *Proceedings of the 22nd International Conference on Distributed Computing Systems* (ICDCS '02) (pp. 15-22).

Beaufour, A., Leopold, M., & Bonnet, P. (2002). Smart-tag based data dissemination. *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications* (WSNA'02) (pp. 68-77).

Bernon, C., Chevrier, V., Hilaire, V., & Marrow, P. (2006). Applications of self-organizing multi-agent systems: An initial framework for comparison. *Informatica.*

Bernon, C., Gleizes, M.-P., Peyruqueou, S., & Picard, G. (2002). ADELFE, a methodology for adaptive multi-agent systems engineering. *Proceedings of the Engineering Societies in the Agent World* (ESAW'02) (pp. 156-169).

Blaze, M., Feigenbaum, J., & Lacy, J. (1996). Decentralized trust management. *Proceedings of the IEEE Symposium on Security and Privacy* (pp. 164-173).

Bonabeau, E., Dorigo, M., & Théraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems. Santa Fe Institute studies on the sciences of complexity.* Oxford, UK: Oxford University Press.

Bourjot, C., Chevrier, V., & Thomas, V. (2003). A new swarm mechanism based on social spider colonies: From Web weaving to region detection. *Web Intelligence and Agent Systems, 1*(1), 47-64.

Cahill, V., Shand, E., Gray, N., Dimmock, A., Twigg, J., Bacon, C., English, W., Wagealla, S., Terzis, P., Nixon, C., Bryce, G., Di Marzo Serugendo, J.-M., Seigneur, M., Carbone, K., Krukow, C., Jensen, Y., Chen, Y., & Nielsen, M. (2003). Using trust for secure collaboration in uncertain environments. *IEEE Pervasive Computing Magazine (Special Issue on Uncertainty), 2*(3), 52-61.

Capkun, S., Buttyan, L., & Hubaux, J.-P. (2003). Self-organized public-key management for mobile ad-hoc networks. *IEEE Transactions on Mobile Computing, 2*(1), 52-64.

Castelfranchi, C. (2001). The theory of social functions: Challenges for computational social science and multi-agent learning. *Journal of Cognitive Systems Research, 2*(1), 5-38.

De Wolf, T., & Holvoet, T. (2005). *Emergence versus self-organization: Different concepts but promising when combined* (pp. 1-15). Berlin: Springer-Verlag (LNAI 3464).

Di Marzo Serugendo, G., Gleizes, M.-P., & Karageorgos, A. (2006). Self-organization and emergence in MAS: An overview. *Informatica.*

Di Marzo Serugendo, G. (2005). On the use of formal specifications as part of running programs. *Proceedings of Software Engineering for Multi-Agent Systems IV Workshop* (pp. 224-237). Berlin: Springer-Verlag (LNCS 3914).

Di Marzo Serugendo, G., Karageorgos, A., Rana, O. F., & Zambonelli, F. (Eds.). (2004).

Engineering self-organizing systems. Berlin: Springer-Verlag (LNAI 2977).

Ducatel, K. et al. (2001). *Scenarios for ambient intelligence in 2010.* Technical Report, Institute for Prospective Studies.

Fabrega, M., Lòpez, B., & Masana, J. (2005). How bee-like agents support cultural heritage. *Proceedings of the Engineering Self-Organizing Applications Workshop* (ESOA'05) (pp. 206-220).

Fiadeiro, J. L. (1996). On the emergence of properties in component-based systems. *Proceedings of the International Conference on Algebraic Methodology and Software Technology* (AMAST'96) (pp. 421-443). Berlin: Springer-Verlag (LNCS 1101).

Foukia, N. (2005). IDReAM: Intrusion Detection and Response executed with Agent Mobility. *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS'05), Utrecht, The Netherlands (pp. 264-270).

Glansdorff, P., & Prigogine, I. (1971). *Thermodynamic study of structure, stability, and fluctuations.* New York: John Wiley & Sons.

Grassé, P. P. (1959). La reconstruction du nid et les interactions inter-individuelles chez les bellicositermes natalenis et cubitermes sp. la théorie de la stigmergie: Essai d'interprétation des termites constructeurs. *Insectes Sociaux, 6,* 41-83.

Hales, D., & Edmonds, B. (2003). Evolving social rationality for MAS using "tags." *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS'03) (pp. 495-503).

Hales, D. (2005). Choose your tribe! Evolution at the next level in a peer-to-peer network. *Proceedings of the Engineering Self-Organizing Applications Workshop* (ESOA'05) (pp. 61-76).

Hassas, S., Di Marzo Serugendo, G., Karageorgos, A., & Castelfranchi, C. (2006). Self-organizing mechanisms from social and business/economics approaches. *Informatica.*

Hofmeyr, S., & Forrest, S. (2000). Architecture for an artificial immune system. *Evolutionary Computation Journal, 8*(4), 443-473.

Holland, J.H. (1998). *Emergence—From chaos to order.* Oxford, UK: Oxford University Press.

Jelasity, M., & Babaoglu, O. (2005). T-Man: Gossip-based overlay topology management. *Proceedings of the Engineering Self-Organizing Applications Workshop* (ESOA'05) (pp. 1-15).

Kamvar, S. D., Schlosser, M. T., & Garcia-Molina, H. (2003). The Eigentrust algorithm for reputation management in P2P networks. *Proceedings of the 12th International World Wide Web Conference* (WWW 2003) (pp. 640-651).

Karuna, H. et al. (2003). *Self-organizing in multi-agent coordination and control using stigmergy* (pp. 105-123). Berlin: Springer-Verlag (LNAI 2977).

Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *Computer, 36*(1), 41-50.

Mamei, M., & Zambonelli, F. (2003). *Self-organization in multi-agent systems: A middleware approach* (pp. 233-248). Berlin: Springer-Verlag (LNAI 2977).

Mamei, M., Zambonelli, F., & Leonardi, L. (2002). Co-fields: Towards a unifying approach to the engineering of swarm intelligent systems. *Proceedings of the 3rd International Workshop on Engineering Societies in the Agents*

World (ESAW'03) (pp. 68-81). Berlin: Springer-Verlag (LNCS 2577).

Mano, J.-P., Bourjot, C., Lopardo, G., & Glize, P. (2006). Bio-inspired mechanisms for artificial self-organized systems. *Informatica.*

Varela, F. (1979). *Principles of biological autonomy.* Elsevier.

Weeks, S. (2001). Understanding trust management systems. *Proceedings of the IEEE Symposium on Security and Privacy* (pp. 94-105).

Wooldridge, M. (2003). *An introduction to multi-agent systems*. New York: John Wiley & Sons.

Zhu, H. (2005). Formal reasoning about emergent behaviors of MAS. *Proceedings of the 17th International Conference on Software Engineering and Knowledge Engineering* (SEKE'05).

Chapter XXX
An Evolutionary Algorithm for Decisional Assistance to Project Management

Samuel Rochet
LESIA, INSA, France

Claude Baron
LESIA, INSA, France

ABSTRACT

Manufacturers must always develop products faster and better to satisfy their client's requirements. To help them, we have developed and experimented with a methodology to improve the management process by connecting it with the design process. An issue for the project manager is to select an organization from among the possible ones to schedule the project tasks in order to reach the targets in terms of costs, duration, quality, and so forth. This constitutes a tricky operation, because many options defined during the design can be associated to each task of the project. Choosing and optimizing the schedules is a combinatorial problem which can be solved by heuristic. This document explores the use of evolutionary algorithms to help the decision maker. It introduces the industrial context and presents our methodology to connect the design and project management processes, expressing the problem as a multi-objective optimization one. We detail the scenarios selection process and demonstrate which performances are obtained.

INTRODUCTION: THE INDUSTRIAL CONTEXT AND OUR RESEARCH OBJECTIVES

The globalization of markets and an intensive competition deeply modified the companies' organization. Manufacturers must have a great ability to react in order to satisfy the customers' requirements with shorter times to market and higher product quality. The durations of the innovation cycle were thus notably reduced in all industrial sectors: transportation, telecom-

munication, and so forth. To obtain such results, manufacturers had to considerably modify their working methods. Generally speaking, they need to keep the costs reduced to best evaluate the risks, to capitalize the knowledge and know-how, and to avoid reproducing previous errors. Therefore, today, designing innovative systems is essentially based on the mastery and the fast integration of several technologies, methods, tools, and know-how, both on the technical and on the organizational plans. A strong level of complexity, on the technical plan, characterizes the process design due to the variety and the interdependence of the design choices, but also on the organizational plan, because of the distributed feature of project management.[1] Furthermore, worried about best satisfying their customers' requirements, manufacturers have to specifically customize their products.

For a very long time, development was organized into two phases. A phase of design came first during which the system functional and organic architectures were determined in order to respect the functional requirements. Then, a project management phase intervened to schedule a list of activities deducted from the design and verifying the satisfaction of the non-functional requirements (mainly risks, quality, delays, and costs). However, this organization presents two major drawbacks:

- *A minor design decision can have major repercussions on the organization of activities leading to obtain the product.* For example, moving a structural element on an aircraft will induce technical changes with repercussions on various project activities.
- *Reciprocally, management decisions can also have consequences on the way of designing the system* (giving up a too expensive or too much risked technology, subcontracting parts of the system, etc.).

This mode of organization is not satisfactory: a close collaboration between teams turns out to be necessary. It is thus essential to build and to formalize the design at the same time as the project management. The time to market, as well as the risks of incoherence due to the lack of interactions between design and organization, would indeed be considerably reduced if the technical, administrative, and financial decisions leaned on a unique data model that would be shared by all the partners and based on the previous experiences (failures and successes). This model should be able to contain all project information to reuse them, but it should also be able to favor the traceability of design and management choices.

Such a model is not currently available, and tools integrating design and management aspects, notably by pointing out the consequences of a design decision on the organization (and reciprocally), are today still lacking. Now, it becomes essential for a decision maker to have methods and tools to organize his project according to the design choices. Indeed, design engenders numerous scenarios due to the existence of options in the choice of technologies, architectures, material, or software allocations of the functions. The representation and the management of such scenarios are complex, strongly combinatorial, and are not generally available in project management tools, which are classically based on static models. However, when trouble occurs during the project, or when a major deviation occurs from the initially envisaged organization of activities, it seems important to us that the choice of a new working plan be guided by its coherence from a technical point of view with regard to the design options already taken, but also from a project point of view with regard to the management choices made before. In this context, project manager has to first identify a set of possible scenarios to lead the project. Then, among

these, he has to select one or several that will allow him to reach the project technical and non-technical targets as closely as possible. He finally must organize his project and arbitrate among solutions, about which he often does not have enough information because they are generally dependent on design decisions.

In this context, our first objective is thus to help the project manager to define, follow, and adapt a coherent organization of project activities. This essentially relies on two actions: favoring the project manager coordination with the technical design team, and assisting his decisions from among the several options to organize the project. Our second objective is to make the management process robust and adaptive in light of external technical, social, or economical hazards, in order to reach the fixed project targets as closely as possible. However, selecting a limited set of "best" scenarios among thousands of scenarios is a large-scale combinatorial problem.

After a brief presentation of the methodology that we propose in order to couple design and project management and the context of our research, this chapter focuses on the problem of choosing a scenario from among the several possible ones to lead a project. This problem can be considered as a multi-criteria problem of distances minimization between the performances reached by the scenario (in terms of costs, delays, quality, risk associated to the project) and the fixed targets. Such a problem is highly combinatorial, and it is not possible to examine all possible values of variable combinations to find a solution. Consequently, no partial enumeration-based exact algorithm can consistently solve them. However, by exploiting the problem-specific characteristics, heuristic methods aim at a relatively limited exploration of the search space, thereby producing acceptable quality solutions, in short computing times. We thus suggest use of evolutionary algorithms to make the selection of scenarios. This chapter explains how to apply these to our problem and demonstrates their efficiency.

OUR METHODOLOGY TO CONNECT SYSTEM DESIGN AND PROJECT MANAGEMENT PROCESSES: A SOLUTION TO INDUSTRIAL REQUIREMENTS

Today's Industrial Requirements

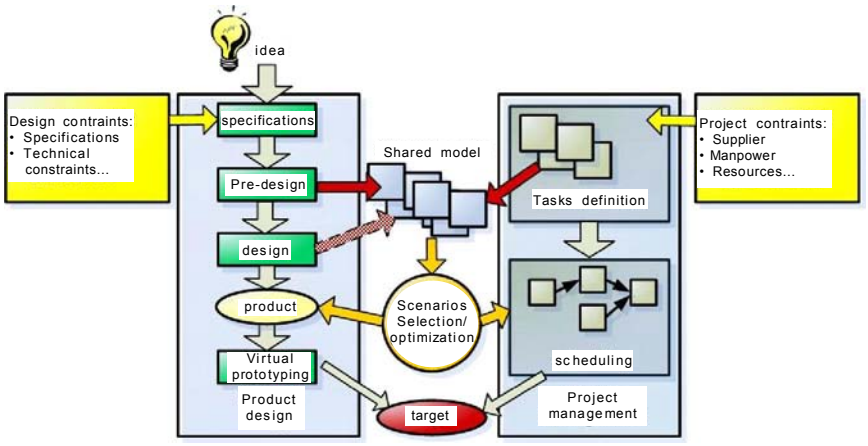
The accelerated development of technologies offers a wide range of materials, components, production modes, and so forth to the engineer to design products, whose lifetimes are becoming shorter and shorter. Manufacturers, in this very competitive international context, need to speed up transactions, reduce operating costs, and improve performance and customer service. A main problem comes from the fact that generally, once the product characteristics are established, manufacturers must master the product design methodology and manage the project simultaneously. These two processes are led in parallel, but are usually separate, without connected tools.

This current practice presents risks of incoherence that usually lead to increased costs and delays: innovation process is not enough correlated with economic parameters, and project management is led with an insufficient knowledge of the technical difficulties. Obviously, these risks could be reduced if all technical, administrative, and financial decisions were based on a shared data model.

Our Solution

The originality of our proposition is to consider that both design and management processes can be associated from the very first steps of

Figure 1. Connection between design and management processes



high-level design and the definition of the project tasks, then throughout the product development cycle (see Figure 1). Indeed, on the design side, the partitioning of a system into reusable subsystems is coupled with the partitioning of corresponding reusable project tasks to organize the product development on the other side. So, the identification of technical options for the product (performance, reliability, etc.) participates to the definition of the project tasks, which add their own constraints (costs, delays, suppliers, etc.).

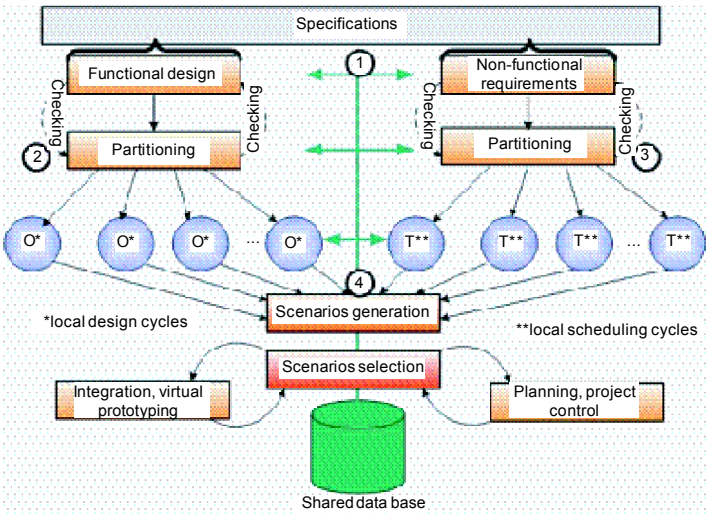
It is then possible to define a "shared" data model for the project, integrating constraints from both design and management sides. Thus, each actor can keep his own point of view on data: on the design side, the high-level simulation of interconnected subsystems leads to virtual prototyping and verification, and on the management side, the definition of different tasks leads to several possible project organizations. From this shared data model will be generated and selected some valid project scenarios. The main interest is that these optimal scenarios, relying on shared functional and non-functional data, participate to the definition of a coherent project organization (task schedule). The manager, if he needs to reduce the global project cost, will be able to envisage at which technical level the problem can have repercussions, and the designer will choose different architectures for the product among the possibilities offered by the design.

The Partners, the Coupled Methods, the Interconnected Tools

These studies on the interests to couple design and management processes are conducted within the framework of a collaborative project between several research groups (the LAAS from the French National Center for Scientific Research, the LESIA Laboratory, and many others) and industrial partners (Airbus, Mentor Graphics, etc.). We presently are in a phase where, following a prototyping approach, we are connecting several individually developed methods and tools to build a shared platform. With these tools, we plan to connect the design and project management processes through the use of the following methodology, whose steps appear in Figure 2:

1. Initially, the textual specifications are formalized by separating technical requirements from project constraints. The formalized data will thus be used during the

Figure 2. Methodology proposed: Several scenarios are generated starting with information provided from design and management, each of them respecting the initials specifications. An optimal scenario is selected and applied to the two previous processes.



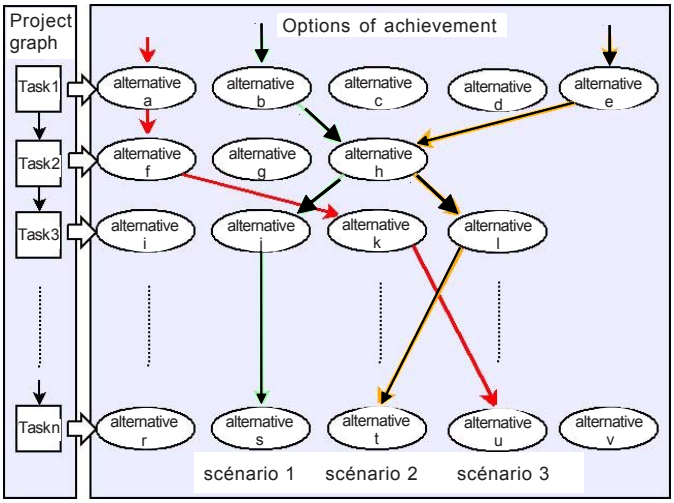
project for semi-automatic validations of the choices. These requirements can represent objectives (product performances, low assembly cost) or constraints (limited resources, delays).

2. Then, a specification tool allows representing the various products' functional architectures with respect to the technical constraints. After this step, the functional architectures are manipulated in order to build physical architecture of products (Baron, Rochet, & Gutierrrez, 2005; Rochet, Hamon, Gutierrez, Baron, & Esteve, 2005). At this stage, a second verification of the model coherence with initial technical constraints is made. One thus obtains several possible product structures able to respect the technical constraints.
3. Considering these different architectures, the project team proceeds to various calls for tenders that will be used as a basis to elaborate a panel of possible organizations for the project.
4. On this basis, we are able to construct several possibilities to conduct the project called *scenarios,* each one respecting the project's constraints, between which a tricky choice has to be made. This step will be detailed in the remaining part.

Our objectives thus are to provide the decision maker with a reduced set of project's scenarios from which he will make a choice.

Note that in this description, the process we recommend to connect design and management processes can appear as a linear process. However, once the first technical choices are made, the project team can find several options to organize the project. It is thus these two processes that are led in parallel with a strong connection between design and control. This leads to the construction of a project's representation where all the possibilities of realiza-

Figure 3. Generation of scenarios from the alternatives of the project tasks; alternatives are associated at each task, which can be realized indifferently by one of them.



tion are included and on which both engineers and managers can rely to define the final product and a way to lead the project in a more coherent manner.

After this brief introduction to the project, let us now see how a project is represented in our shared model.

Representation of a Project

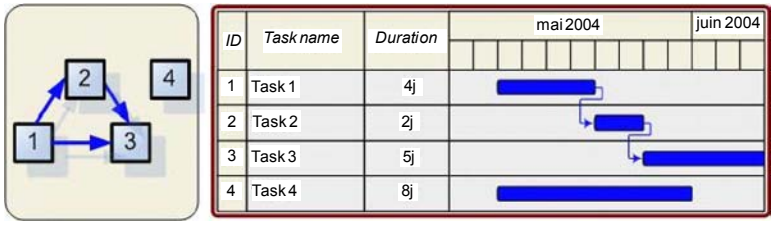
Definition of a Scenario

We define a *scenario* as a way to lead the project in order to reach the objectives fixed by the client. The generation of the different scenarios is made on the basis of technical considerations, but it also takes into account non-technical parameters relative to project management: costs, delays and supply constraints, quality, certification, risks, and so forth. At this stage, we determine which tasks are to be achieved without presupposing how to achieve them. To each task are associated some options of achievement that we call "alternatives." For example the task "obtain the component C" can be indifferently achieved by the alternatives "buy C to the supplier A," "design C on our own," "adapt and customize the component Y," and so on. The level of details at which the alternatives are described depends on the requirements and on the type of project.

In large-scale projects, a task can correspond to a complex phase implicating several teams for a long time; in simple ones, the production process can be more precisely modeled and followed. The model's precision is at the discretion of the person in charge, which can adapt it to its needs.

Whatever the project size, the combinations of task alternatives then lead to numerous solutions for the project organization that are coherent with the general product specifications (as illustrated in Figure 3). A scenario can thus be defined as a combination of task alternatives which corresponds to an option for the whole project organization satisfying the technical specifications and the strategic requirements of the project at the same time (Gutierrez, 2004).

Figure 4. Project graph example with corresponding Gantt diagram



Expression of the Constraints between Tasks in a Project Graph

A project progress is never sequential. To build valid scenarios, constraints between tasks (precedence, resources, etc.) must be taken into account. We have chosen to represent these constraints on a graph; this representation is very close to the formalism proposed by Beck (1999). Let us now introduce the elements of this graph one by one.

The first types of handled constraints are the temporal relations between tasks and more exactly the precedence relations, which constitute the main type of relation in a project. They are represented using an oriented graph in which tasks are represented by numbered squares and precedence relations by directed arcs. Figure 4 gives the example of a possible Gantt diagram for a graph in which task 1 must be achieved before tasks 3 and 2, and task 2 before task 3; task 4 can be achieved independently from the others.

The precedence relation, as defined above, allows the handling of classical scheduling problems, assuming the tasks and their relations are completely known. However, in our problem, our goal is not to represent a single scheduling of the project, but to synthesize in a single and unique global model all possibilities to schedule it. From this representation, the different ways to conduct the project can be identified and extracted, between which the choice of an optimal solution must be made. We can also imagine that some decisions can be made that will determine if a set of tasks must be achieved, for example when several technologies can be chosen to implement a component. The use of alternatives at the task level only is not thus sufficient because the choice of the technology will have impact on the project progress: it is necessary to have an element of the graph representing a choice between the tasks themselves in the project. We thus introduce the "decision nodes," represented with a pair of numbered circles to express this choice. The first circle, with a double outline, represents an initial node; it opens the choice. The second, with a simple outline, represents a final node (Figure 5); it closes the choice.

Nodes represent an exclusive "OR" in project development; a single sequence of tasks is then allowed between an initial and a final node. For

Figure 5. Structure of project with decision nodes; tasks are represented by squares, choices by two associated nodes, and precedencies by the arrows.

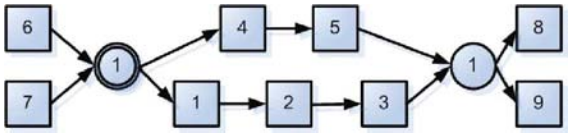
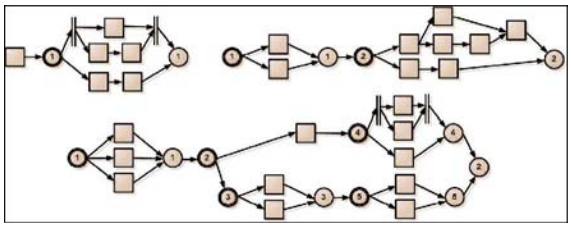


Figure 6. Examples of project graphs



example, there are two possibilities to achieve the project of Figure 5: either achieve the tasks 1, 2, 3, 6, 7, 8, and 9, or the tasks 4, 5, 6, 7, 8, and 9. We call "path" a set of elements (tasks or nodes) belonging to a same sequence between the two nodes of a choice. In the previous example, there are two paths: [4, 5] and [1, 2, 3].

To complete this representation, a final question must be considered: how do we represent two tasks that must both be achieved just after a decision node? It is necessary to introduce the "AND" operator, symbolized by two vertical lines in Figure 6.

Now, the basics structures of the graph model have been introduced, but a set of rules must be satisfied to avoid inconsistent graphs:

1. the graph must be acyclic,
2. each node belongs to a pair constituted by an initial and a final node,
3. the element preceding an "AND" can only be an initial node, and
4. no direct relations can exist between elements of different paths.

In a more formal way, we can write:

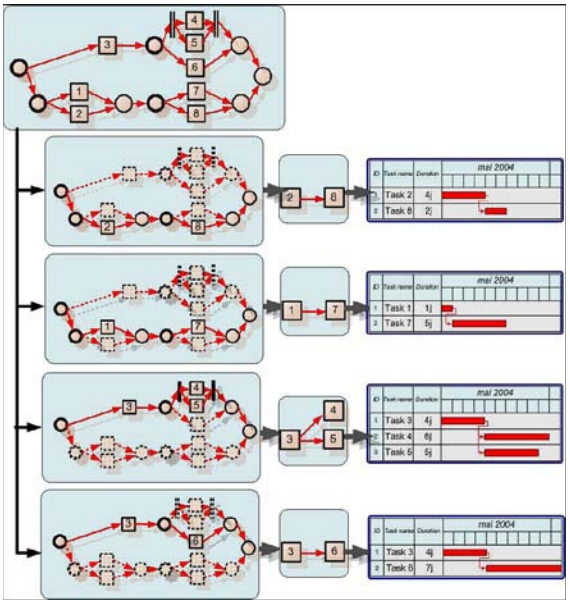
- Let t be a task and T the set of tasks t_i. Each task t_i of the set T owns m_i alternatives.
- Let n^s be an initial node of the graph, N^s the set of initial nodes, n^f a final node of the graph, N^f the set of the final nodes, n^c a couple of nodes (an initial node n^f and its associated final node n^f), and N the set of couples of nodes n^c.

Each node n^c_i of the set N owns n_i possible choices to reach its final node n^f_i from its initial node n^s_i.

Of course, each alternative and task has its own attributes and constraints such as duration, need resources, and so forth, which make it possible to build a schedule starting this representation.

On the basis of this formalism, useful to represent all the possibilities to follow/lead a project progress, complex project structures can be described. The graph structure in Figure 6 allows the automatic generation of different project scenarios according to the active path in the project graphs (Figure 7), reduced graphs, each one representing a scenario. These reduced graphs then individually lead to a specific

Figure 7. Scenarios generation and scheduling; starting on the project graph, reduced graphs can be deduced, each one corresponding to a specific scheduling.



scheduling. After this operation, the main project characteristics are known: non-functional ones like cost and duration, other important evaluation criteria such as quality risk, and functional ones like product performance for example. The functional and non-functional characteristics will be the base of the evaluation of the scenarios, and of course of their associated scheduling.

So we dispose of a synthetic representation on which all the possibilities to lead the project are present. We will see after how the choice can be made, but first, the next section presents how a scenario is evaluated.

What is a Good Scenario?

The previous section presented a synthetic project representation, from which are deduced different scenarios, each one leading to a specific scheduling. The problem now is to select the "best" scenarios.

In a simplified way, a scenario that reaches the set of objectives is a good scenario (this idea will be detailed further on). Let us consider a performance vector v of a scenario (Equation 1: Performance vector) in which are taken into account n criteria. t_{vi} represents the value of the objective i and e_{vi} the estimated value of the i^{th} criterion. The ideal scenario would be a scenario that is optimum on all considered criteria. That is to say, v is optimal if $\forall i, x : v_i \leq x_i$.

$$v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} |t_{v1} - e_{v1}| \\ |t_{v2} - e_{v2}| \\ \vdots \\ |t_{vn} - e_{vn}| \end{pmatrix} \quad (1)$$

Unfortunately, such solutions do not always exist. Then we have to find a compromise between these various, and often contradictory, criteria (between project duration and risk for example). This is why the concept of dominance in Pareto meaning (Pareto, 1906) is used to compare two scenarios.

Let u and v be two scenarios; if all the components of v are strictly lower or equal to those of u, with at least a strictly lower component, then we can say that the vector v corresponds to a better solution than u. In this case, v dominates u in Pareto meaning. This can be formally expressed this way: $v \overset{p}{\prec} u$ (Equation 2: Pareto optimality).

$$v \overset{p}{\prec} u \Leftrightarrow \forall i \in [1;n] : v_i \leq u_i \wedge \exists j \in [1;n] : v_j < u_j \quad (2)$$

where n represents the number of problem's objectives.

If a scenario is not dominated by another one, then it belongs to the set P_{op} of the Pareto-optimal scenarios, called Pareto front. This concept can be illustrated by Figure 8 where we can find the optimal solutions for a minimization problem on the borderline of the possible solutions space.

Our objective is then to find, in the set of valid scenarios, those that offer the most inter-

Figure 8. Graphical representation of the Pareto front for a two objectives minimization problem; each solution at the right of the Pareto front is suboptimal.

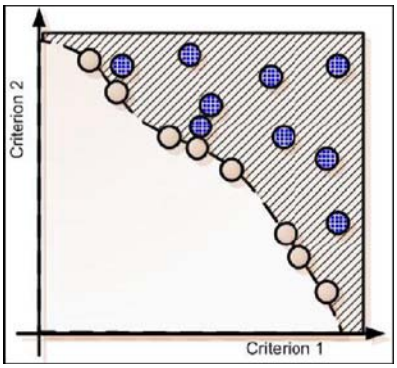
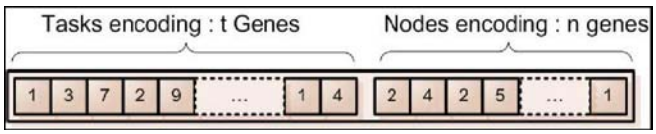


Figure 9. Chromosome's code: An integer string with the first genes coding the alternatives of tasks, and the last genes coding the method of selection in the project graph for each choice



esting compromises: the Pareto-optimal scenarios. Among these, the decision maker can carry out his final choice.

OBTAINING THE PARETO-OPTIMAL SCENARIOS WITH AN EVOLUTIONARY ALGORITHM

Having presented the problem of the selection of project scenarios, this chapter will now focus on the process to obtain the Pareto-optimal solutions. Theses solutions thus constitute the first subset of "good" scenarios to offer to the decision maker to organize his project. We will see in the next part how to improve this process by classifying the scenarios.

We have seen that a way to lead the project can be represented by a path in a graph. However, finding an optimal scenario in this type of graph is a difficult problem to solve. The combinatorial aspect generally prevents us from exploring all the possible solutions on such large-scale problems. In addition, we have no mathematic formulation of our problem on which mathematic tools like derivation, for example, could be used. That is why we need to use a non-exact method to solve this problem. We have chosen to use an heuristic based on evolutionary algorithm because an algorithm working on a population is more likely to find the Pareto front solutions than an algorithm working with a single solution (simulated annealing or taboo search for example) that needs to be executed several times. As describes in Coello's chapter of this book, a strong development of the "Evolutionary Multi-Objectives Optimization" tends to prove the efficiency of such method.

Before describing the algorithm used, we first present how the individuals are coded.

The Chosen Coding of Individuals

To build a scenario, it is necessary to choose one alternative for each task and the decision made at each choice of the project—that is, the path followed after every initial node N^d_i. Two levels of choices can be identified: the first is to select a path in the project graph, and the second is to select one alternative by task.

We chose to code the chromosomes in the following way: in the first part, the chromosomes are coded according to the alternatives associated to each task; in the second part, they are coded according to the active path of each node, indicating the tasks to be achieved (see Figure 9).

Let Tab_T be a table of length t = card (T) (number of tasks) and Tab_N be a table of length n = card (N) (number of pair of nodes). Each element i of the Tab_T represents the number of the selected alternative for the task T_i, and each element j of Tab_N represents the number of the choices made between an initial node N^d_j and the corresponding final node N^a_j.

A chromosome Ch is the concatenation of the two tables Tab_T and Tab_N.

Roughly speaking, we can consider that the second part of the chromosome acts as a filter because it helps to determine what tasks must be achieved. The choice of an alternative for the tasks that must not be achieved is thus indifferent, and two different chromosomes can represent the same scenario once decoded. However, in spite of the drawback to have a space of genotypes larger than the one of phenotypes, this representation allows having valid individuals after a crossover or a mutation. We thus avoid an additional verification step of individuals' viability and their eventual reconstruction.

The Evolutionary Procedure

The chosen evolutionary procedure relies on the classical sequence of an evolutionary algorithm. After an initialization of the population, we repeat the steps of evaluation, selection, crossover, and evaluation until obtaining the stop criterion.

The steps of our algorithm are now detailed.

Initial Population Initialization

Without any a priori information, the initial population is randomly generated in order to cover the search space as well as possible.

Selection

We have chosen the "roulette wheel" strategy where the probability for each individual to be present in the following generation is directly proportional with its fitness f_i.

Crossover

As said previously, the representation chosen to code the individuals introduces a predominance of the genes coding the choices on the genes coding the task alternatives. It is thus not advised to use a single point crossover or N-points with N too weak, for which we risk a premature convergence. To avoid that, we use a uniform crossover. Each gene of the child thus has the same probability of being inherited from one parent or the other. We thus suppose that a distributational bias[2] is less penalizing than a positional bias.[3] Figure 10 gives an example of crossover and the impact of this one on the generated scenarios. We can observe in this example that the child comprises many identical elements to each of its parents (task 1 relative to parent 1, and tasks 6, 7, and 8 relative to parent 2), but also a more complex combination as task 3, which is to be realized as that of parent 1 but with the alternative 2, which was coded from parent 2. Thereafter, we will note the crossover probability p_c.

Mutation

We use a uniform mutation by replacing the value of a gene by another value belonging to the field of the possible values for this gene with the probability p_m. Figure 11 gives an example of mutation on the basis of the Figure 10 project. The second gene mutates from 5 to 2 and thus involves a change of the alternative selected for the second task. It is not the fifth any more, but the second alternative that is selected. In the same way, a gene coding the choices can change, thus involving a new set of tasks to realize.

Evaluation

After each generation, we evaluate the individuals of the new population. Evaluation is decomposed into three steps:

1. Find the value of the various criteria of the scenario—that is, decode the chromo-

Figure 10. Crossover example: illustration of a possible child of parents 1 and 2, and its associated reduced graph

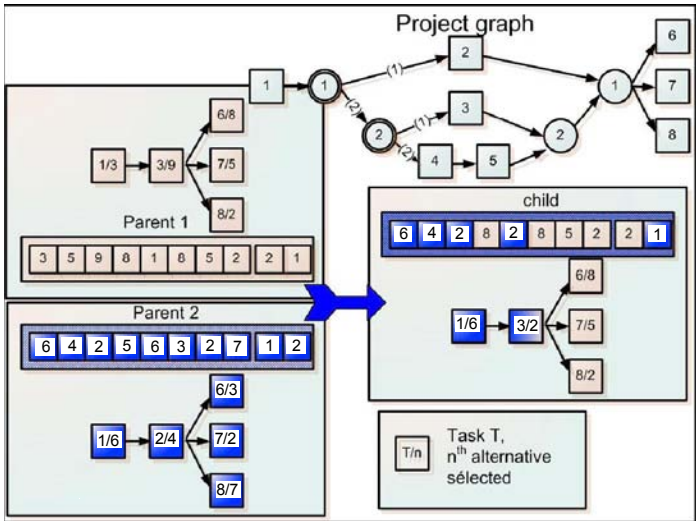
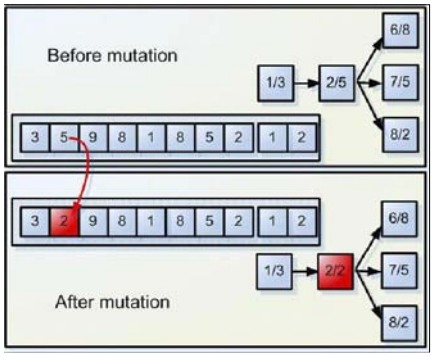


Figure 11. Mutation example



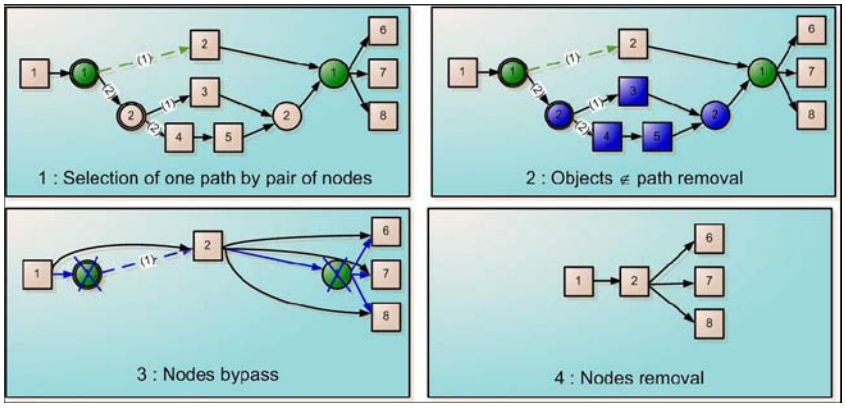
some in order to build the reduced project graph from which a schedule can be carried out which will provide us with these values (see Figure 7).

2. Form the reduced project graph, and determine the scenario performances and more precisely the difference between the scenario performances and the objectives vector.
3. Give a fitness to the individual according its performance.

Project Graph Reduction

Before each evaluation of a chromosome, it is necessary to transform the complete project representation into a simplified representation in which the choices are eliminated. To carry out this reduction, we associate to each individual a project graph where the selected alternatives of each task are given according to the first part of its chromosome. Then we successively reduce this graph by carrying out, for each pair of nodes, the following phases (see Figure 12): decoding of the chromosome to determine the selected way, suppressing the objects (tasks and nodes and "AND") between the initial node and the final node not belonging to the selected path, creating precedence links between the tasks before final nodes after initial nodes, and, finally removing the current pair of nodes.

Figure 12. Project graph reduction: the reduction is a succession for each pair of nodes of the removal of objects not belonging to the path selected, the bypass of the nodes, and their removal.



Performance Evaluation of the Scenario

This reduced project graph corresponds to a more conventional scheduling problem (no choices between tasks, no more than one alternative to realize a task). Then we can use traditional scheduling methods to determine the value of the various scenario criteria. Currently, to illustrate our approach and to test its efficiency, we work with two essential criteria: cost and delay. Other criteria, like quality, product performance, or risk, can be considered in the same way since the method allows evaluating it.

From the values of the various parameters, we can easily determine the differences between the desired and reached values for each individual of the population. Once the performances of the whole population are evaluated, we then allocate to each individual a note that represents its adaptation to the environment f_i. This will be reused during the selection stage.

Fitness Function Attribution

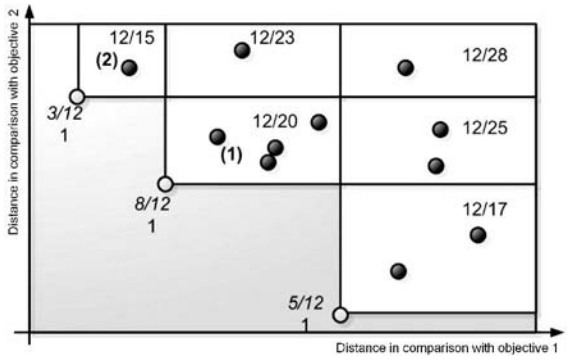
Evaluation makes it possible to find the scenarios whose variations compared to the objectives are minimal (thus in the Pareto front) while developing the population diversity. For fitness function calculation, we have chosen to use a method inspired[4] from the *strength Pareto evolutionary algorithm* (SPEA) (Zitzler & Thiele, 1999).

We worked with two sets of individuals: P being the population and I_{fp} being the whole set of individuals in the Pareto front.

If a new individual of the population P appears to be Pareto-optimal, then a copy of it is created in I_{fp} before suppressing, if necessary, the individuals that would not be Pareto-optimal anymore. The fitness calculation of individuals is broken up into two steps:

- **Step 1:** To each individual of I_{fp} is associated s_i, a value representing its strength, which is equal to a, the number of solutions that it dominates in P, divided by ¼ the population size plus one Eequation 3: Strength of Pareto front's individuals).
- **Step 2:** Each individual I_p having a fitness f_i is equal to the contrary of the sum of the strength of the individuals that dominate them increased by one (Equation 4: Fitness function).

Figure 13. Example of fitness allocation



$$s_i = \frac{\alpha}{\mu + 1} \tag{3}$$

$$f_i = \frac{1}{1 + \sum_{\substack{p \\ i,i<j}} s_i} \tag{4}$$

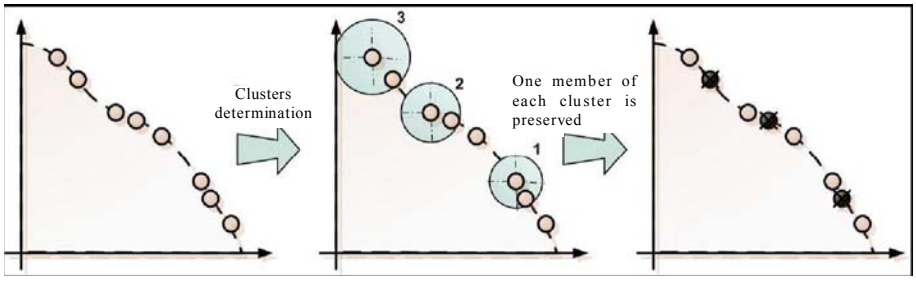
Thus, an individual is less powerful when it is dominated by I_{fp} individuals (see Figure 13).

This way favors the individuals in the Pareto front. That allows supporting convergence towards the good scenarios. The separation of the individuals belonging to the Pareto front from the rest of the population allows filtering of solutions and introduces a memory. Lastly, a niching is implicitly effectuated during fitness calculation. The objective space covered by the Pareto-optimal set can be divided in subspaces (or niches) according to the individual's distribution. If a niche contains a high number of solutions, its contribution to the strength is high. In return, this high strength implies a weak fitness for individuals belonging to this niche (Figure 13, niche 1, and conversely, Figure 13, niche 2). One thus obtains a subpopulation size stabilization into niches.

However, in some kind of optimization problems, the Pareto-optimal set can be extremely large or even contain an infinite number of solutions. Even if, in practice, it is limited by the population size, a restriction of the size of I_{fp} is unavoidable (Zitzler & Thiele, 1998). If the number of individuals in the Pareto front exceeds a fixed threshold, a method of clustering is used to reduce it. The average linkage method (Morse, 1980) is used. In this, a group of close individuals is replaced by the one that is the most representative of the group, and as many times as necessary to obtain the good number of individuals in I_{fp} (see Figure 14).

With the generation of the initial population, each individual of the Pareto front represents a basic cluster. Then, at each step, two clusters are selected to be incorporated in a larger one until the given number of clusters is reached. Both clusters are selected according to the nearest-neighbor criterion. When the division process is finished, I_{fp} is made up by preserving

Figure 14. Example of Pareto front reduction using the average linkage method; grouping close individuals creates clusters, and a single individual is conserved in each cluster.



only one individual by cluster, the individual in the center of the cluster.

Stop Criterion

Currently, we use a strict limitation of the number of generations. However, we consider the use of more sophisticated stop criteria, taking into account the problem complexity and the obtained solutions quality.

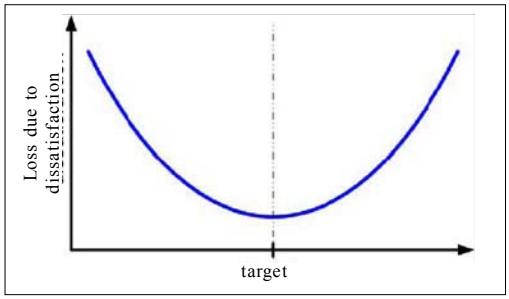
PARETO-OPTIMAL SCENARIO CLASSIFICATION

The above process helps the decision maker build a schedule for his project and aim to reduce the problem complexity (select a valid and optimized schedule from a set of tasks) by selecting a finite set of "good" scenarios. For selection, we use the Taguchi loss function. This section first introduces the Taguchi loss function, presents the use of Taguchi to determine the cost of the selected scenarios, and lastly explains how to classify the best scenarios.

Tagushi Loss Function

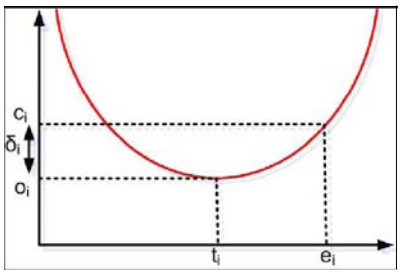
Designing a system consists of obtaining an exact conformance with the functional requirements (execution, quality, reliability, testability) while respecting technical constraints. However, the project manager must handle a management process that must lead to reaching the target, while respecting the constraints of delays and means (budget, manpower, machines, suppliers, etc.). In order to measure how precisely the target is reached and, within the global objective, to analyze how well the previously mentioned different constraints used to define it are respected, we decided to use Taguchi's approach, as proposed by Yacoub (2004).

Figure 15. Loss function of Taguchi



Taguchi methods, developed by Dr. Genichi Taguchi (Taguchi, 1986), fundamentally refer to techniques of quality engineering that embody new quality-related management techniques. The entire concept can be described in a basic idea: quality should be measured by the deviation from a specified target value, rather than by conformance to predefined tolerance limits. Traditionally, quality is modeled as a step function: a product is either good or bad. This view assumes a product is uniformly good. Taguchi believes that the customer becomes increasingly dissatisfied when performance moves out of the target. He suggests a quadratic curve to represent customer dissatisfaction with a product's performance. The curve is centered on the target value, which provides the best performance according to the customer. This is a customer-driven design specification rather than an engineer-driven design. As seen in Figure 15, there is some financial loss incurred at the upper consumer tolerance. This loss is due to the fact that products of inferior quality are less well sold, and that the superior quality products induce additional costs of design or manufacturing. So, we dispose of an image of the long-term impact of every quality criterion on the product by associating to every criterion i a cost $c_i = o_i + k_i(e_i - t_i)^2$ (see Figure 16) with o_i being the optimal cost, t_i the target value, and e_i the estimated value. K is a user-defined constant.

Figure 16. Representation of additional cost for a given parameter



Use of Tagushi Loss Function to Determine the Cost of the Selected Scenarios

Once sub-optimal individuals are eliminated, the number of possible solutions considerably decreases. We then use the Taguchi loss function to determine the total cost of each selected scenario. For this, we can aggregate the whole costs of each parameter. Let C be the scenario costs vector (Equation 5: Scenario cost vector) in which O is the optimal cost vector and D the overcost vector. The total cost of the scenario is then given by Equation 6: Total cost.

$$C = O + \Delta = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ o_n \end{pmatrix} + \begin{pmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \delta_n \end{pmatrix} = \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ o_n \end{pmatrix} + \begin{pmatrix} k_1(e_1 - t_1)^2 \\ k_2(e_2 - t_2)^2 \\ \vdots \\ k_n(e_n - t_n)^2 \end{pmatrix} \quad (5)$$

$$G = \sum_{i=1}^{n} c_i = \sum_{i=1}^{n} o_i + \sum_{i=1}^{n} \delta_i = \sum_{i=1}^{n} o_i + \sum_{i=1}^{n} k_i (e_i - t_i)^2 \quad (6)$$

Classification of Scenarios

Once the cost of the scenarios known, it is easy to classify the scenarios before presenting them to the project manager. From a practical point of view, our goal is to provide him with a sample of the "best" scenarios (according to the relative importance given by the project manager to each parameters) on which he can visualize the impacts of different design or project management alternatives on the set of projects and/or visualize the effects of a variation on the importance of one of the parameters (variation of d_{ki} around k_i) and thus carry out the obtained results validation. One then seeks to determine the scenarios with the minimum cost

$$G = \sum_{i=1}^{n} c_i .$$

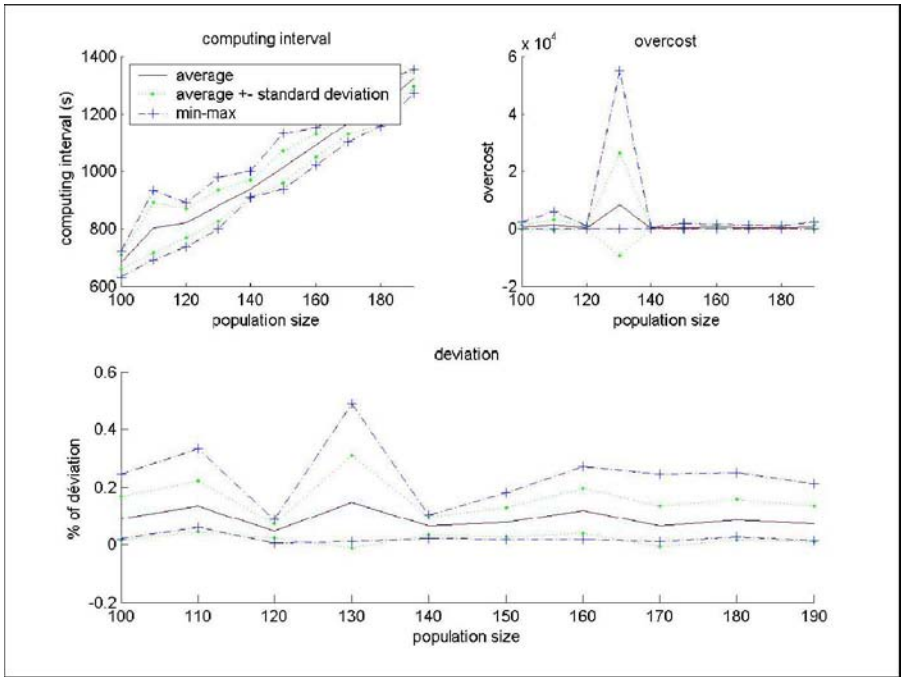
OBTAINED RESULTS AND PERSPECTIVES

To implement the above method, we have developed a tool presently used to validate our approach and to find the optimal evolutionary parameters.

This section describes two types of experiments we already conducted: some to fix the algorithm parameters, others to evaluate the quality of the obtained solutions.

To study the influence of the parameters on the algorithm, we tried to obtain the Pareto-optimal scenarios of randomly generated test problems of high complexity. These were constituted with 100 tasks, each one of 20 alternatives, which represents $20^{100} = 1.27.10^{130}$ possible scenarios. We generated these problems to test in a statistical way the performances obtained on relatively complex problems, which would have been impossible to define manually. We considered, to fix the project targets, that the objectives to reach were defined from one of the scenarios of the problem arbitrarily chosen. We thus tried, during the search, to find this last one. To compare the results provided from two different problems, we introduced an indicator measuring the deviation relative to the

Figure 17. Results obtained according to the size of the population



objectives (Equation 7: Deviation) that allow us to compare the results independently of the values of the optimal parameters of each problem. As we realized tests on randomly generated problems with different objective's values, the same variation for two different projects can be more or less significant if objectives have a high value or not. That is why we prefer to work on the ratio between the deviation from the objective and the objective itself. To determine the influence of the various parameters on the quality of the solutions found as well as on the computing time, we have generated a set of possible regulations for different sizes of population (see Figure 17).

$$d = \frac{\sum_{i=1}^{n} |e_i - t_i| / t_i}{n} \qquad (7)$$

In Figure 17, we can observe that the computing time is a linear function of the size of the population while the average deviation seems stable with regard to the population. On the other hand, the standard deviation, for its part, decreases when the population increases. From these observations, we considered that the acceptable size of the population was situated around 150 individuals. Further, the gain obtained at the level of the quality of the solutions found is not important with regard to the necessary computing time. Thus we had to adjust the values of p_c and p_m by successive tries. The parameters that seem to give the best results on our set of test problems correspond to the values p_m = 17% and p_c = 74%.

Let us now consider the quality of the obtained solutions. By using the parameters above, we obtained during a search the results of

Figure 18. Performance obtained

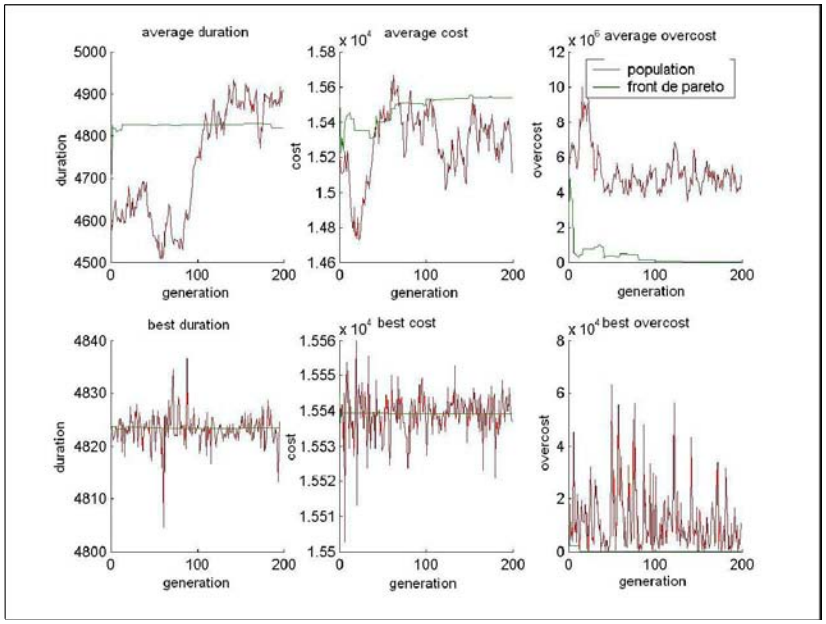
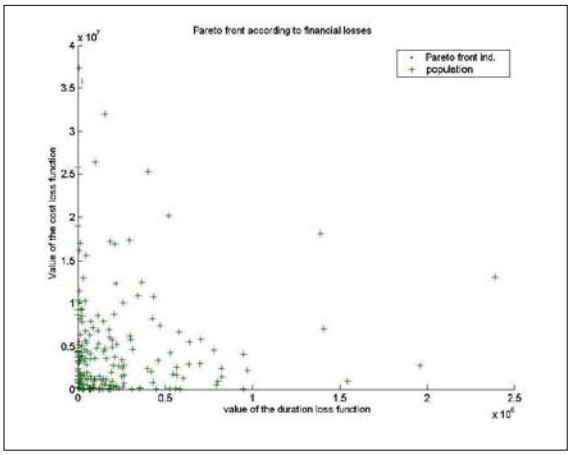
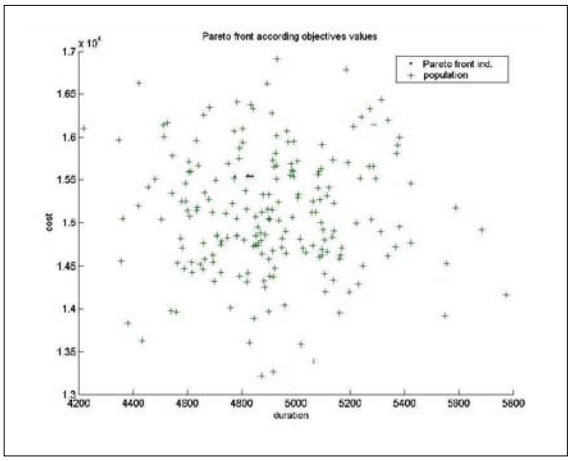


Figure 19. Population repartition for generation 200



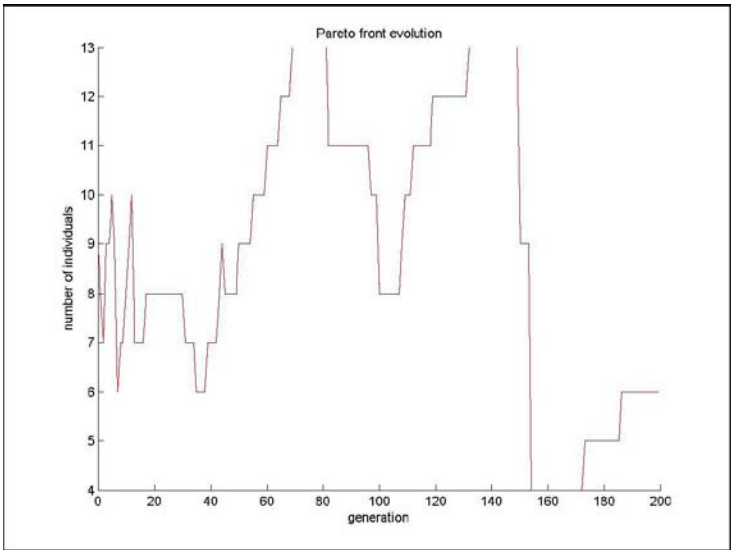
(a) according to financial losses



(b) according to decision criteria

Figures 18 and 19. In Figure 18, we can observe the distribution of the population, as well as the individuals of the Pareto front according to the parameter values and the financial loss. In Figure 19, we can see the evolution of the performances obtained during a search. We can notice that, during this search, the achieving of the desired duration was very fast be-

Figure 20. Number of individuals in Pareto front during the search



cause it took place during the very first generations. It is the progressive increase of the scenario cost that allowed us to find, afterwards, a global compromise decreasing the value of the additional cost.

Finally, Figure 20 reveals the evolution of the number of individuals in the Pareto front during the search. This number is included between 4 and 13 during the search, and the experiment finally ends with six solutions. The number of solutions presented to the decision maker is then completely acceptable. In a general way, let us note here that the number of solutions that will be presented to the project manager evaluation will depend on the problem and will not result from an a priori choice.

The obtained results are thus encouraging, notably regarding the quality level of the proposed solutions and the computation time. Even if "the" optimum solution can scarcely be obtained, the obtained solutions usually are very close to the fixed targets. To strengthen the validation of our approach, we envisage two perspectives at the moment: the first one is to test our tool on real cases supplied by our industrial partners, and the second is to continue to develop this method by studying the performances of other heuristics, such as ant colonies for example.

CONCLUSION

This chapter proposed a new methodology to manage design projects in a more coherent way. To reach his targets, the project manager must define a project by organizing the project tasks and envisaging an initial schedule at the beginning of the project. Then, during the project, whenever anything occurs that modifies the project progress, he must adapt the project organization. When a rescheduling is needed, he must select new tasks or new alternatives to realize some tasks. These alternatives come from the design; the decision maker must choose a new scenario from among the valid ones obtained by the combination of several alternatives to realize the project tasks. Choosing and

optimizing a scenario for the project progress in order to reach the project target as closely as possible thus results in a combinatorial problem that only an heuristic method can solve. This chapter explored the use of evolutionary algorithms to help the decision maker. It first introduced the industrial context of the study, then expressed the problem as a multi-objective optimization one, and explained how we proceed to connect both system design and project management processes. Finally, it detailed the evolutionary algorithm and illustrated it through a selection of the results we obtained. We began experimenting with this approach with manufacturers on the basis of a shared database associating functional and non-functional information; we obtained good results: the selected scenarios, which the tool offers for a final choice to the decision maker, are of good quality (they lead to reaching the targets very closely) and are obtained in a very short time. We are now considering another interesting approach, based on a different but very innovative meta-heuristic method—ant colonies—that seems to be well adapted to the problem of finding the bests paths in a graph.

REFERENCES

Baron, C., Rochet, S., & Gutierrez, C. (2005, June). Proposition of a methodology for the management of innovative design projects. *Proceedings of the 15th Annual International Conference on System Engineering* (INCOSE), Rochester, NY.

Beck, J. (1999). *Texture measurement as a basis for heuristic commitment techniques in constraint-directed scheduling.* Unpublished doctoral dissertation, Department of Computer Science, University of Toronto, Canada.

Gutierrez, C. (2004, June). *Méthode et outil pour une conception système cohérente avec la conduite de projet, application l'étude d'un dispositif multicapteur autonome et interrogeable* (mémoire de stage). LESIA INSA, Toulouse, France.

Morse, J. (1980). Reducing the size of the non-dominated set: Pruning by clustering. *Computers and Operations Research, 7*(1-2), 55-66.

Pareto, V. (1906). *Manuale di economia politica.* Milan: Piccola Biblioteca Scientifica. (Translated into English by Ann S. Schwier (1971), *Manual of political economy.* London: Macmillan.

Rochet, S., Hamon, J.-C., Gutierrez, C., Baron, C., & Esteve, D. (2005, June). Vers une démarche de conception cohérente—proposition d'une méthode et outils pour la gestion des alternatives de conception. *Proceedings of the 6th Congrès International de Génie Industriel,* Besançon, France.

Taguchi, G. (1986). *Introduction to quality engineering.* White Plains, NY: Kraus International Publications.

Yacoub, M. (2004). *Conception de systèmes: Evaluation technico-économique et aide la décision dans la conduite d'un projet "recherché."* PhD thesis, LAAS, France.

Zitzler, E., & Thiele, L. (1998, May). An evolutionary algorithm for multi-objective optimization; the strength Pareto approach. *TIK Report, 43.*

Zitzler, E., & Thiele, L. (1999). Multi-objective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257-271.

ENDNOTES

1 Production in large-scale projects is indeed more and more distributed between the actors of a project (production sites, subcontractors) and can cover large geographic zones.

2 Each parent is highly likely to transmit 50% of its genes.

3 Parents transmit contiguous gene sequences.

4 In the algorithm proposed by Zitzler and Thiele (1999),

$$f_i = -\left(1 + \sum_{\substack{p \\ i,i<j}} s_i\right).$$

We use a different formula in order to increase the selection by enlarging distance between "bad" and "good" individuals.

Chapter XXXI
How Genetic Algorithms Handle Pareto-Optimality in Design and Manufacturing

Nirupam Chakraborti
Indian Institute of Technology, India

ABSTRACT

An informal analysis is provided for the basic concepts associated with multi-objective optimization and the notion of Pareto-optimality, particularly in the context of genetic algorithms. A number of evolutionary algorithms developed for this purpose are also briefly introduced, and finally, a number of paradigm examples are presented from the materials and manufacturing sectors, where multi-objective genetic algorithms have been successfully utilized in the recent past.

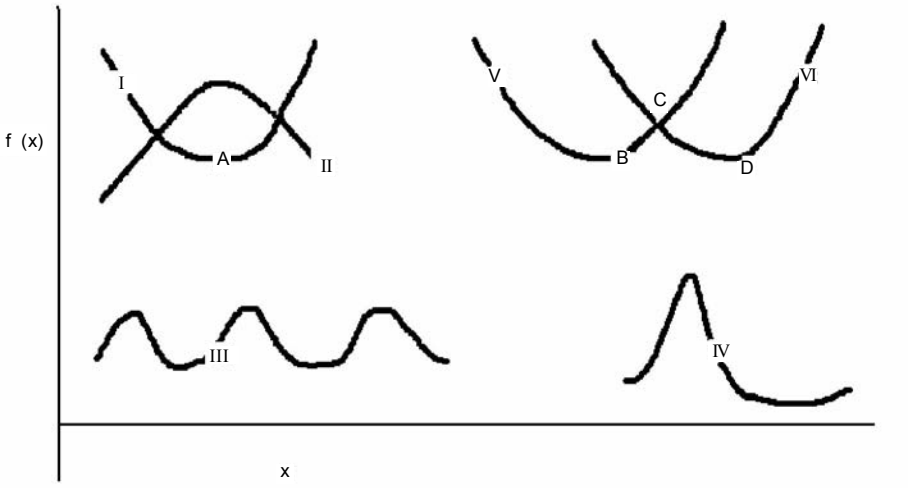
INTRODUCTION

Why Pareto-Optimality and What it is

Making decisions based upon a single criterion is increasingly becoming a difficult task in the complex scenario of design and manufacturing, as we encounter it today. More than one condition routinely affects the complex industrial processes, both at the design and the manufacturing stage. Several criteria that need to be satisfied simultaneously often become conflicting, rendering the search for an absolute and unique optimum in many cases as nearly impossible. I will further elaborate this point using the schematic diagram shown in Figure 1, where a total of six functions (I to VI) are shown schematically and vertical lines drawn through the points A to D would determine some unique combinations, either in the function pairs I and II or V and VI. Now suppose that using these functions we want to accomplish any of the following sets of tasks:

a. Minimize I and at the same time Maximize II
b. Minimize (or Maximize) III and at the same time Minimize (or Maximize) IV
c. Minimize V and at the same time Minimize VI

Figure 1. Elaborating the concept of Pareto-optimality



The point 'A' marked in Figure 1 is perhaps an appropriate choice for the task 'a',[1] and the task 'b' cannot be performed meaningfully, as the functions III and IV exist in the different regimes of the variable space. A close inspection of the functions V and VI will however reveal that an obvious and unique choice is not possible in case of task 'c'. The three points in B, C, and D marked onto functions V and VI lead to the scenario shown in Table 1.

If we use the information provided in Table 1 for a simple comparison between the points B, C, and D, we immediately realize the fact that if any one of them is *better* than another in terms of one objective, either V or VI, then it is invariably *worse* in terms of the other. Such solutions therefore represent a compromise between various objectives and are termed as *non-dominated* solutions. The other possibility would be one solution *dominating* the other. For that to happen the dominating solution should *at least* be as good as the other in terms of *all* the objective functions, and must fare better in terms of *at least* one. This we specifically call a *weak dominance,* since one can, and some people do, implement a *strong dominance* condition which necessitates that the dominating solutions be better in terms of *all* the objective functions.[2]

Table 1. The non-dominated points

	How good in terms of V?	How good in terms of VI?
B	Better than C, Better than D	Worse than C, Worse than D
C	Worse than B, Better than D	Better than B, Worse than D
D	Worse than B, Worse than C	Better than B, Better than C

To put it simply, Pareto-optimality, the elegant concept proposed by Italian mathematician Vilfredo Pareto (1848-1923), amounts to a quest for the best possible non-dominated solutions. The *Pareto-front*[3] is a representation of the Pareto-optimal points in the functional space and represents the best possible compromises or trade-offs between the objectives. Beyond the original concepts of Pareto (1906), an enormous amount of mathematical analyses have already gone into this subject, of which a comprehensive treatise is provided by Miettinen (1999); readers are referred to it.

In the arena of design and manufacturing, the introduction of Pareto-optimality represents

a paradigm shift as compared to the single-objective optimization that is still quite ubiquitous there. Firstly, based upon just one objective function, not that many reliable decisions can be taken in toady's complex engineering world, and in a real-life design or manufacturing problem, the objectives are very often conflicting to each other. What one needs to work out in a situation like this is the best possible trade-off between these conflicting objectives; mathematically it consists of a multi-objective analysis leading to a Pareto-optimality. One also needs to realize that optimization of a weighted sum of several objectives, which can be performed through a single-objective approach, is at best a point in the Pareto-front and never tells the complete story which would be absolutely essential for an efficient design or a successful manufacturing.

Secondly, the presence of a Pareto-front provides the designers and manufacturers an option to choose from. A decision, based upon their own preference and requirement, can now be made from several alternates of equal merit, where the unique choice provided by a single-objective optimization offers no such flexibility, and could be even quite difficult to implement in a real-life situation.

I will now provide a quick overview of some of the techniques that enable us to compute Pareto-optimality in a genetic way. The readers interested in the classical techniques are once again referred to Miettinen (1999).

THE MULTI-OBJECTIVE ALGORITHMS

The Choice between the Classical and Genetic Methods

The word *classical* is perhaps a cliché that is commonly attributed to a class of algorithms capable of carrying out a multi-criteria optimization task. In general, most of these algorithms are calculus based, many use the gradient information, a lot of them try to render the multi-objective problems into a pseudo-single objective problem, usually by assigning weights, and try to capture one point on the Pareto-front at a time. There are however known exceptions to most of these features in some algorithms belonging to this so-called classical category. Public domain software products belonging to these methods are not that many. An excellent service to the scientific community is however provided by the developers of the highly efficient and user-friendly software NIMBUS (Miettinen & Mäkelä, 1995), by making it freely usable through the Web.[4] In this chapter I will focus however on the evolutionary class of multi-objective algorithms employing the basic concepts of genetic algorithms of diverse kind. Such algorithms are necessarily *non-calculus type,* they use a *population-based approach* instead of a single guess value and its continuous update, and also necessarily, they adopt some *recombination* and *reproduction* procedures in one form or the other. One point needs to be emphasized at this stage that the industrial problems often lack a mathematically differentiable objective function. The solution space is almost inevitably non-smooth, not to mention discontinuous in many cases. Genetic algorithms are geared to negotiate such rough terrains in a very efficient way. Their robustness arises out of the fact that they need not use any gradient information, and their *population*-based computing strategy preempts any requirements for initializing the computation at a close proximity of the actual solutions, which many gradient-based techniques will actually require, granting the evolutionary techniques a distinct edge over their many traditional counterparts.

A Glance at a Few Recent Evolutionary Multi-Objective Algorithms

When it comes to the choice of evolutionary multi-objective algorithms, the users indeed find plenty (Deb, 2001). To create the necessary ambience for describing their usage in the area of design and manufacturing, the pertinent algorithms need to be properly initiated to the readers at large. However, considering the fact that two recent books (Deb, 2001; Coello Coello, Van Veldhuizen, & Lamont, 2002) are extensively dedicated to genetic algorithm-based multi-objective algorithms, and excellent discussions are even available in some of the recent doctoral dissertations (e.g., Okabe, 2004); this discussion at this stage can be essentially stripped to a bare minimum. Therefore, the salient features of only the algorithms that will be mentioned in this chapter from time to time are provided in Table 2.[5] There is no dearth of alternate algorithms capable of doing very similar work, and many of them, as indicated before, are detailed in Deb (2001).

Table 2. Brief introduction to a few multi-objective genetic algorithms

ALGORITHM	What IT DOES?
Strength Pareto Evolutionary Algorithm (SPEA) developed by Zitzler and Thiele (1999)	Works on an elitist strategy and maintains two populations. The genetic operations are carried out on the main. The elite population is controlled within a maximum size by using a tailor-made clustering algorithm, which also adds to its computing burden. In SPEA if an elite member i dominates n_i individuals in the main population of size N, then a *strength* value Γ_i is attributed to it, such that $\Gamma_i = \frac{n_i}{1 + \mathrm{N}}$ (1) SPEA works in a minimization sense, and thus, the individuals with lower strength are considered to be better. Fitness of an *individual* in the main population χ_k, in turn, depends on the strengths of all the elite members that dominate it. For a total of I such dominating members, the fitness here is defined as: $\chi_k = 1 + \sum_{i=1}^{I} \Gamma_i$ (2) The same individual may be present in both the populations, and will receive strength in one instance and fitness in the other. The two populations are temporarily mixed and the main population is recreated through repeated binary tournament selections amongst the combined population members. The new generation of the main population is now once again ranked and the solutions of rank one are copied onto the elite population. Any member in the elite that is dominated by a rank one member of the main population now deleted. Also, the clustering procedure is invoked if necessary. A newer version of this algorithm, known as SPEA II (Zitzler, Laumanns & Thiele, 2002) has done away with the clustering concepts.
Distance based Pareto Genetic Algorithm (DPGA) developed by Osyczka and Kundu (1995)	Works with two populations, an elite population of variable size, and a fixed size main population. In general, δ_{ij}, the normalized Euclidean distance between the i^{th} member of the elite and the j^{th} member of the main population is used for fitness scaling and is calculated as: $\delta_{ij} = \sqrt{\sum_{k=1}^{N} \left(\frac{\mathfrak{I}_k^i - \Re_k^j}{\mathfrak{I}_k^i} \right)^2}$ (3)

Table 2. Brief introduction to a few multi-objective genetic algorithms (cont.)

	where N is the total number of objective functions. $\Im_k^i$ is the k^{th} objective function of the i^{th} member of the elite, and $\Re_k^j$ is the same for the j^{th} member of the main population. The fitness of the j^{th} member in the main population is initiated to an arbitrary value of ϕ and is altered on the basis of δ_j^*, the Euclidean distance between this member and its closest elite neighbor in the functional hyperspace. If this particular member dominates any of the elite members, or remains non-dominated to all, a copy of it is incorporated in the elite population after removing all the dominated elites, and its fitness in both the copies is assigned as $\phi + \delta_j^*$. Alternately, a main population member dominated by any member of the elite gets to stay in the main population with a fitness of $\phi - \delta_j^*$. The main population is subjected to genetic reorganization through the usual route of reproduction, crossover and mutation.
The Non-Dominated Sorting Genetic Algorithms: NSGA and NSGA II (Deb, 2001)	NSGA and NSGA II are actually widely different in terms of their underlying principles. They have been assigned similar acronyms just because they are developed by the same research groups. (Deb, 2001). Among them, NSGA II is currently is one of the most widely used algorithms in its category. NSGA sorts the population based upon dominance, initially assigns a minimum fitness to all the population members and changes them based upon the *Sharing and Niching* concepts well documented in earlier review literature (Chakraborti, 2004). In NSGA II, a new set of solution is however, selected through a *crowded tournament selection* operator, which requires calculation of a *crowding distance* which essentially measures the unoccupied region surrounding a particular solution in the objective function space. A larger crowding distance is favored as it helps to maintain the population diversity, and allows calculation of the maximum spread of the Pareto-front. On the basis of their rank and the crowding distance, the following hierarchy is maintained among the candidate solutions during a tournament selection: 1. A solution of better rank wins the tournament. 2. Between two solutions of same rank, the one with larger crowding distance is favored.
Pareto Converging Genetic Algorithm (PCGA) developed by Kumar and Rockett (2002)	PCGA uses a number of *tribes* containing small populations tend to evolve in isolation. *Immigration* is allowed on a periodic basis. It's a greedy scheme where after the crossover and mutation, the offspings are pitted against the original population, and accepted if only they are better. Rank-based pseudo fitness is assigned to the population members and selection is conducted using a Roulette wheel approach. It has a specific convergence criterion which is fixed through a *Rank Histogram*.

ON MULTI-OBJECTIVE DESIGN

What a Good Design Requires

When an engineer designs something, be it a simple needle or a space rocket, the first thing that she/he has in mind is its utility. Whatever the object is, it should serve the purpose for which it is being built. Although this may be the vital criterion, it alone may not be adequate for a good design. Several other criteria, including a few abstract ones, aesthetic requirements for example, may have to be optimized. For example, one may not like to live in a house that is built for comfort, but otherwise looks like a garbage dump! Even to achieve the prime objective in this case, comfortable living, a number of other objectives need to be satisfied. The planning of the house should include optimal utilization of space; it should have adequate ventilation, proper heating and cooling systems as applicable, and so on. In essence, a good design in this case, like most others, is actually a multi-objective problem. As expected, the objective function space here is often non-differentiable. Since no mathematical function can be possibly constructed that demarcates a good-looking house from an ugly looking one,

the evolutionary approach often becomes not just the best, but the only viable alternate. Multi-objective evolutionary design problems can be constructed virtually from every discipline. Designing a chocolate bar is actually a multi-objective problem as much as is designing a superalloy for the advanced aerospace applications. The first was elegantly discussed by Bagchi (2003), and the second recently became the topic of some very extensive research work by Egorov-Yegorov and Dulikravich (2005). In order to provide the readers a glimpse of the advanced engineering design world where the multi-objective evolutionary approaches are very steadily gaining ground, I will now briefly focus on a few materials design problems. The evolutionary methodology adopted in those studies is generic enough to be applicable to any discipline, wherever the reader's interests are.

Designing Advanced Materials: Few Select Examples

The lifeline for any manufacturing unit would be a proper design of the material that it uses. One can optimally design a material based upon the property requirements. It is possible to take up such a job experimentally. However, quite often the job becomes very cumbersome and tends to get out of hand. Take for example the case of the so-called superalloy design. Such materials contain several elemental components; only for some specific compositions they exhibit their required physical and mechanical properties. In a recent study dealing with superalloys meant for the hot sections of a jet engine (Egorov-Yegorov & Dulikravich, 2005), it has been correctly pointed out that coming up with the alloy composition experimentally for this system would require costly experimentation for about 10 years, whereas the aircraft manufacturers are coming up with a new aircraft every four to five years! The way a multi-objective evolutionary approach can sort out this problem in a much shorter period of time is elaborated as follows.

Designing a Superalloy

The composition of a total of 17 elements—C, S, P, Cr, Ni, Mn, Si, Cu, Mo, Pb, Co, Cb, W, Sn, Al, Zn, and Ti—need to be adjusted in the superalloy that Egorov-Yegorov and Dulikravich (2005) studied. Based upon the property requirements, they decided to use the following as the objective functions:

1. maximize the stress that it can sustain,
2. maximize the temperature up to which it can be used, and
3. maximize the time that it takes before actually rupturing.

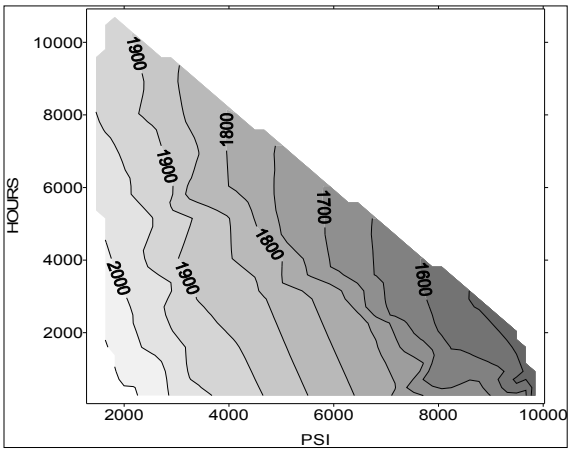
A number of constraints were used along with these objective functions. The evolutionary method that these researchers have used is novel and quite complicated. They call it *indirect optimization based upon the self-organization (IOSO) algorithm*. Along with some basic evolutionary concepts, it makes use of Artificial Neural Nets and Sobol's Algorithm (Sobol, 1976), and it is robust enough to be used for multi-objective problems of diverse kind. A typical 2-D section of the Pareto-frontier computed by these authors is presented in Figure 2.

It is worth mentioning at this stage that a similar alloy design problem for the alloy steels was recently solved by another research group (Mahfouf, Jamei, & Linkens, 2005), which used both single and multi-objective formulations, and found excellent results using SPEA II for the multi-objective problem.

Designing a Magneto-Rheological Fluid

This is an interesting material design problem where both distance-based Pareto genetic al-

Figure 2. Isothermal Pareto sections between hours to rupture and stress (Egorov-Yegorov & Dulikravich, 2005; reproduced by permission of Taylor & Francis, Inc., http://www.taylorandfrancis.com)



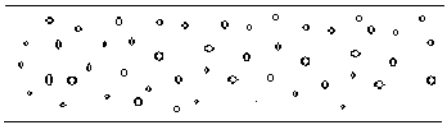
gorithm (DPGA) and strenth Pareto evolutionary algorithm (SPEA) were successfully used (Chakraborti, Mishra, Banerjee, & Dewri, 2003). The basic concept behind *magneto-rheological fluids* is to impregnate a fluid of known viscosity with small particles of magnetic materials. When a magnetic field is applied to such fluids, the magnetic particles align themselves as shown in Figure 3, and the system instantaneously demonstrates a solid-like behavior with measurable values of yield stress and other properties, which again can be varied significantly by adjusting the intensity of the magnetic field and the other parameters, and also can be effectively reversed by turning off the magnetic field. Thus, using these fluids, one can, in principle, instantly create a material of required strength as and when needed. A widespread engineering application of these fluids is currently on the anvil, covering a wide range of devices, be it an aircraft that requires an efficient breaking system or a modern cable bridge that would need protection against high winds and storms.

Two objective functions were used to design this fluid. Among them, the objective function f1 represented the yield stress of the fluid, and the second objective function f2 represented the force needed to separate two adjacent magnetic particles inside it. Four possible design scenarios were next examined through the multi-objective formulations:

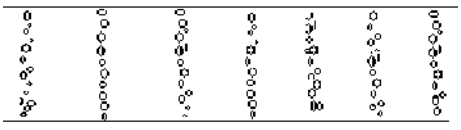
- **Maximize both f1 and f2:** Stable fluid with high yield stress
- **Maximize f1 and Minimize f2:** Unstable fluid of high yield stress
- **Minimize f1 and Maximize f2:** Stable fluid of low yield stress
- **Minimize both f1 and f2:** Unstable fluid of low yield stress

The Pareto-front corresponding to each of these possibilities can provide useful guidelines when it comes to a real-life application. A user or a producer of such fluids may try to operate in the first regime, while trying to avoid the last by all means. For some other applications, one

Figure 3. Magneto-rheological fluid (a) without magnetic field and (b) magnetic field applied

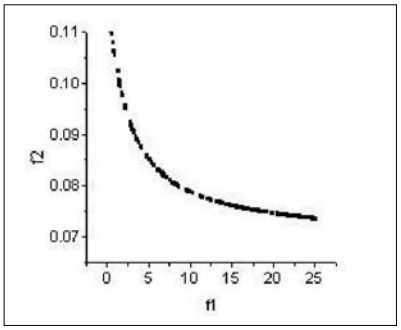


(a)

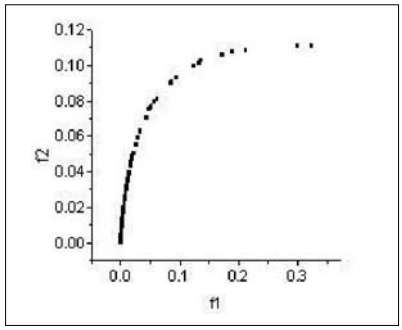


(b)

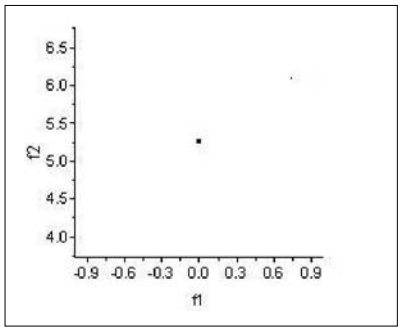
Figure 4. The Pareto fronts for Magneto-rheological fluid (Chakraborti et al., 2003)



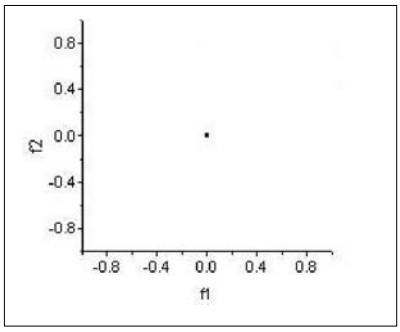
(a) max f1 and max f2



(2) max f1 and min f2



(c) min f1 and max f2



(d) min f1 and min f2

of the two remaining alternates may very well suffice. Even in a particular regime, the corresponding Pareto-front may provide the user with a number of options, which could be very useful in a practical scenario. The computed Pareto-fronts are shown in Figure 4. The results show an interesting trend. In the last two cases the Pareto-fronts have virtually collapsed to a point with little or no spread at all. So the low-yield stress fluid, or in other words the one with low strength, would form only under a very specific condition. The designer is really not left with many options in such a situation, if the target remains designing one such material.

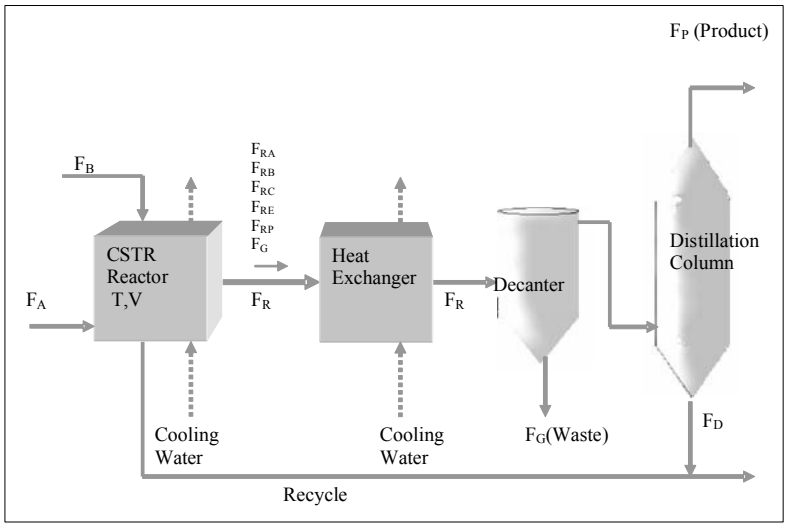
We will now concentrate on some manufacturing applications.

GENETIC PARETO-OPTIMALITY APPLIED TO MANUFACTURING PROCESSES

Regarding the State of the Art

As evident from some recent review articles (Chakraborti, 2002, 2004; Oduguwa, Tiwari, & Roy, 2004), genetic algorithms are now firmly placed in analyzing the manufacturing processes of diverse kinds, pertinent to various industries. A substantial amount of such studies are multi-objective in nature. The scope of this chapter is not to present an extensive survey of what has been done in this area. I would rather demonstrate some paradigm applications to industries of different type. In this connection I

Figure 5. A schematic representation of the Williams and Otto Chemical Plant



will pick up one study of a chemical plant, two from the metal industry, and a fourth one applied to a paper mill. We begin with the chemical plant problem.

Multi-Objective Analysis of a Chemical Plant

The William and Otto Chemical Plant shown schematically in Figure 5 is a typical test problem in the literature as it bears a striking resemblance to real-life chemical processing plants, offering enough complexities to act as a test bed for various non-linear programming (NLP) algorithms and their relative performance.

Here the chemical reaction is carried out in a continuous stirred tank reactor (CSTR). The product stream releases heat through a heat exchanger, and the by-product G is separated in a decanter that follows it. This by-product is subjected to a waste treatment process, adding to the operational cost. The overflow is treated in a distillation column, where the product P is produced. A part of the distillation column underflow is recycled into the CSTR unit, and the rest is used as a plant fuel.

Very recently the William and Otto Chemical Plant was subjected to a multi-objective analysis using PCGA (Chakraborti, Mishra, Aggarwal, Banerjee, & Mukherjee, 2005). The objective of this study was to maximize the annual return on the fixed investment of the plant, and to simultaneously minimize a large number of system constraints required for the plant's proper functioning. The annual return of this plant depends on a number of factors. Those include (1) sales volume, (2) raw material cost, (3) waste treatment cost, (4) utilities cost, (5) sales, administration, and research expenses, (6) plant fixed charge, and (7) plant investment. The objective functions treated all of these as variables. The computed Pareto-front is shown in Figure 6.

In earlier studies (Ray & Szekely, 1973) this problem was tackled as a constrained single-objective problem, and some gradient-based techniques were used to obtain the solution. This often worked unsatisfactorily for an NLP system of such complexity, and the extent of constraint violation was often unacceptable. The evolutionary multi-objective analysis provided a much more amicable solution, elaborat-

Figure 6. The Pareto-front for Williams and Otto Chemical Plant (Chakraborti et al., 2005)

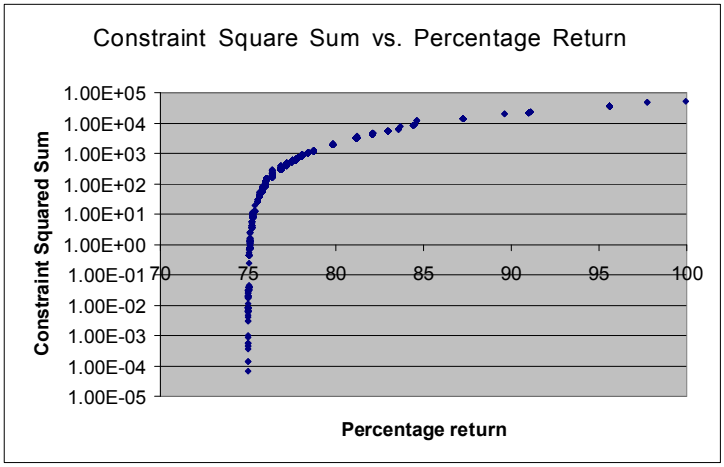
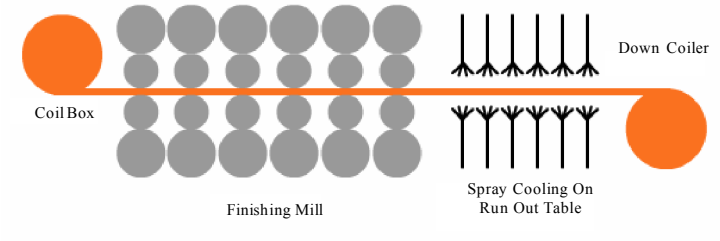


Figure 7. The finishing mill in a hot rolling facility



ing the precise nature of the trade-off between the percentage return and the corresponding violation of constraints.

An Example from the Steel Industry

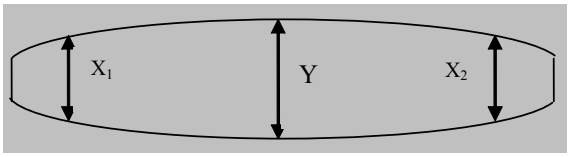
Thick slabs are rolled into sheets of small thickness in most integrated steel plants and the present example pertains to it. A roughing mill does the initial reduction in thickness and the final dimensions are achieved in a finishing mill, shown schematically in Figure 7.

The strips that come out of the rolling mill are not exactly flat. The difference between the thicknesses at the center and the edge leads to the so-called *crown* in the rolled sheets as elaborated in Figure 8.

$$\subset = Y - \frac{(X_1 + X_2)}{2}$$

The crown in the rolled sheets is imparted from various sources. The rolls themselves have some curvature, which changes further during the rolling operation because of thermal expansion and the wear of the rolls. The sequence in which the slabs of different thickness and other properties are fed to the rolls also plays a very important role in controlling the crown, as it affects the roll wear. In order to save the rolled sheet from rejection, its crown needs to be controlled within an acceptable limit. In a recent study done in collaboration with a reputed steel company (Nandan et al.,

Figure 8. Crown (⊂) and its measurement



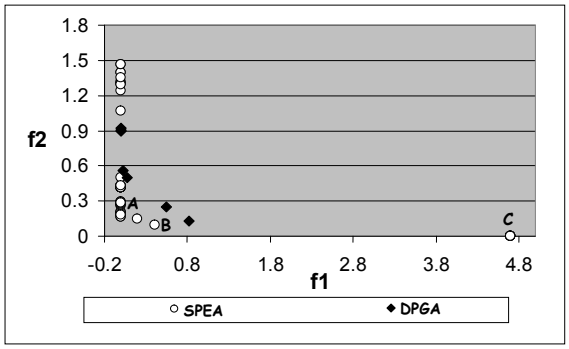
2005), this problem was handled in a multi-objective fashion using both DPGA and SPGA. There attempts were made to minimize the crown from all sources, simultaneously adjusting the sequence of rolling in such a fashion that the wear and the related damages in the rolls are also kept to a minimum. A typical Pareto-front is shown in Figure 9. In general, for this problem DPGA converged faster than SPEA and both gave acceptable results. However, SPEA is able to demonstrate a better spread in the Pareto-front (e.g., the point C in Figure 9) and also some better non-dominated points (e.g., the points A and B marked in the same figure).

On Three-Dimensional Guillotine Cutting

Guillotine cutting involves cutting a piece of metal from edge to edge and is a technique of immense practical importance in many manufacturing processes. A three-dimensional extension of the guillotine cutting would deal with the optimal arrangement of cuboidal orders on a master cuboid shown in Figure 10; an optimal cutting procedure is demonstrated in Figure 11.

Owing to considerable computing complexities, a multi-objective treatment of such problems, until recently, has remained virtually intractable. Only now this problem could be meaningfully addressed to using both DPGA and SPEA (Vidyakiran, Chakraborti, & Mahanty, 2005). What was introduced in this

Figure 9. Pareto-fronts computed for the rolling problem using SPEA and DPGA for 500 generations. The objective functions f1 results in minimizing roll damage while f2 tends to minimize crown (Nandan et al., 2005).



study is a method of evolving the optimal layout for any particular *order* consisting of the task of producing a number of cuboids of various size and shape from a master cuboid. Two objective functions were introduced. The idea *was* to seek a configuration that would produce a minimum amount of scrap for a minimum number of *turns,* defined as the number of times the master block needs to be reoriented, in order to cut it from edge to edge. The first criterion ensured good economics, and the sec-

Figure 10. An optimal master cuboid with five orders (Vidyakiran et al., 2005)

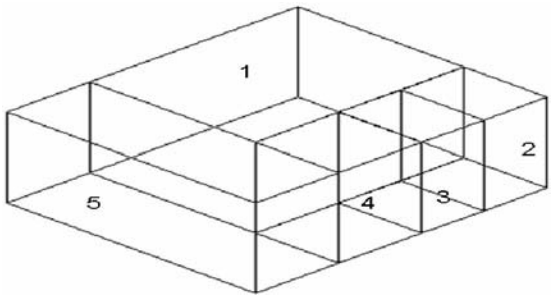
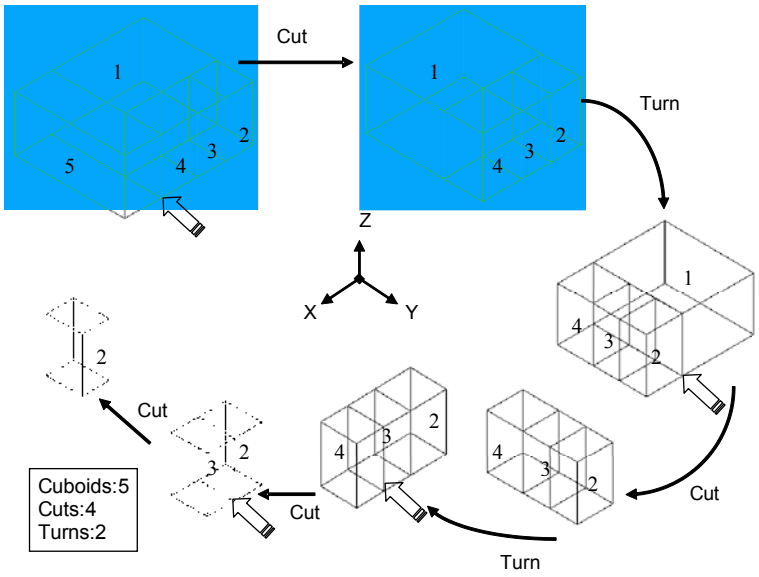


Figure 11. An optimal cutting procedure (Vidyakiran et al., 2005)



ond one was essential to speed up production. These two objectives are, however, conflicting in nature. The task of producing the minimum amount of scrap may not be possible with very few turns of the master block and vice versa.

In this study SPEA performed better than DPGA. Some typical Pareto-fronts are shown in Figure 12.

An Example from the Paper Industry

In a recent work (Toivanen, Hämäläinen, Miettinen, & Tarvainen, 2003) NSGA has been applied for a multi-objective designing the *headbox* of a paper making machine. The headbox, as shown in Figure 13, is located in a strategic position in the paper mill. It receives the pulp slurry from the storage tank and feeds an endless wire screen through some controlled openings. The slurry gets drained in the wire screen forming the *fiber mat,* and most of the remaining water gets removed while it subsequently passes through the rubber roll press. The paper ultimately gets dried and assumes its final form while it is fed to the so-called *yankee,* which is essentially a very large heated cylinder with a highly polished surface. After this stage the paper is wound into a large *parent roll* feeding the *rewinders where* it is cut and rewound into smaller rolls.

Figure 12. Pareto-fronts calculated by DPGA and SPEA for a 10 cuboid order. The probabilities of crossover and mutation were 0.6 and 0.1 respectively (Vidyakiran et al., 2005)

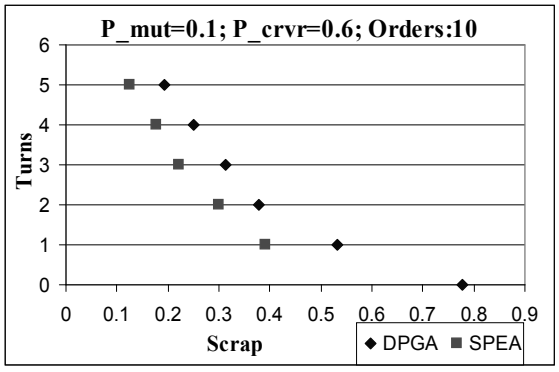
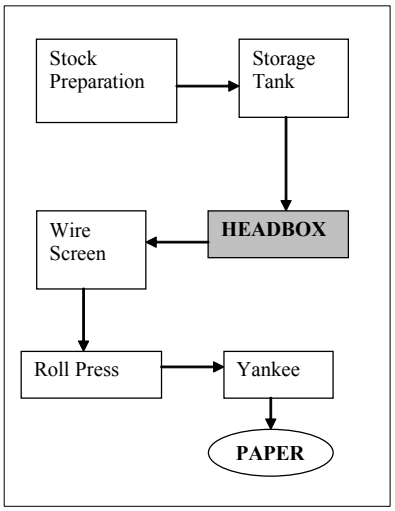


Figure 13. Location of the headbox in the paper mill

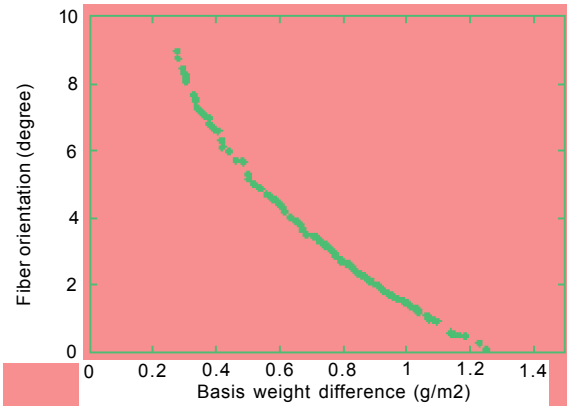


In this entire process a crucial role is played by the headbox. It controls the mass distribution of the fibers and also their orientation. This in turn affects the quality of the finished product in a very significant way. The multi-objective study of the paper mill headbox (Toivanen et al., 2003) has essentially focused on two important properties of the paper, namely the orientation and weight profile. Both of these parameters depend on the nature of the fluid flow in the head box. A solution of Navier-Stokes equation was therefore coupled with an NSGA-based bi-criteria optimization. Not too many design changes are possible in the headbox. However, the geometry of the slice box, the end portion of the headbox through which the slurry exits, could be varied while computing the flow. Some target values were assigned to both the orientation of the fibers and also to the wet-based weight profile. The deviations from both of them were minimized. The first objective function ensured that the basis weight of the paper produced should be even, and the second one ensured that the wood fibers in the paper should be properly oriented; these two objectives are conflicting. A clear-cut Pareto-front resulted in this analysis, as shown in Figure 14, providing an ample choice to work out an acceptable trade-off between the two objectives.

How Genetic Algorithms Handle These Problems

No universal methodology exists in this case. All the practical examples cited in this chapter are problems of significant complexity, and in most cases evaluation of the objective functions remains to be a formidable task and, to accommodate that, the evolutionary algorithms often required some significant adaptations. The strategy that worked in one problem often was not good for another: the nature of encoding the population thus varied from the problem to problem and so did the nature of crossover and mutation.[6] A binary encoding worked well for a number of studies, for example the chemical plant problem as well as for designing the magneto-rheological fluid. In some other work—

Figure 14. Pareto-front in the paper machine headbox study (Toivanen et al., 2003; reproduced by permission of Taylor & Francis, Inc., http://www.taylorandfrancis.com



the paper mill problem, for example—the investigators preferred a floating point encoding. Both the traditional binary and real encoding however were deemed inappropriate for the metal cutting problem described earlier, and this requires further elaboration. There a typical chromosome took the form:

6 10 8 4 2 Y(1) X(-4) 5 Y(3) 9 Z(4) 3 7 X(1)1 Y(1) X(1) X(-6) X(-6)

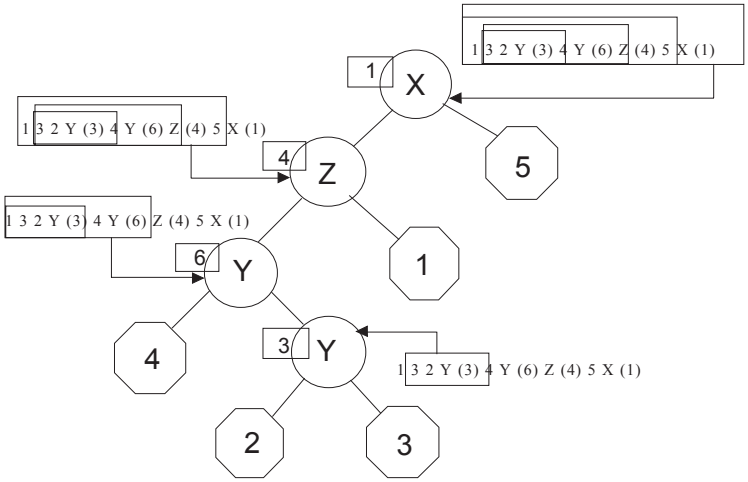
This example denotes a random chromosome representation of 10 cuboids or orders. To decode this, one needs to realize that the number in the parentheses after each axis operator is the orientation number and the remaining numbers represent the cuboids.[7] Such a chromosome can be better understood in the form of a tree. The operators take the nodal positions, and the cuboid numbers are the leaves of the tree. The orientation numbers are specified at each operator.

Thus, for a typical chromosome 1 3 2 Y(3) 4 Y(6) Z(4) 5 X(1), the corresponding tree can be constructed as shown in Figure 15. The readers are referred to the original references for further information.

CONCLUSION

Besides providing an informal but effective treatment of the concept of Pareto-optimality, the idea behind this chapter has been to present, through some select examples, the importance of multi-objective evolutionary treatments for the problems in the domain of design and manufacturing. The examples selected in this chapter from the industries of diverse nature simply demonstrate that the multi-objective problems actually are ubiquitous in the domain of design and manufacturing, and the advantage that an evolutionary approach would offer in solving them is very significant. As for the case studies discussed in this chapter, the problems like metal cutting or the paper machine headbox design would be very cumbersome to address using any gradient-based method. For the Williams and Otto Chemical Plant problem, the genetic algorithms have provided a far better

Figure 15. A typical chromosome and the corresponding tree (Vidyakiran et al., 2005)



insight than what the other techniques could. Similar advantages could be highlighted for most of the examples that I have included here.

It is also reassuring that now one can choose from an ample number of available evolutionary algorithms. A Pareto-front developed through the judicious use of a decision maker is perhaps one of the best things that a manufacturing industry can aspire for, and as we find in the examples cited here, there, in terms of its relevance and efficacy, an evolutionary approach can indeed make a very significant difference.

REFERENCES

Bagchi, T. P. (1999). *Multi-objective scheduling by genetic algorithms*. Boston: Kluwer Academic.

Chakraborti, N., Mishra, P. S., Banerjee, A., & Dewri, R. (2003). Genetic algorithms in materials processing: Few select case studies. In G. Bugeda, J.-A. Désidéri, J. Periaux, M. Schoenauer, & G. Winter (Eds.), *Proceedings of the International Congress on Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems* (EUROGEN 2003). Barcelona: CIMNE.

Chakraborti, N., (2002). Genetic algorithms in ferrous production metallurgy. *Surveys on Mathematics for Industry, 10*(4), 269-291.

Chakraborti, N. (2004). Genetic algorithms in materials design and processing. *International Materials Reviews, 49*(3-4), 246-260.

Chakraborti, N., Mishra, P., Aggarwal, A., Banerjee, A., &. Mukherjee, S. S. (2006). The Williams and Otto Chemical Plant re-evaluated using a Pareto-optimal formulation aided by genetic algorithms. *Applied Soft Computing, 6*(2), 189-197.

Coello Coello, C. A., Van Veldhuizen, D. A., & Lamont, G. B. (2002). Evolutionary algorithms for solving multi-objective problems. New York: Kluwer Academic.

Egorov-Yegorov, I. N., & Dulikravich, G. S. (2005). Chemical composition design of superalloys for maximum stress, temperature and time-to-rupture using self-adapting response surface optimization. *Materials & Manufacturing Processes, 20*(3), 569-590.

Kumar, R., & Rockett., P. (2002). Improved sampling of the Pareto-front in multi-objective genetic optimizations by steady-state evolution: A Pareto converging genetic algorithm. Evolutionary Computation, *10*(3) 283-314.

Mahfouf, M., Jamei, M., & Linkens, D. A. (2005). Optimal design of alloy steels using multi-objective genetic algorithms. *Materials & Manufacturing Processes, 20*(3), 553-567.

Miettinen, K., & Mäkelä, M. M. (1995). Interactive bundle-based method for non-differentiable multi-objective optimization: NIMBUS. *Optimization, 34*, 231-246.

Miettinen, K. (1999) *Non-linear multi-objective optimization*. Boston: Kluwer Academic.

Nandan, R., Jayakanth, R., Moitra, S., .Rai, R., Mukhopadhyay, A,. & Chakraborti, N. (2005). Regulating crown and flatness during hot rolling: A multi-objective optimization study using genetic algorithms. *Materials & Manufacturing Processes, 20*(3), 459-478.

Oduguwa, V., Tiwari, A., & Roy, R. (2005). Evolutionary computing in manufacturing industry: An overview of recent applications. *Applied Soft Computing, 5*(3), 281-299.

Okabe, T. (2004). *Evolutionary multi-objective optimization: On the distribution of offspring in parameter and fitness space*. Aachen: Shaker Verlag GmbH.

Osyczka, A., & Kundu, S. (1995). A new method to solve generalized multi-criteria optimization problems using the simple genetic algorithm. *Structural Optimization, 10*(3), 94-99.

Pareto, V. (1906). *Manual di economia politica.* Translated by A. S. Schwizer, London: MacMillan, 1971.

Ray, W. H., & Szekely, J. (1973). *Process optimization with applications in metallurgy and chemical engineering.* New York: John Wiley & Sons.

Sobol, I. M. (1976). Uniformly distributed sequence with an additional uniform property. *USSR Computational Mathematics and Mathematical Physics, 16,* 236-242.

Toivanen, J., Hämäläinen, J., Miettinen, K., & Tarvainen, P. (2003). Designing paper machine headbox using GA. *Materials & Manufacturing Processes, 18*(3), 533-541.

Vidyakiran, Y., Chakraborti, N., & Mahanty, B. (2005). A genetic algorithms-based multi-objective approach for a three-dimensional guillotine cutting problem. *Materials & Manufacturing Processes, 20*(4), 697-715.

Zitzler, E., & Thiele, L. (1999). Multi-objective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257-271.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the strength Pareto evolutionary algorithm for multi-objective optimization. In K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, & T. Fogarty (Eds.), *Evolutionary methods for design, optimization, and control.* Barcelona: CIMNE.

KEY TERMS

Classical Techniques: Multi-objective optimization routines that are not evolutionary in nature. Most are based upon rigid mathematical principles and are commonly gradient based.

Continuous Stirred Tank Reactor (CSTR): A specially designed agitated vessel used for carrying out chemical reactions.

Crown: A quantifier for the surface flatness of the rolled metal sheets.

Guillotine Cutting: Edge-to-edge cutting of materials, most commonly metals.

Magneto-Rheological Fluid: A *smart material* for engineering applications. Its physical and mechanical properties can be tailor-made by adjusting a magnetic field.

Non-Calculus-Type Algorithms: The optimization algorithms that do not require any derivatives or gradient information. Commonly used to describe genetic and evolutionary algorithms.

Pareto-Optimality: The concept of the family of optimized solutions for a multi-objective problem proposed by Vilfredo Pareto (1848-1923).

Superalloy: A specially designed category of multi-component alloys used in advanced engineering applications.

William and Otto Chemical Plant: A hypothetical chemical plant used as a common test problem in chemical engineering literature.

Yankee: A device used in the paper mills which essentially is a very large heated cylinder with a highly polished surface.

ENDNOTES

1 A good example can be created from thermodynamics: this optimum could represent a composition where the entropy gets maximized for a minimum of free energy.

2 My intention here is to demonstrate how the fundamental concepts associated with Pareto-optimality can be conceptualized and demonstrated in an informal fashion. However, formal and mathematically rigorous definitions of all the concepts presented here can be readily provided. For example, we can proceed to define weak dominance as follows:

- Consider Minimizing Objective Functions: $f_i(\mathbf{X})$ i = 1, 2,….., I
- Subject to Constraints: $g_j(\mathbf{X})$ j= 1, 2, ……., J
- where $\boldsymbol{X} = (x_k : k = 1,2,.....,K)$ is a *K-tuple vector of variables,*
- defining an *I-tuple vector of objectives:* $\Omega_i = (f_i : i = 1,2,....,I)$
- the condition for weak dominance between any two objective vectors is taken as: $(\Omega_l \prec \Omega_m) \Leftrightarrow (\forall_i)(f_{il} \leq f_{im}) \wedge (\exists_i)(f_{il} < f_{im})$

Further details are available in Chapter VI of this book.

3 Some people prefer to call it a *Pareto-frontier,* particularly in a situation of more than two objectives, as the best non-dominating front is essentially a boundary between some portions of the feasible and the infeasible regions.

4 It is available at: http://nimbus.mit.jyu.fi/

5 This list by no means is all inclusive and does not constitute an endorsement for these algorithms from the author's side. In fact, these algorithms are selected just becomes of their usage in the applications cited in this chapter.

6 For example, the usual single- and two-point crossovers sufficed in the case of the chemical plant problem, a position-based crossover generated the sequence of slabs in the metal rolling problem, while the authors of the paper mill problem preferred a heuristic crossover. Similarly, the nature of mutation also varied widely from one problem to the other.

7 Here the cuboid or order numbers are represented as integers. The axis operator provides information about the Cartesian axes along which two cuboids align themselves, and the orientation number provides information about the nature of orientation between the two cuboids. Six different orientations were defined for any given axis, as detailed in Vidyakiran et al. (2005).

Section V

Operations and Supply Chain Management

Chapter XXXII
Evolutionary Optimization in Production Research

Christos Dimopoulos
Cyprus College, Cyprus

ABSTRACT

This chapter provides a short guide on the use of evolutionary computation methods in the field of production research. The application of evolutionary computation methods is explained using a number of typical examples taken from the areas of production scheduling, assembly lines, and cellular manufacturing. A detailed case study on the solution of the cell-formation problem illustrates the benefits of the proposed approach. The chapter also provides a critical review on the up-to-date use of evolutionary computation methods in the field of production research and indicates potential enhancements as well as promising application areas. The aim of the chapter is to present researchers, practitioners, and managers with a basic understanding of the current use of evolutionary computation techniques and allow them to either initiate further research or employ the existing algorithms in order to optimize their production lines.

INTRODUCTION

Modern manufacturing companies strive to stay competitive by producing high-quality customized items at the lowest possible cost. In addition, modern companies need to react quickly (be 'agile') to sudden changes in the economic environment. These characteristics can only be achieved through the continuous optimization of all stages of the production process.

The use of exact optimization methodologies in production research is significantly constrained by the computational complexity of a considerable number of problems in this area (Garey & Johnson, 1979). For this reason, the solution of many optimization problems is handled

through the use of non-analytical techniques (heuristics) that are able to find at least sub-optimal solutions in acceptable computational times. An obvious drawback of these methodologies is their inability to escape local optima that are frequently encountered in multimodal solutions' spaces. Meta-heuristic techniques provide mechanisms that enable an algorithm to escape local optima under certain conditions. *Simulated annealing* (Kirkpatrick, Gelatt, & Vecchi, 1985) and *tabu search* (Glover, 1985) are some notable meta-heuristic techniques that have been used widely in the past; however, the most renowned technique is *evolutionary algorithms* (EAs) (Eiben & Smith, 2004).

The aim of this chapter is to provide an introduction on the use of evolutionary computation methods in the area of production research. Initially, a number of EA-based solutions that can be utilized in a number of critical manufacturing optimization areas are presented. A case study application of an evolutionary algorithm for the solution of the well-known cell-formation problem is described in the following section. The chapter continues with a review of the potential benefits, the practical considerations, and the potential hazards of using evolutionary algorithms for the solution of production research problems. The final section of the chapter summarizes the conclusions and highlights areas of future research.

APPLYING EVOLUTIONARY ALGORITHMS IN THE AREA OF PRODUCTION RESEARCH

The scientific field of production research is concerned with the solution of problems encountered in a manufacturing environment. In practice, many of these problems are handled sub-optimally through purpose-based heuristics or rules-of-thumb (e.g., dispatching rules for scheduling problems). Initial applications of evolutionary computation in the area of production research appeared in the beginning of the 1990s. The motivation behind their use was their ease to express a solution in the form of a permutation or a string of parameter values. This is especially useful for the solution of problems in certain optimization areas such as scheduling and design. From that point onwards, the number of EA applications has been rising constantly. More importantly, these applications expanded in all areas of production research and provided the framework for the development of efficient hybrid optimization techniques, as well as multi-objective optimization techniques.

The following paragraphs provide simple examples on how EA-based solutions can be constructed in some critical areas of production research, based on typical applications that have been published in the literature. The list of solution encodings presented in this chapter is used for illustrative purposes only and is by no means exhaustive. In addition, there is no suggestion that these solution encodings perform better than alternative ones that have been proposed in the literature. A discussion on the relative competence of EAs in this area of research is provided in a following section of this chapter. The interested reader can find comprehensive reviews of evolutionary computation applications in the area of production research in the works of Dimopoulos and Zalzala (2000) and Aytug, Khouja, and Vergara (2003).

Production Scheduling

Description

Scheduling is an optimization area with applications in various scientific fields. In the context of production research, scheduling is the part of

the production planning stage concerned with the assignment of job operations on workstations and the allocation of time slots over a specified planning period. The various permutations of parameters such as job types and constraints, number of machines, and optimization objectives give rise to specific instances of scheduling problems. Most scheduling problems are notoriously difficult to solve analytically for non-trivial problem instances due to their computational intractability (Garey, Johnson, & Sethi, 1976).

Designing Evolutionary Algorithms for Scheduling Problems

Scheduling was perhaps the first production research area that attracted the attention of evolutionary computation researchers. The majority of reported applications concern the solution of static job-shop and flowshop scheduling problems, although many other important areas such as dynamic scheduling have been considered.

In a flowshop (sequencing) scheduling problem, all jobs are to be processed by a single machine or through a production line of machines with each job having the same sequence of operations on each machine. It is relatively easy to design an EA for the solution of this type of problem, since a schedule can be encoded as an integer permutation of all jobs to be processed (Murata, Ishibuchi, & Tanaka, 1996):

{2, 3, 6, 5, 1, 4,…}

This EA-encoded candidate solution indicates that job 2 will be processed first, followed by job 3, job 6, and so forth. The typical one-point crossover operator cannot be applied to this encoding scheme since its application might produce infeasible solutions. Fortunately, due to the similarity of this problem to the well-known traveling salesman problem, a significant number of EA recombination operators that have originally been designed for this problem exist. These operators guarantee the efficient searching of the solutions' space as well as the feasibility of resulting offspring.

Designing an EA-candidate solution for job-shop scheduling problems is a more awkward task, since several jobs with precedence constraints on their operations have to be processed on a number of machines. A scheduling solution should provide both the order of the jobs to be processed on individual machines and the time slice allocated to these jobs. Initial EA applications (Burns, 1993) proposed solution encodings in the form of complete schedules. While this approach required no further processing of the solution for the evaluation of its performance, it required special-built operators to ensure the feasibility of offspring solutions. For this reason the majority of reported EA applications employ some form of an indirect solution representation where only the processing sequence of job operations is defined and a schedule builder is responsible for the transformation of the encoded solution into a valid schedule. A typical example of this approach is an indirect mapping of job operations on a string of integer numbers, as suggested by Bierwirth, Mattfeld, and Kopfer (1996):

{J3, J2, J2, J3, J1, J1,…}

This encoding constitutes an indirect representation of a schedule, indicating that the first operation of the third job should be scheduled first, followed by the first operation of the second job, the second operation of the second job, and so forth. As stated earlier, this representation does not automatically create a job schedule since there is no encoding information about the time slice allocated to job operations. The schedule builder is responsible for the

transformation of the encoded solution into a valid schedule based on the number, type, and capacity of available machines and the processing requirements of job operations. While specific recombination operators should be used with this type of solution representation as well, these operators are usually variants of the operators that are used by EAs for the solution of flowshop scheduling problems. An alternative form of indirect solution representation that has been used in the past eliminates the problem of feasibility considerations by employing the concept of dispatching rules, which are widely used in manufacturing practice. The following solution encoding has been proposed by Herrmann, Lee, and Hinchman (1995):

{EDD, FIFO, SPT, …}

where:

EDD: Earliest Due Date dispatching rule
FIFO: First In First Out dispatching rule
SPT: Shortest Processing Time dispatching rule

In this type of representation, each position in the string represents an available machine for the problem considered, and the corresponding value indicates the dispatching rule that should be used for the scheduling of this machine. The previous solution states that machine 1 should be scheduled using the EDD rule, machine 2 should be scheduled using the FIFO rule, and so forth. A schedule builder is again necessary for the construction of a valid schedule.

While academic research in scheduling has focused mainly on the solution of static scheduling problems, practical manufacturing considerations are oriented towards the solution of dynamic scheduling problems. The use of EAs for the solution of dynamic scheduling problems can be facilitated by decomposing the planning period into a number of job windows, each one containing a predefined number of jobs (Fang & Xi, 1997; Bierwirth, Kopfer, Mattfeld, & Rixen, 1995). Scheduling can be performed at the end of a 'job window', or after the occurrence of an unexpected event (machine breakdown, change in due dates, etc.) using a typical EA algorithm similar to the ones described in the previous paragraphs.

The bibliography of EA applications in the field of scheduling is enormous. The review of Cheng, Gen, and Tsujimura (1996) provides a comprehensive survey on the encoding schemes, schedule builders, and operators available for the design of an evolutionary scheduling algorithm.

Using Evolutionary Algorithms in Assembly Line Optimization

Description

Once an assembly sequence of a product has been selected, the production planner must decide on the allocation of assembly operations at individual workstations. This famous problem, widely known as the assembly line balancing problem, is modeled in two main forms, based on the optimization objectives: Type-I problems aim to minimize the assembly cycle time given a fixed number of assembly workstations, while Type-II problems aim to minimize the total number of assembly workstations given a fixed cycle time. Assembly line balancing problems are usually illustrated with the help of precedence graphs. These graphs illustrate the assembly tasks that need to be performed, as well as the precedence constraints between these tasks. Due to the computational intractability of the problem, numerous exact and heuristic algorithms have been proposed for its solution over the last decade (Scholl & Becker, 2003).

Designing Evolutionary Algorithms for Assembly Line Optimization

A considerable number of EA methodologies have been proposed for the solution of the assembly line balancing problem. A standard EA solution representation does not exist since all the encodings that have been proposed in the literature require special repair mechanisms or operators in order to provide valid solutions after the application of recombination operators.

A typical example is the solution representation proposed by Kim, Kim and Kim (2000). In this representation an EA-candidate solution consists of a string of integers with length equal to the total number of tasks in the problem considered:

{1, 2, 1, 2, 3, 3, 4, ….}

The value of each integer indicates the workstation allocation of the task identified by the integer's index position in the string. The previous candidate solution indicates that assembly task 1 is assigned to workstation 1, assembly task 2 is assigned to workstation 2, assembly task 3 to workstation 1, and so forth. The existence of assembly sequence constraints means that this encoding will result in infeasible task assignments when used with typical recombination operators such as one-point crossover. Penalty functions, especially constructed heuristics and decoding procedures, can be used to alleviate this problem.

Another solution representation that can be employed is as an integer permutation of all assembly tasks to be performed (Leu, Matheson, & Rees, 1994):

{2, 1, 3, 4, 5, 7, 6}

The previous string indicates a feasible assembly sequence of seven parts for an assembly line balancing problem. The sequence itself does not contain any information about the assignment of tasks to individual workstations. However, construction schemes that have originally been designed for exact solution methodologies (dynamic programming, branch, and bound) can be employed for this purpose. These schemes progressively assign a given sequence of tasks to workstations on either a task-by-task or a station-by-station basis (task-oriented or station-oriented construction schemes respectively). The application of typical recombination operators does not guarantee the feasibility of offspring solutions when the above encoding scheme is used. Leu et al. (1994) overcome this problem using a modified version of the two-point crossover operator that explicitly considers the precedence feasibility of exchanged tasks between the strings.

Several alternative encoding schemes for the solution of the assembly line balancing problem have been suggested in the literature. Scholl and Becker (2003) provide a review of all assembly line balancing solution methodologies, including a detailed description of all methodologies that are based on the principles of evolutionary computation.

Cellular Manufacturing Systems

Description

Cellular manufacturing is the application of the group technology concept in the area of production research and more specifically in the area of job shop production (Burbidge, 1971). The manufacturing plant is organized into groups of machines (machine cells) that process associated groups of parts (part families). The intuition behind this design is that parts within cells are processed in a single line, realizing in

that way the benefits of mass production within a job-shop environment. The central problem during the design of a cellular manufacturing system is the grouping of machines into cells and parts into associated families, widely known as the cell-formation problem.

Designing Evolutionary Algorithms for Cellular Manufacturing Problems

The cell-formation problem has been a subject of EA studies from the early days of the field. Since the cell-formation problem is essentially a grouping problem and not a sequencing problem, the representations that were discussed in the previous sections are not applicable in this case. The traditional EA solution representation was originally introduced by Venugopal and Narendran (1992). A solution is depicted as a string of integers with length equal to the total number of machines in the plant:

{1, 3, 2, 2, 2, 3,….)

Each string value indicates the cell in which the corresponding machine defined by the integer's index position is currently grouped. A decoding of the previous example will result in machine 1 grouped in cell 1, machine 2 grouped in cell 3, machine 3 grouped in cell 2, and so forth. The application of standardized recombination operators such as one- and two-point crossover could yield solutions containing empty cells. This problem can be overcome with the use of repair mutation operators (Gupta, Gupta, Kumar, & Sundaram, 1996), which progressively scan the solution string and reassign machines to cells until all cells are assigned with at least one machine. The main drawback of the above encoding scheme is that it requires the pre-specification of the total number of cells in the plant as a parameter of the optimization run, a feature that as indicated in the cellular manufacturing literature can hide natural machine groupings. Despite this drawback, the representation has been used in the majority of published EA applications.

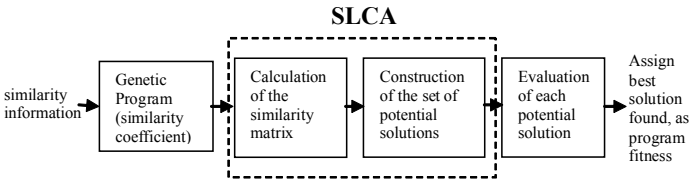
Although the previous encoding provides an efficient way for the representation of solutions in simple versions of the cell-formation problem, it is unable to provide solutions for advanced versions of the cell-formation problem that explicitly consider the existence of alternative process plans for each part in the plant. Gravel, Nsakanda, and Price (1998) have suggested an enhanced version of this encoding scheme in which the main solution string is supported by a second solution string that defines the allocation of specific process plans to individual parts:

{2, 3, 1, 3, 2,.....}

Each part has a predefined number of process plans. The length of the supporting string is equal to the total number of parts for the problem considered. The above string, under a given machine-cell configuration provided by the main solution string, denotes that process plan 2 will be used for the production of part 1, process plan 3 will be used for the production of part 2, and so forth. The application of recombination operators is performed separately on each of the solution strings.

Very few EA solution encodings for the cell-formation problem differentiate from the basic scheme described in the previous paragraphs. A significantly different genetic programming-based encoding scheme has been proposed by Dimopoulos and Mort (2001) for the solution of simple and advanced formulations of the cell-formation problem. A detailed description of this application is presented as a case study in the following section.

Figure 1. The GP-SLCA framework



CASE STUDY: A GENETIC PROGRAMMING METHODOLOGY FOR THE SOLUTION OF THE CELL-FORMATION PROBLEM

GP-SLCA (Dimopoulos & Mort, 2001) is an EA based on the principles of genetic programming that has been custom designed for the solution of simple and advanced formulations of the cell-formation problem. GP-SLCA efficiently combines a genetic programming algorithm with a traditional clustering technique called *single linkage cluster analysis*. Although GP-SLCA does not use a typical encoding for the problem considered, it clearly illustrates potential benefits that can be gained from the use of evolutionary computation algorithms.

GP-SLCA

A Short Description of the GP-SLCA Framework

GP-SLCA is a hybrid EA that incorporates a typical clustering technique—single linkage cluster analysis, or SLCA (McAuley, 1972)—within a genetic programming framework for the solution of simple and advanced formulations of the cell-formation problem. SLCA employs the traditional Jaccard's similarity coefficient in order to calculate a value of similarity between each pair of machines in a simple cell-formation problem. Jaccard's similarity coefficient is defined as follows:

$$S_{ij} = \frac{a_{ij}}{a_{ij} + b_{ij} + c_{ij}}$$

where:

S_{ij} : similarity between machines i and j

a_{ij} : number of parts processed by both machines i and j

b_{ij} : number of parts processed by machine i but not by machine j

c_{ij} : number of parts processed by machine j but not by machine i

The calculated values form a similarity matrix that becomes the basis for the process of creating machine groupings. Initially, all machines are considered ungrouped. The similarity matrix is scanned for the highest similarity value between a pair of machines. The corresponding machines are grouped at this similarity level. The same operation is repeated for the next higher similarity value. The algorithm continues in a similar fashion until all machines are grouped into a single cell. A number of different solutions (machine groupings) are created during this process based on the desired level of similarity between machines (similarity threshold).

GP-SLCA exploits the characteristics of the SLCA technique, but instead of using a pre-defined similarity coefficient, a genetic programming algorithm is responsible for the generation and evolution of a population of similarity coefficients. Each of these coefficients is passed through the SLCA process and a corresponding set of solutions is pro-

Example 1.

Procedure Main

initialise population of randomly created similarity coefficients
run procedure SLCA for each coefficient
loop
    loop
        select individuals for crossover or mutation
        apply genetic operators and form new coefficients
    until a new generation has been formed
    run procedure SLCA for each coefficient
until termination criterion is true

Procedure SLCA

compute similarity matrix
loop
    create machine cells for the highest level of similarity coefficient
    assign parts to machine cells
    calculate the fitness value of the cell configuration
    if solution is the best recorded so far, best=current solution
until a single cell has been formed
assign the best solution found as fitness of the individual

duced. All potential solutions are evaluated, and the solution with the best value in relation to the optimization objective considered is assigned as a fitness value to the genetically evolved similarity coefficient. A diagrammatic explanation of the GP-SLCA framework is illustrated in Figure 1.

The algorithmic explanation of the GP-SLCA framework in pseudo-code form is as seen in Example 1.

Application of GP-SLCA to a Typical Cell-Formation Problem

An illustrated example will be used to provide further understanding of the GP-SLCA operation. Figure 2 depicts a typical medium-sized cell-formation problem originally proposed by Burbidge (1971). Each positive entry in the table indicates that a component of the corresponding column requires a processing operation on the machine of the corresponding row. This test problem has been widely employed in cell-formation literature, and consequently many comparative experimental results are available for comparison purposes.

The set of functions for the genetic programming part of GP-SLCA comprises typical mathematical operations: addition, subtraction, multiplication, and protected division (a function that accommodates the division of zero exception). For the set of terminals, typical inputs like the ones used by Jaccard's similarity coefficients are employed:

a_{ij} : number of parts processed by both machines i and j
b_{ij} : number of parts processed by machine i but not by machine j
c_{ij} : number of parts processed by machine j but not by machine i

Figure 2. Burbidge's cell-formation test problem

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
1																																						1				1	
2		1								1																		1				1					1	1		1		1	
3							1										1																		1	1	1							
4					1				1						1				1		1		1						1															
5					1			1	1						1	1	1		1		1		1						1				1								1		1
6	1	1				1	1	1			1	1	1			1		1					1									1	1	1			1		1	1		1	1
7	1												1												1																		
8	1	1	1					1	1		1	1			1				1	1	1		1	1			1	1			1						1	1			1		1
9		1		1						1								1										1				1					1	1		1		1	
10	1											1	1												1	1				1										1			
11		1							1											1				1			1		1		1												
12											1											1		1			1		1		1												
13		1																						1																		
14	1				1												1																		1							
15					1								1						1		1											1								1		1
16	1					1		1										1													1						1	1			1	

Table 1. Koza tableau for the example GP-SLCA application

Parameters	Values
Objective	maximisation of grouping efficacy
Terminal set	α_{ij}, b_{ij}, c_{ij}, d_{ij} (defined earlier)
Function set	+, -, ×, % (protected division function)
Population size	500
Crossover probability	.9
Mutation probability	.1
Number of generations	50

d_{ij} : number of parts processed by neither machine i nor machine j

The aim of the GP-SLCA process is to evolve combinations of the function and terminal sets (in the form of similarity coefficients) that will subsequently produce optimal or near-optimal cell configurations according to the optimization objective. Note that the choice of these sets is based on the understanding of the problem considered, the optimization objective used, and the intuition of the designer. For example, Dimopoulos and Mort (2001) have successfully used a different terminal set for the solution of a cell-formation problem that explicitly considers the sequence of operations between machines.

In the experimental results presented in this section, the objective of the GP-SLCA algorithm is the maximization of the grouping efficacy value, a measure that is calculated as follows:

Table 2. Cumulative performance of GP-SLCA on the test problem considered

Best value of efficacy recorded	**0.567901**
Number of times this value was found	**5**
Mean best value of efficacy per run	**0.566063**

$$\Gamma = \frac{e - e_0}{e + e_v}$$

where:

Γ: grouping efficacy

e: total number of processing operations

e_0: total number of processing operations performed outside designated cells

e_v: total number of processing operations performed inside designated cells

The design parameters of the GP-SLCA algorithm can be summarized with the help of the so-called Koza (1992) tableau illustrated in Table 1. Note that the protected division function returns the value of 1, when the denominator evaluates to zero.

Since GP-SLCA is a stochastic algorithm, 20 experimental runs were conducted on the problem considered. The cumulative performance of the algorithm is described in Table 2. The best result found by GP-SLCA and the corresponding best results of alternative solution methodologies are illustrated in Table 3.

It can be seen that GP-SLCA provided an improved performance on the problem consid-

Table 3. Comparison of GP-SLCA with alternative solution methodologies on the problem considered

Solution Methodology	*Grouping efficacy*
GP-SLCA	**0.5679**
GRAFICS (Srinivasan & Narendran, 1991)	**0.5439**
ZODIAC (Chandrasekharan & Rajagopalan, 1987)	**0.5376**
GA-TSP (Cheng et al., 1998)	**0.5389**
GA - Integer Programming (Joines et al., 1996)	**0.4926**

Figure 3. Genetically evolved similarity coefficient that produced the machine grouping with the best performance

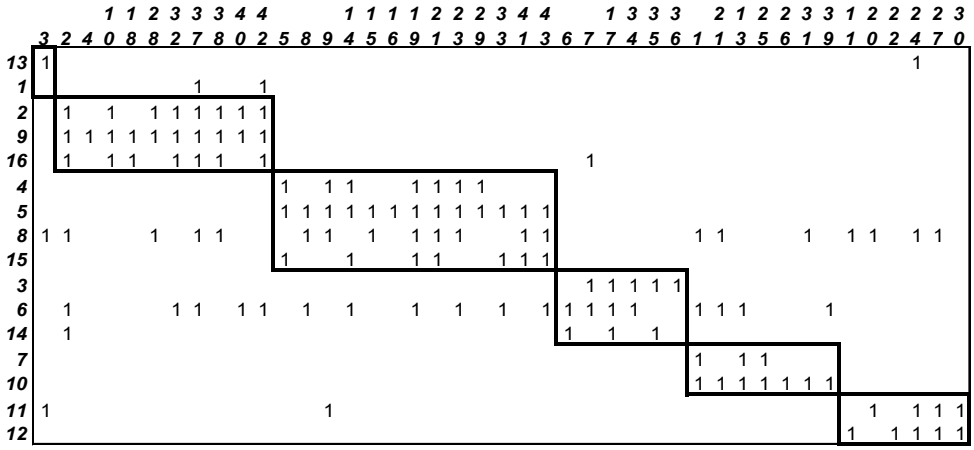
$$a+d+\frac{d}{b}-\frac{a}{b}-2a(b-a)+\left(\frac{ad}{b+c}\right)+\frac{2a(b-a)}{d\left[2a-\frac{c\left(\frac{ab}{d/b}+d\right)}{a^2}\right]-\frac{2d+c}{a^2}+\frac{\left[a+\frac{a+d-\left(\frac{a/b}{d-b}\right)}{(a+b)/d}\right]}{a}}$$

ered, in relation to alternative optimization methodologies, both in terms of the average and maximum values of grouping efficacy attained. These results are only indicative of the framework's performance. A detailed analysis of the above algorithm, as well as substantial experimental results on the entire range of test problems for which comparative results exist, can be found in Dimopoulos and Mort (2001).

A closer examination of the solution that created the best value of grouping efficacy reveals that it was generated through the SLCA procedure from the genetically evolved similarity coefficient illustrated in Figure 3.

The structure of this coefficient is much more complicated in comparison to typical man-made similarity coefficients; however, it has been evolved with the sole purpose of finding an optimal grouping for the specific cell-formation problem (it does not generalize to other problem instances). A run of the SLCA algorithm using the above coefficient results in a set of potential machine groupings. One of these solutions produces the best value of grouping efficacy for the problem considered ("=0.5679). A graphical illustration of this solution is depicted in Figure 4. Note that each rectangle on the matrix designates a cell of machines and associated parts.

Figure 4. Cell configuration that produces the best objective function value for Burbidge's test problem (Γ=0.5679)



A DISCUSSION ON USING EVOLUTIONARY ALGORITHMS FOR THE SOLUTION OF PRODUCTION RESEARCH PROBLEMS

In the previous sections an introduction on the use of evolutionary computation methods in the area of production research was provided. This description can serve as a basis for discussing the relative competency, the benefits, and the practical considerations of using evolutionary computation methods in this area of research.

Competency of Evolutionary Algorithms in the Area of Production Research

A wide range of tools exist that can help engineers with the task of optimizing the production process. The choice of a particular tool is based on various parameters such as relative efficiency, computational complexity, ease of implementation, availability as a commercial product, computational requirements, connectivity with other software products, business policy, and human factors. While researchers measure the competency of their algorithms in relation to alternative optimization methodologies using a single optimization objective, this characteristic alone does not guarantee their adoption in practical manufacturing environments.

The relative competency of EAs as a generic optimization methodology is very difficult to assess, especially since particular instances of the algorithms differ significantly in the representation scheme, recombination operators, and the parameters used during the evolutionary process. Empirical results (Dimopoulos & Zalzala, 2000; Aytug et al., 2003) indicate that various instances of EAs have been frequently (but not always) producing results that are competitive to alternative optimization methodologies in the area of production research. This fact indicates that EAs can be used with a certain degree of safety in terms of their optimization capabilities.

Another dimension that should be considered is the consistency of EA results. Since EAs are stochastic optimization algorithms, it cannot be guaranteed that an optimization run will yield an optimal or a near-optimal solution. While published results do not indicate alarming differences in the performance between various optimization runs, it would be difficult for an engineer to employ EAs in cases where performance guarantees are absolutely critical. However, it should be noted that during the last decade, EA researchers have made significant advances in producing performance guarantees for specific instances of EAs in specific problem cases (Eiben & Rudolph, 1999).

A valid criticism of EA research in the past is the failure to consistently compare the computational cost of EAs in relation to alternative optimization methodologies. The reality is that most instances of EAs are very fast. There are specific instances of EAs (like some genetic programming implementations) that might require significant computational resources. However, the issue of computational cost has to be evaluated in relation to the problem considered. While the speed of an algorithm is very important in the case of a dynamic scheduling problem, it becomes less important in the case of a facility layout problem. The algorithm described in the previous section attempts to provide an efficient design for a cellular manufacturing system that, once evolved, may be used for a period of years. In this case the focus is certainly on the quality of the solution rather than the computational complexity of the algorithm.

Potential Benefits

The most important benefits of using EAs in the area of production research stem from their

unique problem-solving approach. The main attraction of EAs is their flexibility in solution representation and their simple fitness assignment scheme. EAs provide engineers with a non-mathematical way to design solutions that correspond to realistic considerations of specific manufacturing plants. In addition, EAs provide a number of standardized techniques for handling all types of constraints that can exist in practical manufacturing optimization problems. While these characteristics alone do not guarantee the optimality of solutions, they provide the basis for the design of robust algorithms, especially in cases where a mathematical description of the problem considered is either not possible or computationally intractable.

More importantly, the flexibility in the basic evolutionary process means that they can be combined very efficiently with alternative optimization methodologies. These EA-based hybrid algorithms provide optimization frameworks that exploit the positive characteristics of each methodology combined. Experimental results indicate that this approach constitutes a very promising direction for the solution of difficult optimization problems in this area of research.

The main area where EAs seem to have a competitive advantage in relation to alternative optimization methodologies is the area of multi-objective optimization. The existence of production research problems that aim to simultaneously optimize multiple objectives of contradicting nature has been largely ignored in the production research literature (apart from the bi-objective case), despite the fact that they represent a realistic modeling of manufacturing practice. EAs provide the environment and a set of tools that allow the simultaneous evolution of a set of solutions from which the decision maker can make an informed choice (Coello Coello, 1999).

The benefits that EAs can provide to production engineers are clearly illustrated in the case study presented in this chapter: GP-SLCA efficiently hybridizes a traditional clustering technique with a genetic programming algorithm into a robust optimization algorithm for the solution of cell-formation problems. At the same time, it can be easily modified to explicitly consider cell-size or other type constraints (Dimopoulos & Mort, 2000). Advanced versions of the cell-formation problem can also be considered by modifying the objective function of the algorithm or by using different types of similarity inputs (Dimopoulos & Mort, 2000). Finally, GP-SLCA has been successfully combined with the NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002) evolutionary multi-objective optimization technique, providing a framework for the solution of large-sized multi-objective cell-formation problems (Dimopoulos, 2005).

Practical Considerations

Despite the considerable number of EA-related publications in the area of production research, their adoption rate in manufacturing environments has been very slow. This is due to a number of considerations that make industrial engineers and managers reluctant to use them. One of these considerations is the biology-related background of EAs. While a detailed knowledge of the theory of evolution is not needed for the design of an EA, since EAs employ a loose interpretation of this theory, the use of the biology-inspired terminology predisposes engineers for the opposite.

Another major consideration is the fact that the implementation of an EA from 'scratch' requires programming skills that a significant number of managers or industrial engineers do not possess. Unfortunately, only a small number of commercial implementations of EAs exist either as independent applications or as parts of a larger software platform in the area of production research that can help engineers to readily exploit and appreciate the benefits of their use.

Finally, a practical consideration that is regularly noted in EA literature is the unavailability of guidelines for setting the parameters of EA optimization runs (population size, number of generations, probability of applying genetic operators, selection mechanism, etc.). Parameter setting in EAs is itself a difficult optimization problem. Since in the majority of cases there is no mathematical background for a scientific calculation of these parameters, experience and trial optimization runs normally provide the basis for parameter setting. Unfortunately, the sensitivity of an algorithm's performance to the change of parameters is very much problem dependent.

CONCLUSION AND FUTURE RESEARCH

In the previous sections a basic introduction on the principles of evolutionary computation methods and their use in the area of production research was provided. EAs constitute a useful tool in the hands of managers, production designers, and engineers. This chapter does not aim to overemphasize their competency in relation to alternative optimization methodologies. On the contrary, as it has been discussed earlier, significant benefits are gained when EAs are combined with alternative optimization methodologies and when problem-specific information is incorporated in their design. In addition, EAs provide a natural framework for the solution of multi-objective production research problems that are very difficult to approach with traditional optimization methods.

A considerable amount of EA applications covering the entire production research field have been reported over the last decade. The next challenge for researchers in this area is to transform their basic algorithms that provide promising experimental results on literature test problems into software packages that can be readily used in realistic production environments. This step requires the cooperation of academic experts and production engineers, as well as the necessary managerial support. It is hoped that reviews like the one provided in this chapter can help production engineers and managers to understand the operation of evolutionary computation methods as well as the potential benefits of their use in this area of research.

REFERENCES

Aytug, H., Khouja, M., & Vergara, F. E. (2003). Use of genetic algorithms to solve production and operations management problems: A review. *International Journal of Production Research*, *41*(17), 3955-4009.

Bierwirth, C., Mattfeld, D. C., & Kopfer, H. (1996). On permutation representations for scheduling problems. *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature* (pp. 310-318).

Bierwirth, C., Kopfer, H., Mattfeld, D. C., & Rixen, I. (1995). Genetic algorithm based scheduling in a dynamic manufacturing environment. *Proceedings of the 1995 IEEE Conference on Evolutionary Computation* (pp. 439-443).

Burbidge, J. L. (1971). Production flow analysis. *Production Engineer, 50,* 139-152.

Burns, R. (1993). Direct chromosome representation and advanced genetic operators for production scheduling. *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 352-359).

Chandrasekharan, M. P., & Rajagopalan, R. (1987). ZODIAC—an algorithm for concurrent formation of part families and machine-cells. *International Journal of Production Research, 25*(6), 835-850.

Cheng, C. H., Gupta, Y. P., Lee, W. H., & Wong, K. F. (1998). A TSP-based heuristic for forming machine groups and part families. *International Journal of Production Research, 36*(5), 1325-1337.

Cheng, R., Gen, M., & Tsujimura, Y. (1996). A tutorial survey of job-shop scheduling problems using genetic algorithms. Part I: Representation. *Computers and Industrial Engineering, 30*(4), 983-997.

Coello Coello, C. A. (1999). A comprehensive survey of evolutionary-based multi-objective optimization techniques. *Knowledge and Information Systems, 1*(3), 269-308.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182-197.

Dimopoulos, C. (2005, July 21-26). A genetic programming methodology for the solution of the multi-objective cell-formation problem. *Proceedings of the Joint Conference on Information Systems* (JCIS 2005), Salt Lake City, UT.

Dimopoulos, C., & Mort, N. (2000). Solving cell-formation problems under alternative quality criteria and constraints with a genetic programming-based hierarchical clustering algorithm. *Proceedings of the 6th International Conference on Control, Automation, Robotics and Vision* (pp. 3445-3446).

Dimopoulos, C., & Mort, N. (2001). A hierarchical clustering methodology based on genetic programming for the solution of simple cell-formation problems. *International Journal of Production Research, 39*(1), 1-19.

Dimopoulos, C., & Zalzala, A. M. S. (2000). Recent developments in evolutionary computation for manufacturing optimization: Problems, solutions and comparisons. *IEEE Transactions in Evolutionary Computation, 14*(2), 93-113.

Eiben, A. E., & Smith, J. E. (2003). *Introduction to evolutionary computing*. Berlin: Springer-Verlag.

Eiben, A. E., & Rudolph, G. (1999). Theory of evolutionary algorithms: A bird's eye view. *Theoretical Computer Science, 229*(1-2), 3-9.

Fang, J., & Xi, Y. (1997). A rolling horizon job shop rescheduling strategy in the dynamic environment. *International Journal of Advanced Manufacturing Technology, 13*(3), 227-232.

Fogel, L. J., Owens, A. J., & Walsch, M. J. (1966). *Artificial intelligence through simulated evolution*. New York: John Wiley & Sons.

Garey, M., & Johnson, D. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. San Francisco: W. H. Freeman.

Garey, M., Johnson, D.S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research, 1,* 117-129.

Glover, F. (1990). Tabu search: A tutorial. *Interfaces, 20*(3), 79-94.

Gravel, M., Nsakanda, A. L., & Price, W. (1998). Efficient solutions to the cell-formation problem with multiple routings via a double-loop genetic algorithm. *European Journal of Operational Research, 109,* 286-298.

Gupta, Y., Gupta, M., Kumar, A., & Sundaram, C. (1996). A genetic algorithm-based approach to cell-composition and layout design problems. *International Journal of Production Research, 34*(2), 447-482.

Herrmann, J. W., Lee, C. Y., & Hinchman, J. (1995). Global job-shop scheduling with a genetic algorithm. *Production & Operations Management, 4*(1), 30-45.

Holland, J. H. (1975). *Adaptation in natural and artificial systems.* Ann Arbor: University of Michigan Press.

Joines, J. A., Culbreth, C. T., & King, R. E. (1996). Manufacturing cell design: An integer programming model employing genetic algorithms. *IIE Transactions, 28*(1), 69-85.

Kim, Y. J., Kim, Y., & Kim, Y. J. (2000). Two-sided assembly line balancing: A genetic algorithm approach. *Production Planning and Control, 11,* 44-53.

Kirkpatrick, S., Gelatt Jr., C. D., & Vecchi, M. P. (1985). Optimization by simulated annealing. *Science, 220,* 671-679.

Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection.* Cambridge, MA: MIT Press.

Leu, Y. Y., Matheson, L. A., & Rees, L. P. (1994). Assembly line balancing using genetic algorithms with heuristic-generated initial populations and multiple evaluation criteria. *Decision Sciences, 25,* 581-606.

McAuley, J. (1972). Machine grouping for efficient production. *Production Engineer, 51*(2), 53-57.

Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Genetic algorithms for flowshop scheduling problems. *Computers and Industrial Engineering, 30*(4), 1061-1071.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen evolution.* Stuttgart: Frommann-Holzboog.

Scholl, A., & Becker, C. (2003). State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research, 168,* 666-4009.

Srinivasan, G., & Narendran, T. T., (1991). GRAFICS—a nonhierarchical clustering algorithm for group technology. *International Journal of Production Research, 29*(3), 463-478.

Venugopal, V., & Narendran, T. T. (1992). A genetic algorithm approach to the machine-component grouping problem with multiple objectives. *Computers & Industrial Engineering, 22*(4), 269-480.

Chapter XXXIII
Ant Colony Optimization and Multiple Knapsack Problem

Stefka Fidanova
Bulgarian Academy of Science, Bulgaria

ABSTRACT

The ant colony optimization algorithms and their applications on the multiple knapsack problem (MKP) are introduced. The MKP is a hard combinatorial optimization problem with wide application. Problems from different industrial fields can be interpreted as a knapsack problem including financial and other management. The MKP is represented by a graph, and solutions are represented by paths through the graph. Two pheromone models are compared: pheromone on nodes and pheromone on arcs of the graph. The MKP is a constraint problem which provides possibilities to use varied heuristic information. The purpose of the chapter is to compare a variety of heuristic and pheromone models and different variants of ACO algorithms on MKP.

INTRODUCTION

Combinatorial optimization is a process of finding the best or optimal solution for problems with a discrete set of feasible solutions. Applications occur in numerous settings involving operations management and logistics. The economic impact of combinatorial optimization is profound, affecting diverse sections. While much progress has been made in finding exact solutions to some combinatorial optimization problems (COPs), many hard combinatorial problems (NP-problems) are still not exactly solved in a reasonable time and require good meta-heuristic methods. The aim of meta-heuristic methods for COPs is to produce quickly good-quality solutions. In many practical problems they have proved to be effective and efficient approaches, being flexible to accommodate variations in problem structure and in the objectives considered for the evaluation of solutions (Lonnstedt, 1973). For all these rea-

sons, meta-heuristics has probably been one of the most stimulated research topics in optimization for the last two decades. Examples are decision making problems.

The ant colony optimization (ACO) is a new meta-heuristic method. ACO algorithms are applied in real life and industrial problems for which a good solution for a short time is required. ACO achieves good results for problems with restrictive constraints like multiple knapsack problem. It represents a multi-agent system where low-level interaction between single agents results in a complex behavior of the whole ant colony. It imitates the behavior shown by real ants when searching for food. Ants are social insects that live in colonies and whose behavior is aimed more to the survival of the colony as a whole than to that of a single individual component of the colony. An important and interesting aspect of ant colonies is how ants can find the shortest path between the food sources and their nest. Ants communicate information about food sources via a chemical substance called pheromone, which the ants secrete as they move along.

Analogously, ACO is based on the indirect communication of a colony of simple agents, called "artificial" ants, mediated by "artificial" pheromone trails. The pheromone trails in ACO algorithms serve as distributed numerical information, which ants use to probabilistically construct solutions to the problem to be solved and which ants adapt during the algorithm's execution to reflect their search experience. Artificial ants not only imitate the behavior described, but also apply additional problem-specific heuristic information. The idea is developed by Moyson and Manderick (1988). The first example of ant algorithm is Ant System (Dorigo, Maniezzo, & Colorni, 1996), and it has been applied to and provided solutions for various hard combinatorial optimization problems. Recently, different versions of the ACO algorithms such as the ant colony system (Dorigo, 1999a), the ant algorithm with elitist ants (Dorigo, 1999b), the max-min ant system (Stützle & Hoos, 2000), the ant algorithm with additional reinforcement (Fidanova, 2002), and the best-worst ant system (Cordón, Fernàndez de Viana, & Herrera, 2002) have been applied to many optimization problems. Examples are the traveling salesman problem (Dorigo, 1999a), the quadratic assignment (Gambardella, Taillard, & Agazzi, 1999), the vehicle routing (Gambardella, Taillard, & Agazzi, 1999), and the multiple knapsack problem (Fidanova, 2003).

The multiple knapsack problem (MKP) is a hard combinatorial optimization problem with wide applications which enlists many practical problems from different domains like financial and other management. It is an interesting problem of both practical and theoretical point of view: practical because of its wide application; theoretical because it is a constraint problem and gives various possibilities for heuristic constructions.

The aim of this chapter is to introduce ACO and its application on MKP.

ANT COLONY OPTIMIZATION ALGORTHM

All ACO algorithms can be applied to any COP. They follow specific algorithmic scheme. After the initialization of the pheromone trails and control parameters, a main loop is repeated until the stopping criteria are met. The stopping criteria can be a certain number of iterations, a given CPU time limit, or a time limit without improving the result or if some lower (upper) bound of the result is known and the achieved result is close enough to this bound. In the main loop, the ants construct feasible solutions, and then the pheromone trails are updated. More precisely, partial problem solutions are seen as

states: each ant starts from random state and moves from a state i to another state j of the partial solution. At each step, ant k computes a set of feasible expansions to its current state and moves to one of these expansions, according to a probability distribution specified as follows. For ant k, the probability p_{ij}^k to move from a state i to a state j depends on the combination of two values:

$$p_{ij}^k = \begin{cases} \dfrac{\tau_{ij}\eta_{ij}}{\sum_{l \in allowed_k} \tau_{il}\eta_{il}} & if\ j \in allowed_k \\ 0 & otherwise \end{cases} \quad (1)$$

where:

- η_{ij} is the attractiveness of the move as computed by some heuristic information, indicating a priori desirability of that move;
- τ_{ij} is the pheromone trail level of the move, indicating how profitable it has been in the past to make that particular move (it represents therefore a posterior indication of the desirability of that move); and
- $allowed_k$ is the set of remaining feasible states.

Thus, the higher the value of the pheromone and the heuristic information are, the more profitable it is to include state j in the partial solution. In the beginning, the initial pheromone level is set to τ_0, which is a small positive constant. In nature there is not any pheromone on the ground at the beginning, or the initial pheromone is $\tau_0 = 0$. If in ACO algorithm the initial pheromone is $\tau_0 = 0$, then the probability to chose next state will be $p_{ij}^k = 0$ and the search process will stop from the beginning. Thus it is important that the initial pheromone value is positive.

The pheromone level of the elements of the solutions is changed by applying the following updating rule:

$$\tau_{ij} \leftarrow \rho.\tau_{ij} + \Delta\tau_{ij} \quad (2)$$

where the rule $0 < \rho < 1$ models evaporation and $\Delta\tau_{ij}$ is an added pheromone. The ACO algorithms differ in pheromone updating. There exist various versions of ACO algorithms such as the ant system (Dorigo et al., 1996), the ant colony system (Dorigo, 1999a), ACO with elitist ants (Dorigo, 1999b), the max-min ant system (Stützle & Hoos, 2000), the ant algorithm with additional reinforcement (Fidanova, 2002, pp. 292-293), the best-worst ant system (Cordón et al., 2002), and so on. The main difference between them is pheromone updating.

Ant System

The first ant algorithm is ant system. In this algorithm all pheromone is decreased, and after that every ant adds a pheromone corresponding to the quality of the solution. More precisely:

$$\Delta\tau_{ij}^k = \begin{cases} (1-\rho) f(S^k) & \text{if it is maximization problem} \\ (1-\rho)/f(S^k) & \text{if it is minimization problem} \end{cases}$$

where:

- $\Delta\tau_{ij}^k$ is the pheromone added by the ant k;
- S^k is the solution achieved by ant k;
- $f(S^k)$ is the value of the objective function.

In any ant system, better solutions and elements used by more ants receive more pheromone.

Ant Colony System (ACS)

The main features of the ACS algorithm are as follows: to use the best found solution, after each iteration the ants—which construct the best solution from the beginning of the trials—add pheromone; to avoid stagnation of the search, the pheromone on other solutions is decreased. Local pheromone updating and global pheromone updating in ACS are applied. In the local pheromone updating the value of the pheromone on used elements decreases and comes between initial pheromone τ_0 and the old value of the pheromone. It is a kind of diversification of the search in the search space.

$$\tau_{ij} \leftarrow \rho.\tau_{ij} + (1-\rho).\tau_0$$

In the global pheromone updating, the ant that constructs the best solution adds another pheromone depending on the quality of the solution.

$$\tau_{ij} \leftarrow \alpha.\tau_{ij} + (1-\alpha).\Delta\tau_{ij}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} f(S^k) & \text{if it is maximization problem} \\ 1/f(S^k) & \text{if it is minimization problem} \end{cases}$$

The main idea of ACS is to enforce the pheromone of the best found solution and at the same time to diversify the search.

Ant Algorithm with Elitist Ants

In this ant algorithm only one or a fixed number (n) of ants add pheromone. The pheromone corresponding to other elements is only evaporated. Thus the pheromone of the best n solutions is forced. It is a kind of intensification of the search around the best found solutions.

Max-Min Ant System (MMAS)

The main features of MMAS algorithm are as follows:

- To exploit the best found solution—after each iteration only one ant adds a pheromone.
- To avoid stagnation of the search, the range of possible pheromone value is limited to a fixed interval $[\tau_{\min}, \tau_{\max}]$.

In MMAS algorithm the pheromone value is initialized so that after the first iteration all pheromone values are equal to $\tau_{\max}$. In the next iterations only the elements belonging to the best solution receive a pheromone; other pheromone values are only evaporated. The main idea of MMAS is to use fixed lower and upper bounds of the pheromone values. If some pheromone value is less/greater than lower/upper bound, it becomes equal to this fixed lower/upper bound and thus early stagnation of the algorithm is avoided.

Best-Worst Ant System (BWAS)

The main idea of BWAS is to use a pheromone mutation. The pheromone value of the best solution is increased, while the pheromone value of the worst solution is decreased. Thus the probability to choose elements of worst solution in the next iteration becomes lower.

Ant Algorithm with Additional Reinforcement (ACO-AR)

The main idea of ACO-AR is after pheromone updating to add additional pheromone to unused elements. Thus some elements receive additional probability to be chosen and become more desirable. Using ACO-AR algorithm the unused elements have the following features:

- They have a greater amount of pheromone than the elements belonging to poor solutions.
- They have a less amount of pheromone than the elements belonging to the best solution.

Thus the ants are forced to exploit a search space which has not been exploited yet without repeating the bad experience.

ANT ALGORITHM AND CONVERGENCE

The ACO is a meta-heuristic algorithm for approximate solution of combinatorial optimization problems. The construction of a good solution is a result of the agents' cooperative interaction. Failure in local optimum may occur when we perform the ACO algorithm. This can happen when the pheromone trail is significantly higher for one choice than for all others. This means that one of the choices has a much higher amount of pheromone than the others, and an ant will prefer this solution component over all alternatives. In this situation, ants construct the same solution over and over again, and the exploration of the search space stops. It should be avoided by influencing the probabilities for choosing the next solution component which depends directly on the pheromone trails. Various techniques exist to avoid failing into local optimum as re-initialization, smoothing of the pheromone, additional reinforcement, diversification, and intensification of the search.

Re-Initialization

When the ants repeat the same solution over and over again, the pheromone is re-initialized (Stützle & Hoos, 2000) and the algorithm starts from the beginning. The aim is to start to create solutions from other starting states and probably to construct differently from previous solutions. This technique can prevent some failing into local optimums, but the algorithm is not guaranteed to converge to an optimal solution. This technique can be applied to any ant algorithm.

Smoothing of the Pheromone Trails

The main idea of the smoothing (Stützle & Hoos, 2000) is to increase the pheromone trails according to their differences to the maximal pheromone trail as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \delta.(\tau_{\max} - \tau_{ij}),$$

where $0 < \delta < 1$ is a smoothing parameter. The above proposed mechanism has the advantage that the information gathered during the run of the algorithm is not completely lost, but merely weakened. For $\delta = 1$ this mechanism corresponds to a re-initialization of the pheromone trails, while for $\delta = 0$ pheromone trail smoothing is switched off. After the smoothing, the current lower bound of the pheromone increases.

Fixed Bounds of the Pheromone

Other method to prevent early stagnation is to fix the lower and the upper bound of the pheromone (Stützle & Hoos, 2000). Thus if the pheromone becomes less/greater than the lower/upper bound, it becomes equal to this lower/upper bound. Thus there are not choices of very high or very low amounts of pheromone.

Additional Reinforcement

The aim of additional reinforcement is to add additional pheromone on choices of pheromone low amount and thus they become more desir-

able (Fidanova, 2002, pp.292-293). It is a way to force the ants to look for new solutions.

Search Diversification and Intensification

In some ant algorithms, diversification and intensification techniques—like increasing the amount of the pheromone for some choices and decreasing it for others—are used. The aim is to intensify the solution search on one side and to diversify it on the other.

It is important to know whether the algorithm converges to the global optimum. Stützle and Dorigo (2002) proved that if the amount of the pheromone has a finite upper bound and a positive lower bound, then the ACO algorithm converges to the optimal solution. This means that if the probability to choose any element does not converge to zero, then the ACO algorithm converges to the global optimum. Stützle and Dorigo (2002) proved that the Ant Colony System and Max-Min Ant System satisfy the conditions for convergence and thus they converge to the global optimum when the time (number of iterations) converge to infinity.

Additional reinforcement can be applied to any ACO algorithm. Fidanova (2004) has proved that after additional reinforcement of unused elements of any ACO algorithm, it converges to optimal solution when the algorithm is run for a sufficiently large number of iterations independently whether the original ACO algorithm converges.

MULTIPLE KNAPSACK PROBLEM

The MKP has numerous applications in theory as well as in practice. It also arises as a subproblem in several algorithms for more complex COPs, and these algorithms will benefit from any improvement in the field of MKP.

The MKP can be thought of as a resource allocation problem, where there are m resources (the knapsacks) and n objects, and object j has a profit p_j. Each resource has its own budget c_i (knapsack capacity) and consumption r_{ij} of resource i by object j. We are interested in maximizing the sum of the profits, while working with a limited budget. The MKP can be formulated as follows:

$$\max \sum_{j=1}^{n} p_j . x_j$$

$$subject\ to \sum_{j=1}^{n} r_{ij} . x_j \le c_i \quad i = 1,\ldots,m$$

$$x_j \in \{0,1\} \quad j = 1,\ldots,n$$

x_j is 1 if the object j is chosen and 0 otherwise.

There are m constraints in this problem, so MKP is also called the m-dimensional knapsack problem. Let $I = \{1,\ldots,m\}$ and $J = \{1,\ldots,n\}$, with $c_i \ge 0$ for all $i \in I$. A well-stated MKP assumes that $p_j > 0$ and $r_{ij} \le c_i \le \sum_{j=1}^{n} r_{ij}$ for all and. Note that the matrix and the vector are both non-negative.

We can mention the following major applications: problems in cargo loading, cutting stocks, bin-packing, budget control, and financial management may be formulated as MKP. Sinha and Zoltner (1979) propose the use of the MKP in fault tolerance problem, and Diffe and Hellman (1976) designed a public cryptography scheme whose security realizes the difficulty of solving the MKP. Matrello and Toth (1984) mention that two-processor scheduling problem may be solved as a MKP. Other applications are industrial management, team management, naval, aerospace, and computational complexity theory.

The shortest path problem in a transportation network deals with determining the subset of the connected roads that collectively comprise the shortest driving distance or the smallest driving time or the cheapest fair between two cities. The problem is: what subset of lines gives the faster response time for communication between them? Complexity theory is a part of the theory of computation dealing with the resources required during the computation time to solve a given problem. More theoretical application appears either in case a general problem is transformed to a MKP or MKP appears as a sub-problem in solving the generalized assignment problem. It again is used in solving a vehicle routing problem. In addition, MKP can be seen as a general model for any kind of binary problems with positive coefficients (Kochenberger, McCarl, & Wymann, 1974).

In solving MKP one is not interested in solutions giving a particular order. Therefore a partial solution is represented by , and the most recent elements incorporated to *S,* need not be involved in the process for selecting the next element. Also, solutions for ordering problems have a fixed length, as a permutation of a known number of elements is searched. Solutions of MKP, however, do not have a fixed length. In this chapter the solution will be represented by sequence where is 1 if the object *j* is chosen and 0 otherwise.

ACO ALGORITHM FOR MKP

The MKP is an interesting problem from a practical and theoretical point of view: practical, because it involves a lot of real-life and industrial problems; theoretical, because it gives several possibilities for pheromone and heuristic models. One of the basic elements of the ACO algorithm is the mapping of the problem onto a graph. We decide which elements of the problem should correspond to the nodes and the ones to the arcs. The solutions of the problem are represented by paths through the graph.

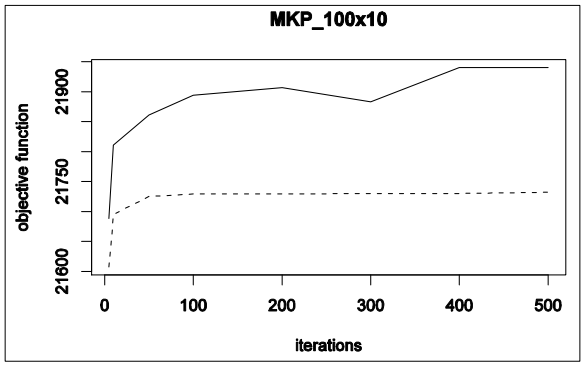
We define the graph of the MKP as follows: the nodes correspond to the objects and the arcs fully connect nodes. Fully connected graph means that after the object *I*, the object *j* might be chosen if there are enough resources and if the object *j* is not chosen yet. At each iteration, every ant constructs a solution. It first randomly chooses the initial object (node in the graph) and then iteratively adds new objects (nodes in the graph) that can be selected without violating resource constraints. Once each ant has constructed its solution, pheromone trails are updated. The pheromone model and heuristic information connected with MKP will be described in detail in the following subsections. Ants start to construct their solution from a random node. Therefore a small number of ants can be used. By experiment, it is found that between 10 and 20 ants are enough to achieve good result, without increasing the number of iterations. Thus the used computer resources such as time and memory are decreased.

Pheromone Model

To solve MKP with ACO algorithm, the key point is to decide which components of the constructed solutions should be rewarded and how to exploit these rewards when constructing new solutions. One can consider two different ways of laying pheromone trails:

- A first possibility is to lay pheromone trails on each selected node of the graph (object). The idea is to increase the desirability of some nodes so that these nodes will be more likely to be selected in constructing a new solution.

Figure 1. Average solution quality: the thick line represents the pheromone on arcs and the dashed line represents the pheromone on nodes



- A second possibility is to lay pheromone trails on the arcs (i,j) of the graph. Here the idea is to increase the desirability to choose node j when the last selected node is i.

The first possibility is closely related to the nature of the problem, as MKP is an unordered problem. To solve MKP with ACO algorithm, Leguizamon and Michalevizc (1999) use the first possibility, while Fidanova (2003) uses the second possibility.

The two pheromone possibilities have been tested on 20 benchmarks of MKP, from OR Library, with 100 objects and 10 constraints (see http://mscmga.ms.ic.ac.uk/jeb/orlib). The number of iterations $K=500$ is fixed for all the runs. For the tests we use ACS algorithm and 20 runs of each of the benchmarks. The initial pheromone parameter is fixed to $\tau_0=0.5$. The evaporation parameters are $\alpha=\rho=0.1$. The number of ants is set to be 10. As shown in Figure 1, there is very early stagnation of the algorithm by pheromone on nodes. This effect can be explained with large pheromone accumulation on some nodes, and thus the ants repeat the same solution over and over again. In the second case the pheromone is dispersed on the arcs. We will illustrate these phenomena with a small example with five objects and one constraint.

Example: $\max(x_1+3x_2+2x_3+x_4+2x_5)$
$2x_1+x_2+3x_3+x_4+3x_5\leq 6$

For heuristic information, let the profit of the objects be used. Thus the objects with greater profit are more desirable. The ACS is applied with one ant. Other parameters are $\tau_0=0.5$, $\alpha=\rho=0.5$. In a first iteration, let the ant start from node 1. Using probabilistic rule the achieved solution in both cases is (x_1,x_2,x_3), and the value of objective function is 6. After updating, the new amount of the pheromone is:

a. pheromone on nodes:
(3.25, 3.25, 3.25, 0.5, 0.5)
b. pheromone on arcs:

Non	3.25	0.5	0.5	0.5
0.5	*Non*	3.25	0.5	0.5
0.5	0.5	*Non*	0.5	0.5
0.5	0.5	0.5	*Non*	0.5
0.5	0.5	0.5	0.5	*Non*

In the second iteration, let the ant start from the node 2. Thus constructed by the ant, the solution in a both cases is (x_2, x_3, x_1). It is the same as in the first iteration, but achieved in different order. The new pheromone is:

a. pheromone on nodes:
(3.937, 3.937, 3.937, 0.5, 0.5)
b. pheromone on arcs:

Non	3.25	0.5	0.5	0.5
0.5	*Non*	3.937	0.5	0.5
3.25	0.5	*Non*	0.5	0.5
0.5	0.5	0.5	*Non*	0.5
0.5	0.5	0.5	0.5	*Non*

In the third iteration, let the ant start from the node 3. The achieved solution by both cases is (x_3, x_2, x_1). The new pheromone is:

a. pheromone on nodes:
(4.109, 4.109, 4.109, 0.5, 0.5)
b. pheromone on arcs:

Non	3.25	0.5	0.5	0.5
3.25	*Non*	3.937	0.5	0.5
3.25	3.25	*Non*	0.5	0.5
0.5	0.5	0.5	*Non*	0.5
0.5	0.5	0.5	0.5	*Non*

Heuristic Information

The second component in the transition probability is the heuristic information. The MKP is a constraint problem, and the constraints can be used for constructing heuristic information in various manners. There are two main types of heuristics: static and dynamic. Static heuristics remain unchanged during the run of the algorithm, while the dynamic heuristics correspond to the current state of the problem. The profit of the objects will be included in the heuristics because it is the most important information for objective function. A better result is expected when we include in the heuristics more information for the problem.

Static Heuristics

Two types of static heuristics are proposed, called "heuristics A" and "heuristics B" respectively.

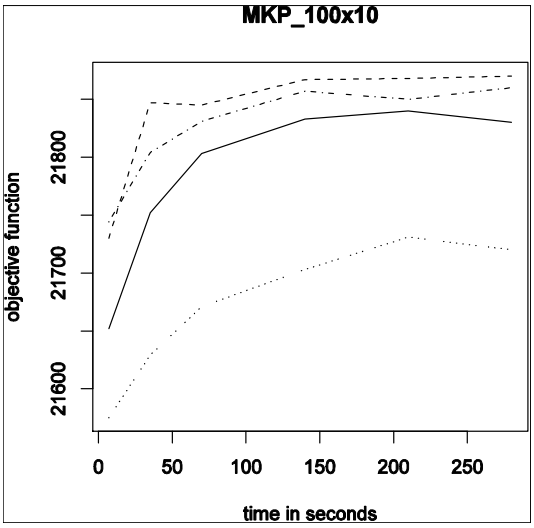
- **Heuristics A:** Let $s_j = \sum_{i=1}^{m} r_{ij}$. For heuristic information we use: $\eta_{ij} = p_j^{d_1} / s_j^{d_2}$, $0 < d_1$ and $0 < d_1$ are parameters. The expenses of the objects are included in heuristic information. Therefore, the objects with greater profit and less average expenses are more desirable. Thus we try to have some balance between expenses and the profit for a given object.
- **Heuristics B:** Let $s_j = \sum_{i=1}^{m} r_{ij} / c_i$. For heuristic information we use: $\eta_{ij} = p_j^{d_1} / s_j^{d_2}$, $0 < d_1$ and $0 < d_1$ are parameters. Thus the heuristic information depends on the profit, the expenses, and the budgets. The objects with greater profit, which use a lesser part of the budget, are more desirable.

Dynamic Heuristics

The third and the forth types of heuristic information are dynamic, and they correspond to the current state of the algorithm. We call them "heuristics C" and "heuristics D" respectively.

- **Heuristics C (Leguizamon & Michalevizc, 1999):** Let $b_i = c_i - \sum_{j=1}^{n} r_{ij} x_j$ be the remainder of the budget before choosing the next object and $s_j = \sum_{i=1}^{m} r_{ij} / b_i$ if $b_i \neq 0$ and $s_j = \sum_{i=1}^{m} r_{ij}$ if $b_i = 0$. For heuristic information we use: $\eta_{ij} = p_j^{d_1} / s_j^{d_2}$, where $d_1 = d_2$. The aim is for the heuristic information to have maximal correspondence to the current state of the algorithm and thus to achieve good result. Leguizamon and Michalevizc (1999) do not verify if $b_i \neq 0$, but because it can happen and there is division by b_i, we add this verification in the algorithm. Thus the objects with greater profit, which use less part of the available budget, will be more desirable.

Figure 2. The graphics show the average solution quality (value of the total cost of the objects in the knapsack) over 20 runs; the dash-dot line represents heuristics A, the dash line represents heuristics B, the dotted line represents heuristics C, and the thick line represents heuristics D



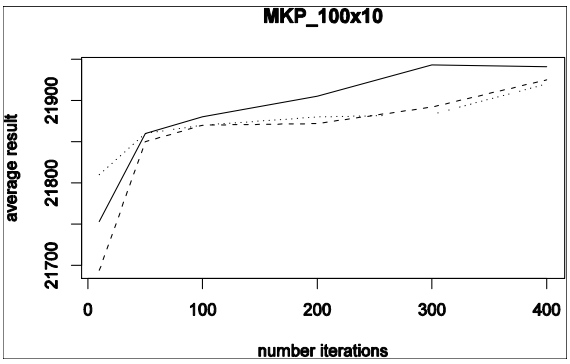
- **Heuristics D:** This is similar to heuristics C, but the parameters d_1 and d_2 can be different. By this heuristics, we can observe the influence of the parameters d_1 and d_2.

The ACO algorithms with four kind of heuristic information have been tested on 20 benchmarks of MKP, from OR Library, with 100 objects and 10 constraints. For the tests we use ACS algorithm with pheromone on arcs. The initial pheromone parameter is fixed to τ_0=0.5. The evaporation parameters are set to α=ρ=0.1. When $d_1 \neq d_2$, d_1=3 and d_2=2, otherwise d_1=d_2=1. The number of ants is set to 10. First we observe that the heuristics B shows advantage over the other tree heuristics. This means that it is more important that the expenses be a small part of the relevant budget. We expected to achieve better results by dynamic heuristics because they correspond to the current state of the problem. In spite of our expectations, we achieved weaker results by dynamic heuristics. Using dynamic heuristics, the chosen object order became important; the desirability of an object was not the same if it was chosen in the beginning of the iteration or later. The MKP is an unordered problem, and for it the order in which the objects are chosen is not important. Thus we can explain better results by static heuristics. Comparing heuristics C and D, we observe the importance of the parameters d_1 and d_2. In the case $d_1 \neq d_2$, the achieved results are better. The parameters d_1 and d_2 show the importance of the profit and the constraints in heuristic information. If d_1 is greater than d_2, then the profit is more important, and in the opposite case the constraints are more important. If both values d_1 and d_2 are great, then the heuristic information is more important than the pheromone in the transition probability.

Comparison between ACO Algorithms

The ACO algorithms are differing in pheromone updating. We compare some ACO algorithms applied on MKP. The ant colony system, the max-min ant system, and the ant algorithm with additional reinforcement have been chosen, because for them it is proven to converge to the global optimum. These ACO algorithms have been tested on 20 benchmarks of MKP with 100 objects and 10 constraints from OR Library. The reported results are average on 20 runs of each benchmark. The pheromone is laid on the arcs and the heuristics B is used. ACO-AR is applied on ant algorithm with elitist ant. The added additional pheromone is equal to the maximal added pheromone. The initial pheromone parameter is fixed to τ_0=0.5. The evapora-

Figure 3. Average solution quality: the thick line represents ACO-AR, the dotted line represents MMAS, and the dashed line represents ACS



tion parameters are $\alpha=\rho=0.1$. The minimal pheromone is set to $\tau_{min}=1000$, and the value of the maximal pheromone is equal to the approximate upper bound of the pheromone (Stützle & Hoos, 2000). The number of ants is set to 10. As shown in Figure 3, ACO-AR outperforms ACS and MMAS. By ACS and MMAS we achieve very similar results. In some of the runs, ACO-AR reaches the best found results in a literature by meta-heuristics methods.

CONCLUSION

In this chapter the ACO algorithms and their implementations on MKP are described. The MKP is represented by graph and the solutions are represented by paths through the graph. We compare two pheromone models, pheromone on the arcs of the graph of the problem and pheromone on the nodes of the graph. We observe that laying the pheromone on the arcs, the algorithm achieves better results. When the pheromone is laid on the nodes on some of them, the pheromone concentration becomes very high and ants choose them with higher probability. We compare four representations of heuristic information. Best results are achieved when the heuristic information depends on the profit, the expenses, and the budgets. The objects with greater profit, which use fewer parts of the budgets, are more desirable. We achieve better results by static heuristics than by dynamics. Using dynamic heuristics the probability to choose the same object at the beginning of the iteration is different than choosing it later, and for MKP the chosen objects order is not important. At the end we compare the results achieved by three of the ACO algorithms, proved to converge to the global optimum, ACS, ACO-AR, and MMAS. We achieve best results by ACO-AR, and in some of the runs the achieved results are equal to the best found in the literature. In the future we will investigate hybridization of the ACO algorithms, combining them with other meta-heuristic techniques and appropriate local search procedures.

ACKNOWLEDGMENT

This work is supported by the Marie Curie program of the European Community by grant MERG-CT-2004-510714.

REFERENCES

Cordón, O., Fernàndez de Viana, I., & Herrera, F. (2002) Analysis of the best-worst ant system ant its variations on the QAP. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *From ant colonies to artificial ants* (pp. 228-234) (LNCS 2463). Berlin: Springer-Verlag.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: Optimization by a colony of cooperative agents. *IEEE Transactions on*

Systems, Man and Cybernetics—Part B, 26(1), 29-41.

Dorigo, M., & Gambardella, L.M. (1999a). Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computing, 1,* 53-66.

Dorigo, M., Di Caro, G., & Gambardella, M. (1999b). Ant algorithms for distributed discrete optimization. *Journal of Artificial Life, 5,* 137-172.

Diffe, W., & Hellman, M.E. (1976). New direction in criptography. *IEEE Transactions in Information Theory, 36,* 644-654.

Fidanova, S. (2002). ACO algorithm with additional reinforcement. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *From ant colonies to artificial ants* (pp. 292-293) (LNCS 2463). Berlin: Springer-Verlag.

Fidanova, S. (2002). Evolutionary algorithm for multiple knapsack problem. In D. Corn (Eds.), *Proceedings of the PPSN VII Workshops,* Granada, Spain (pp. 16-20).

Fidanova, S. (2003). ACO algorithm for MKP using various heuristic information. In I, Dimov, I. Lirkov, S. Margenov, & Z. Zlatev (Eds.), *Numerical methods and applications* (pp. 434-330) (LNCS 2542). Berlin: Springer-Verlag.

Fidanova, S. (2004). Convergence proof for a Monte Carlo method for combinatorial optimization problems. In M. Bubak, G.D. Albada, P.M.A. Sloot, & J. Dongarra (Eds.), *Computational science* (pp. 527-534) (LNCS 3039). Berlin: Springer-Verlag.

Gambardella, L. M., Taillard, E. D., & Agazzi, G. (1999). A multiple ant colony system for vehicle routing problem with time windows. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimization* (pp. 63-76). New York: McGraw Hill.

Gambardella, L. M., Taillard, E. D., & Dorigo, M. (1999). Ant colonies for the QAP. *Journal of the Operational Research Society, 50,* 167-176.

Kochenberger, G., McCarl, G., & Wymann, F. (1974). A heuristics for general integer programming. *Journal of Decision Science, 5,* 34-44.

Leguizamon, G., & Michalewicz, Z. (1999). A new version of ant system for subset problems. *Proceedings of the International Conference on Evolutionary Computations,* Washington (pp. 1459-1464).

Lonnstedt, L. (1973). The use of operational research in twelve companies quoted on the Stockholm Stock Exchange. *Journal of Operational Research, 24,* 535-545.

Matrello, S., & Toth, P. A. (1984). A mixture of dynamic programming and branch-and-bound for the subset-sum problem. *Journal of Management Science, 30,* 756-771.

Moyson, F., & Manderick, B. (1988). The collective behavior of ants: An example of self-organization in massive parallelization. *Proceedings of the AAAI Spring Symposium on Parallel Models of Intelligence,* Stanford, CA.

Sinha, A., & Zoltner, A. A. (1979). The multiple-choice knapsack problem. *Journal of Operational Research, 27,* 503-515.

Stützle, T., & Hoos, H. H. (2000). Max-min ant system. In M. Dorigo, T. Stützle, & G. Di Caro (Eds.), *Future generation computer systems* (Vol. 16, pp. 889-914).

Stützle, T., & Dorigo, M. (2002). A short convergence proof for a class of ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation, 6*(4), 358-365.

Chapter XXXIV
A New Way to Reorganize a Productive Department in Cells

Alessandro Brun
Politecnico di Milano, Italy

Marta Zorzini
Politecnico di Milano, Italy

ABSTRACT

The authors propose an algorithm for the reorganization of a production department in cells, starting from a situation of job shop, chasing the main goal of group technology (GT)—that is, to gather pieces with similar technological cycles and to associate every group of items (family) to a group of machines (cell) able to realize all the necessary activities. To get this result, a behavioral pattern has been developed, having its origin in the ants' way of sorting food, larva, and pupa in an anthill. As first results have shown, such an approach turns out to be interesting, provided that the algorithm parameters are adequately set.

GROUP TECHNOLOGY: AN OVERVIEW

Efficiency under every point of view is necessary for a company wanting to survive and thrive in today's competitive scenario. Among the various approaches that have been proposed to improve the efficiency in manufacturing, there is group technology (GT) (Callagher & Knight, 1973).

GT, which was born originally in Russia, can be defined as a manufacturing philosophy helping to manage diversity by identifying similarities in products and activities. It began exploiting these similarities on a single machine, allowing productivity to rise by 30-40%.

This concept was then widened to encompass more machines at the same time, and the new goal was to gather pieces with similar technological cycles and to associate every group of items (family) to a group of machines (cell) able to realize all the necessary activities (Optiz, 1970; Waghodekar & Sahu, 1984; Fazakerlay, 1974; Ham & Yoshida, 1985; Gallagher & Knight, 1986).

The cellular manufacturing system is a valid alternative to an organization based on process specialization, where in every department there are similar machines, able to realize only a part of the technological cycle. Long and uncertain throughput times are usually the major problems in the latter system. Other problems include an increase in inventory holding cost, untimely product delivery, and loss of sales (Cheng, Goh, & Lee, 2001).

It is widely agreed that GT is a management strategy which affects all areas of a company, like part design, production control, process planning, maintenance, and purchasing, and has a relevant impact on its productivity. The benefits of GT reorganization of manufacturing system, discussed by many authors (Burbidge, 1996; Sarker, 1996; Singh & Rajamani 1996), are the following:

- reduce material handling costs,
- reduce setup time,
- reduce throughput time
- improve delivery performances,
- reduce work-in-progress, and
- improve the quality level.

Nevertheless, Burbidge (1996) claims such advantages are not automatic. They are available, but action must be taken to achieve them.

When the variety and the need of its reduction does not concern only the products, but also the technological cycles, the cell formation problem, needing to be dealt with in order to achieve the above said objectives and concerning the identification of the family of parts and the group of machines on which these parts are to be processed, is very complex. These problems have been highlighted by Kusiak (1987), Nagi, Harhalakis, and Proth (1990), and Kusiak and Cho (1992).

If every end-item to produce had the same technological cycle, the management of the productive system would be rather simple: there would not be out-of-cell flows and the load could be well balanced, thus achieving superior performances. Unfortunately the real world is not so simple, and the problem of reorganizing in cells a productive department of average dimensions is notably complex (Selim, Askin, & Vakharia, 1998). For example there are more than 10^{15} ways to reorganize 15 machines and 45 end items in three cells of 5 machines and 15 end-items each, and the number is bound to explode if the number of machines and items in each cell is free to change.

A high number of GT approaches have been developed to decompose a large manufacturing system into smaller, manageable systems based on similarities in design attributes and manufacturing features.

Classifications of methods of cell formation have been proposed by several researchers. A possible classification is based on the type of solution methodology (Singh & Rajamani, 1996; Selim et al., 1998):

- classification approach using coding systems (hierarchical clustering algorithm, P-median model, multi-objective clustering);
- part-machine group analysis methods (rank order clustering—ROC, ROC2, modified ROC, direct clustering algorithm, cluster identification algorithm);
- algorithms based on similarity coefficients (single linkage clustering, complete linkage clustering, and linear cell clustering);
- mathematical models (quadratic programming, graph theoretic, nonlinear programming models); and
- techniques dealing with combinatorial optimization problems (genetic algorithms, artificial neural networks, adaptive fuzzy systems).

Other classifications can be based on the following dimensions (Singh & Rajamani, 1996):

- heuristic vs. optimizing approaches;
- sequential vs. simultaneous approaches to cell formation;
- single-objective vs. multi-objective approaches; and
- iterative vs. random search algorithms.

Advice on the comparison and determination of the appropriateness of these techniques are given by Shafer and Meredith (1990) and Selim et al. (1998).

According to Selim et al. (1998), major shortcomings of prior research are as follows. First, several cell formation procedures consider only a single objective in identifying cells. Second, heuristic approaches use a subjective evaluation and are affected by the nature of input data. Third, mathematical models are often hard to implement due to computational limitations for large real problems. Fourth, very little attention has been paid to the incorporation of manufacturing flexibility in cellular manufacturing systems.

This chapter is aimed at proposing a nature-inspired algorithm that could improve the underlined criticalities, with particular emphasis on the first one. In the next section the problem to be solved and the industrial case are presented and the proposed ant algorithm is described. The third section focuses on the autoset procedures for the parameters used in the algorithm, while in the fourth section the main results of the experimental analysis are highlighted. Finally, conclusions are drawn and some future research directions are proposed.

THE PROPOSED ANT ALGORITHM

GT techniques can be applied to both problems of production systems design, including purchase decisions for the machines, and configuration problems, that is choices of placement for a certain set of machines and production flow management.

The aim of this chapter is to propose a heuristic approach to the latter typology of problems, with reference to a specific industrial case. Criticalities highlighted during the case study could be solved by reorganizing a portion of a production system into cells.

The proposed approach is based on the properties of the ant system paradigm, which was been extensively studied by researchers from the mid-1990s and whose effectiveness has already been recognized in literature by Van Dyke Parunak (1996a, 1996b).

Ant algorithms have successfully been applied to solve many complex combinatorial problems, such as the traveling salesman problem and the problem of network-traffic optimization (Shtovba, 2004). Relevant applications have been proposed to solve:

- scheduling problems, such as the problem of job scheduling for the final line of assembly mixed model (Dorigo, Maniezzo, & Colorni, 1996), the problem of flowshop scheduling (T'kindt, Monmarché, Tercinet, & Laugt, 2002), the problem of group shop scheduling (including the open shop scheduling problem and the job shop scheduling problem) (Blum & Sampels, 2004), and the production-inventory scheduling problem in a steel continuous-casting plant (Ferretti, Zanoni, & Zavanella, 2004); and
- problems of assembly sequence planning for mechanical products (Wang, Liu, & Zhong, 2005).

The heuristic ant algorithm proposed in this chapter deals with the cell formation problem in a sequential way: first, it determines which machines must be grouped together to form cells by a certain number of iterations; on the basis of the system configuration thus obtained, it is then possible to determine which parts

should be grouped in cells by another series of iterations. Two constraints are considered: the set of available machines is fixed, and the number of cells to form by the algorithm is predefined.

After generating a certain number of alternative configurations for the system, it is possible to compare them with the initial case by an objective function (consisting of the sum of differential costs) to select the best solution.

The Industrial Scenario

The proposed algorithm has been tested in a real case and employed for a washing-machines producer whose production department was formerly organized as a job shop.

The productive cycle can be summarized as follows: in a "cutting department" with nine machines, 700 different items (metal sheets of different sizes) were realized; after that, in a "machining department" with 40 machines, 1,500 different parts were produced and then assembled in a "assembly department." The final output was 6,400 different products, which could be divided in three product families (small, medium, and large). The first two departments were organized as a job shop, while the last one encompassed three assembly lines. Because of the wide range of products and the internal layout, lead time was long and quite uncertain, and associated costs were very high. So, the aim was to reorganize the first two departments in cells using group technology. The proposed algorithm has been applied separately to the two departments, and the number of final cells has been set equal to three (that was the number of products families) for each one.

A Quick Overview of the Proposed Algorithm

For the proposed approach, a behavioral pattern has been developed that has its origin in the ants' way of sorting food, larva, and pupa in an anthill. The parallel between a department and an anthill is summarized in Table 1.

Table 1. Comparison between real life and artificial life

Real Life	Artificial Life
Ant hill	Productive department
Rooms of ant hill	Cells and product families
Object moved by ant (larva, food)	Machine, item

In order to solve the two sub-problems constituting the cell formation problem, two different groups of ants are introduced in the algorithm: the "technologist" ants (tech-ants) have the goal to gather machines in the cells, the "productive" ones (prod-ants) to gather pieces into families. In the algorithm the concept of family is assumed to correspond to that of the associated cell.

Each typology of ant has a different objective to reach:

- The aim of tech-ants is to optimize the match between the capability of a cell and the technological requirement determined by the product family associated to the cell.
- The aim of prod-ants is to optimize the match between the capacity of a cell and the load generated by the family associated to it.

Both specified types of agents move within the department and influence each other. Precisely, it is necessary for tech-ants to alternate with the prod-ants with frequency previously fixed by the user, according to the following general rules:

- An ant wanders within a department as if it was in an anthill: the cells (for the tech-ant) and the families of items (for the prod-ant) are like different rooms of the anthill.
- Every ant can lift, transport, and drop objects as it walks through the anthill.
- When an ant enters the system, it does not transport any object.
- An ant memorizes the last objects encountered on its way. When the ant is not loaded, it may lift up an object and begin to carry it with a probability depending on the similarity degree between such object and the array of last visited objects. On the contrary, when the ant is charged, it can release an object on the basis of the same similarity degree.

The initial system configuration is obtained by applying the ROC algorithm (Singh & Rajamani, 1996), and by considering the constraints of the fixed number of final cells within the system and the presence of at least one machine in every cell.

Starting from this configuration, it is possible to gather the available machines in cells, by a certain number (N) of iterations of tech-ants, which will be described in the following paragraphs. For the configuration thus obtained, the allocation of the different items in cells is determined by M iterations of prod-ants. During this phase some elements distinguishing the proposed approach are considered, such as the annual demand and number of lots for the different products, the load balance among cells, and the target level of saturation for the machines.

Considering the current state of the research, it is possible to apply the proposed approach to reorganize already existing productive departments, but not to design new departments.

Inputs of the algorithm are as follows:

- number and typology of available machines;
- technological capability and target saturation level for the machines;
- number and type of different products; and
- annual demand for the different products.

Starting from these data, the algorithm is able to generate a certain number of robust solutions, but in general it does not necessarily find the best configuration. Since the set of machines within the system is fixed, the objective function adopted to compare the obtained solutions does not include costs for purchasing machines. Costs for standard processing are not considered either, because they are not differential among the different solutions. The relevant elements compared to the initial case are as follows:

- costs for moving machines among cells;
- costs for inter-cell moving of jobs on annual basis;
- costs for facilities and tools on annual basis; and
- costs of overtime processing due to high level of machine saturation.

A synthetic view of the algorithm is presented in Figure 1.

The Tech-Ants

After entering the system, a tech-ant selects a cell from where to start its random walk. As shown by the flow chart in Figure 2, it then analyzes in a random order the machines belonging to the cell. When analyzing a machine, the ant can decide whether to lift up or not the selected machine. The higher the probability to

Figure 1. Overall system structure

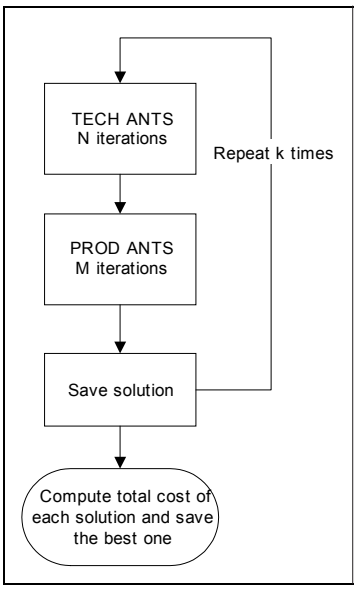
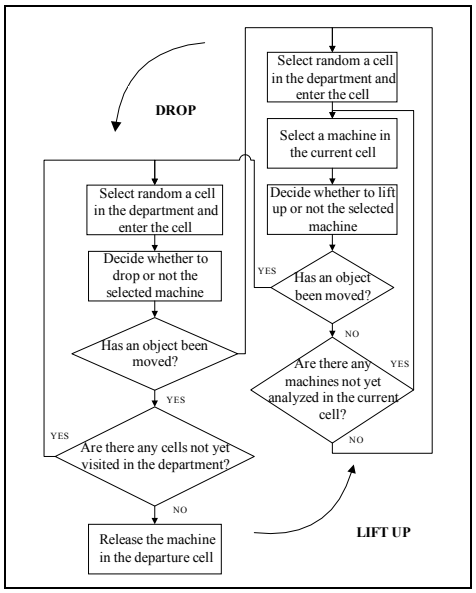


Figure 2. Tech-ants flow chart



lift the selected machine, the lower the degree of similarity of this to the last N machines analyzed in the current cell, where N is an integer number indicating the size of an ant's memory (see "Memory of the Ants" paragraph).

If the ant decides not to lift the machine, it will then select another machine among the remaining ones. If instead it decides to lift the machine (or if it has analyzed all the machines without lifting any), the ant leaves the cell and moves toward the next one. When an ant enters a new cell carrying a machine, it begins to compare the machine with the present ones and it appraises the opportunity of dropping it, considering just as before the degree of similarity between the machines. If the ant does not succeed in dropping the carried machine in any of the cells, the release of the machine is forced in the departure cell. In Figure 2 the distinction between two loops, the loop for lifting up an object and the loop for dropping it, is outlined. Besides, in order to move a machine, a minimum number of ants is necessary.

The input parameters of the algorithm about the tech-ants will be explained in the following paragraphs:

1. similarity coefficient between two machines;
2. probability function of moving a machine;
3. memory of the ants;
4. number of ants in the production system;
5. "weight" of the machines.

Similarity Coefficient between Two Machines

For every given couple of machines, the similarity coefficient measures the number of common items. The higher the number of items that need working on both the machines, the higher the similarity degree between the machines in a range from 0 to 1 (Sarker, 1996). In the past,

several measures of similarity among the machines and parts have been proposed as single linkage clustering (SLC), complete linkage clustering (CLC), average linkage clustering (ALC), and linear cell clustering (LCC) (Singh & Rajamani, 1996).

Traditional coefficients compare one machine with another one, while an ant compares one object with a group of objects saved in its memory. To preserve this behavior and optimize, at the same time, the matching between capability and technological requirement, a "modified single linkage clustering" has been defined. It compares the selected machine with a "virtual machine." In general, a machine can be defined as an array of 1 and 0, where:

- a 1 in a position means that at least a phase belonging to the routing of a specific product (or product line) is carried out by the considered machine; and
- a 0 in a position means that a specific product (or product line) is not worked by the considered machine.

The virtual machine is determined as the average of the machines in the memory of the ant; it can be defined as a vector of numbers between 0 and 1 (not of binary numbers like for the SLC method). This fuzzy logic expresses the contribution of each machine to the definition of "similarity."

The proposed similarity coefficient is as follows:

- the numerator is calculated as the scalar product of the vector associated to the virtual machine and the vector associated to the selected machine; and
- the denominator is calculated as the sum of the terms of the vector whose elements equal the maximum between the homologous elements of the vector associated to the virtual machine and the vector associated to the selected machine.

Table 2. Exemplification of the concept of virtual machine by means of a simple numerical case

Average machine between machine A and machine B			Selected machine C
A		B	C
1	1	1	1
1	0	0,5	1
1	1	1	0
1	1	1	1
1	0	0,5	1
0	1	0,5	0
0	0	0	0
0	0	0	0
0	0	0	1
0	0	0	0

A simple numerical example can be useful to clarify the proposed procedure (see Table 2). The similarity coefficient between the average machine and the selected one is as follows:

$$CS = \frac{1+0,5+1+0,5}{1+0,5+1+1+0,5+0,5+1} = 0,55$$

Probability Function of Moving a Machine (P_{mach})

The probability function defines the inclination of an ant to lift a machine; it is one of the most important features of the model. In fact, this function influences the dynamic of the whole system.

Probability to lift a machine h and to move it to a cell j is based on the coefficient of similarity

(CS_{hj}) in that the higher the CS, the higher the probability. The adopted probability function is as follows (notation explained in Appendix A):

$$P_{mach,hj} = e^{\left[\frac{(CS_{hj}-1)}{\beta * CS_{hj}}\right]}$$

By modifying b, it is possible to adapt the model to different contexts. The trend of the function depending on β value is shown in Figure 3.

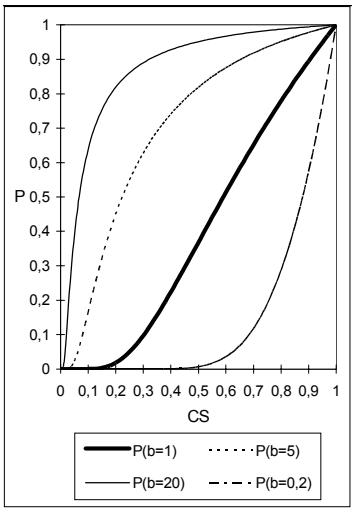
A procedure of auto-set of β parameter has been adopted which will be explained in the following section.

Memory of the Ants (Mem)

The size of an ant's memory is a typical variable of natural models. In particular, to provide food or to build and organize an ant hill, ants have few tools and follow few rules. Their brain works out information received from a space of a few centimeters around their body. Besides, they store only a few seconds of their past in their memory. This peculiarity gives them the capability of sorting their ant hill. If memory was too long, differences would be slight and the system would be static; otherwise, if memory was too short, differences would be unacceptable and the system would be completely chaotic.

In the proposed model the user can set the memory size to a value ranging from 1 to N_{mach} (N_{mach} being the number of machines present in the production department).

Figure 3. Probability function depending on b value



Number of Ants in the Production System (N_{prod}, N_{tech})

One of the rules of artificial life is that an interesting behavior can emerge by combining a multitude of simple elements. That is the reason why a certain number of ants will be introduced in the system (N_{tech} for tech-ants and N_{prod} for prod-ants).

Besides, the behavior of just one ant could not have been simulated because of the introduced weight of the machines. Obviously, the larger the number of ants in a productive department, the higher the fluidity of the system and the higher number of movements makes more likely the achievement of a better solution.

"Weight" of the Machines (SW)

Cost consideration can be introduced into the behavior of tech-ants by considering the weight of the machines: in the algorithm, weights are proportional to costs of inter-cell shifting of machines. In this way, it is possible to balance two different types of costs that are usually conflicting for a fixed set of machines. They are as follows:

- una tantum costs occasionally met to move machines, tools, and facilities; and

- yearly handling costs (due to inter-cell movements of jobs and material handling).

It is possible to associate a "standard weight" to every machine, depending on the number of ants in the department (γ is a percentage value).

$$SW = \frac{N_{tech}}{N_{mach}} * \gamma$$

For each machine, standard weight is increased proportionally to the cost for its inter-cell moving.

The Prod-Ants

The behavior of prod-ants is the following: the single ant enters the system unloaded and, in a random way, it selects a job belonging to a family of the considered department. At this point it considers the content of the work of the job in the associated cell: if the balancing of the loads is better with the product worked in the cell in comparison to the case in which it is transferred to another cell, the ant leaves the job where it was and selects another job; on the contrary, the ant moves the job to another cell. This way, prod-ants look for a set of products that guarantees an optimal saturation of the machinery. On the condition that a "job" replaces a "machine," the flow chart shown in Figure 2 can be considered valid also for prod-ants.

The ants appraise the work content of the machines by the "load vector," which will be described in the following paragraphs. A number of load vectors equal to the number of cells in the department is calculated for each job.

Variables used to determine prod-ants' behavior are the cost of out-of-cell operations, which allows eventual operations realized outside the cell associated to the job to be taken into account, and the target level of machine saturation specified by the user. These variables determine the probability of moving a job which expresses the inclination of an ant to leave the selected job in one cell.

Cost of Out-of-Cell Operations (C)

When the family of machines of a cell is not able to execute all the operations required by a certain item, some activities have to be realized elsewhere, thus implying a handling cost (the duplication of resources is forbidden because the set of machines is fixed).

It is possible to express the cost of out-of-cell operations by the degree of glut of the resources. The cost is depending upon the number of lots (not the number of pieces) worked out-of-cell. In fact, working out-of-cell a job of pieces has more or less the same impact in terms of handling cost and set-up times.

The cost associated to out-of-cell processing of job "i" can be, therefore, defined analytically like the product of two terms:

- a fixed constant C representing the cost of single out-of-cell operation; and
- a variable, declared in percentage form, function of the total number of lots.

The coefficient has, therefore, the following form:

$$\xi_i = \frac{n_i}{\sum_{j \in W} n_j} * C$$

Target Level of Machine Saturation (TL)

Once the cost to be associated to the out-of-cells operations has been defined, the "load

vectors" can be determined with reference to one cell of the department both in the presence and in the absence of the product under examination. Each vector is calculated on the condition that the selected job is worked in one of the cells and its dimension is (n,1), where n is the number of machines in the department. Elements within the vector are the resulting loads for the N machines of the department.

Then the program calculates the deviation (i.e., the distance) of these loads from the target utilization rate specified by the user for every machine. Such target level should ensure a good trade-off between an adequate fluidity of parts within the system and a good load of the machines, subject to the physical and operative constraints.

The inclination of the ants to release the selected product is a function of the weighted sum of these deviations. Precisely, in defining the function it has been guaranteed that:

- the function gives always positive values so that positive and negative values cannot compensate and therefore produce a null result; and
- the trend of the function is different for overload and low saturation conditions, to take into account the more relevant impact of values higher than the target level (impossibility to satisfy the demand level, higher dependence on failures and breakdowns, reworking costs and problems in the production planning) compared to lower rates.

The adopted function, with reference to machine h, is as follows:

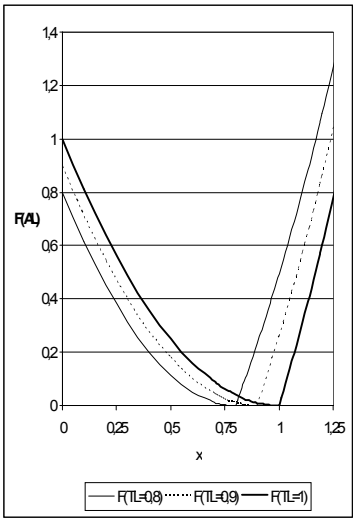
$$\Delta_h(\mathrm{AL}) = \begin{cases} \dfrac{1}{\mathrm{TL_h}}\mathrm{AL_h}^2 - 2\mathrm{AL_h} + \mathrm{TL_h} & \mathrm{AL_h} \le \mathrm{TL_h} \\ e^{\mathrm{AL_h}} - e^{\mathrm{TL_h}} & \mathrm{AL_h} > \mathrm{TL_h} \end{cases}$$

The trend of the curve is parabolic for the range of those load values that are lower than the target level, while it is exponential for the range of higher values. Since the increase rate of the exponential curve is higher than that of the parabolic one, the overload conditions are more penalized than the low saturation ones. As shown in Figure 4, even small variations in the target level can imply relevant changes in the results.

Probability Function of Moving a Job (P_{job})

The inclination of the ants to release the selected product in one cell rather than in another depends on the impact of such decision on the machine loads. Such inclination is consequently expressed by a probability function, which depends on the ratio between the weighted sum of the load vector deviations in two alternative cells. The load vector deviations are weighted differently depending on the machine under examination. The weight to associate to one machine is determined by calculating the refer-

Figure 4. Deviations of loads from the target level



ence job. It has been defined as the weighted average of the annual processing times of the different products processed on each machine. Once the reference job and the availability of each resource have been known, it is possible to calculate the maximum number of standard mix the machinery can produce. The higher the number of units that can be produced is, the lower the weight associated to the resource is.

With reference to two cells i and j, the ratio between the load vector deviations is as follows:

$$x_{ij} = \frac{\sum_{h} \alpha_h * \Delta_{hi}}{\sum_{h} \alpha_h * \Delta_{hj}}$$

The proposed expression for the probability function of moving a job from cell i to cell j is as follows (where x_{ij} is the independent variable and K a multiplicative constant):

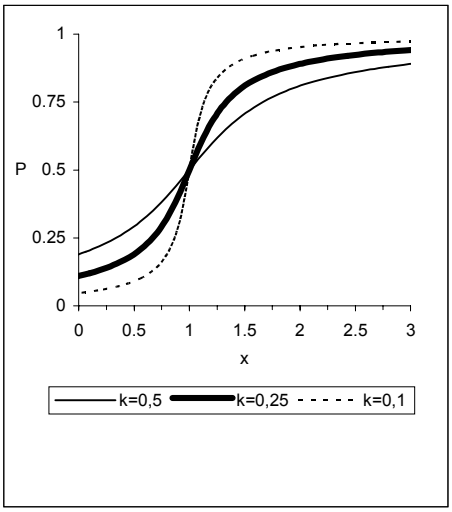
$$\begin{cases} P_{job,ij} = 0 & x_{ij} = 0 \\ x_{ij} = K * \dfrac{0{,}5 - P_{job,ij}}{P_{job,ij} * \left(P_{job,ij} - 1\right)} + 1 & x_{ij} > 0 \end{cases}$$

Probability values are included between 0 and 1, and the function associates a probability equal to 0,5 for x equal to 1. Figure 5 shows the trend of the probability curve for each single variation of K. The parameter K can be auto-set using the same procedure adopted for β parameter, which will be presented in the next section.

AUTO-SET OF THE PARAMETERS

For maximum algorithm effectiveness, parameters are to be carefully set according to the "morphology" of the specific production context considered. For this aim a procedure of auto-setting of parameters has been introduced.

Figure 5. Probability function depending on k value



The procedure is rather simple, and the idea is the same for the two probability functions defined for tech-ants and prod-ants: a partition of the space of solutions is done during every run of simulation, and in the following run only the most "attractive" part is explored.

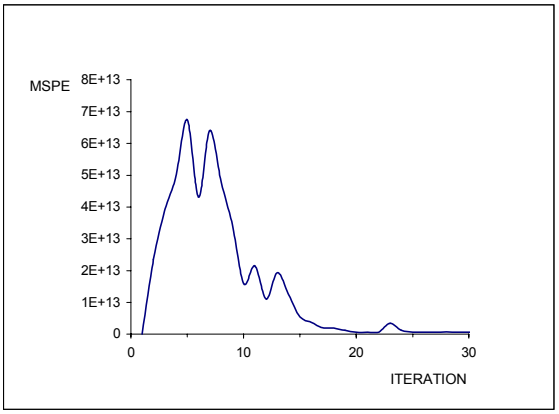
To better understand the logic, it is possible to analyze the procedure for auto-setting β parameter of the probability function of tech-ants.

The user can choose the range in which the parameters can move and the number of iterations to set the parameters' value. The center value of the available range will be used as the initial value of the parameter β. At the end of the considered run, if the number of tech-ants that had a value of the function of probability greater or equal to 0.5 has been greater than the number of tech-ants with a value of the function of probability less than 0.5, that means that b is too high and the system too dynamic; thus, in the following run the available range will be the left one. In other words, in each run a partition

Table 3. Variable input parameters

Variable input parameters
Memory size of tech-ants
Target level of machine saturation
Cost of out-of-cell operations
Ratio tech-ants/prod-ants

Figure 6. The mean square pure error trend



of the space of possible values is set-up and only the best part will be explored on the following run. This process is repeated for a certain number of times (the number of iterations is set by the user).

EXPERIMENTAL ANALYSIS

Factorial Design

Some of the input parameters of the model described in the previous paragraphs (the "tech-ants" and the "prod-ants") have been set before the analysis, whereas the parameters included in Table 3 have been allowed to vary.

Every factor has been studied on three different levels, but for reasons of time and resources, a reduced factorial design encompassing 27 configurations has been tested, with five repetitions for each configuration.

The trend of "mean square pure error," which has been calculated in order to determine the duration of runs of simulation, is shown in Figure 6.

An F test has been carried out to calculate the length of runs. Choosing an amplitude of having an error of I type $\alpha = 0.05$ and an amplitude of the classes $p = 21$, a number N of 74 iterations has been calculated.

Analysis of Results

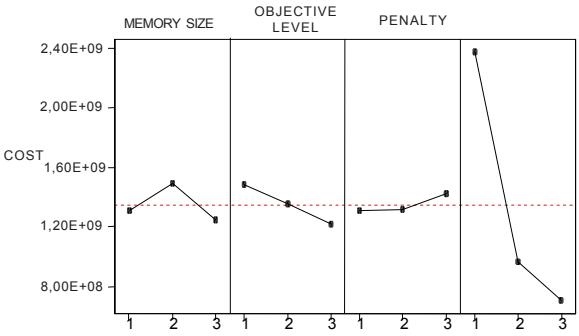
An *analysis of variance (ANOVA)* has been carried out in order to understand the relevance of the tested variables. The ANOVA analysis has pointed out that only three factors are statistically significant:

- memory of tech-ant,
- target level of machine saturation, and
- ratio of tech-ants to prod-ants.

As Figure 7 shows, the system response curve depending on the ratio tech-ants : prod-ants has a greater slope compared to the other factors; this result derives from the disproportion between the number of machines and the number of jobs worked in the production system. That disproportion (some tens of machines vs. some thousands of products) has to be surpassed by a higher level of the ratio.

Another relevant variable has been showed the memory of tech-ants but, unexpectedly,

Figure 7. Main effects plot—data means for cost



best reductions of costs have been found with both high and low levels of the parameter. As a matter of fact, tech-ants move machines with the goal of finding organizational improvements, but organizational improvements do not necessarily correspond to an economic improvement. The used costs structure has penalized the work of tech-ants, so the result can be justified.

The third significant factor has been the target level of machine saturation: configurations of minimum cost have been found with a high level of this variable. Increasing the target level of machine saturation, the necessity of overtime work decreases and, because of the costs structure, that determines a costs reduction.

Not all the two-factor interactions are significant. In particular, the following interactions do not seem to be relevant:

- interaction between memory and target level of machine saturation, and
- interaction between cost and ratio of tech-ants to prod-ants.

On the contrary, the variance analysis has shown the following interactions to be significant:

- interaction between the cost of out-of-cell operations and the level of machine saturation, and
- interaction between the ratio tech-ants: prod-ants and the level of machine saturation.

The cause determining interaction between the cost of out-of-cell operations and the level of machine saturation is quite easy to detect: increasing the level of both parameters, the inclination of agents to minimize the out-of-cell load is maximized.

CONCLUSION AND FUTURE DEVELOPMENTS

In this chapter a heuristic algorithm is presented for solving a multidimensional GT cell formation problem based on cooperation among intelligent agents able to solve problems of realistic size. Applying multi-entity techniques derived from the properties of the ant system paradigm seems to allow some improvements compared to the traditional techniques for grouping machines into cells and parts into families—from both the effectiveness and the efficiency point of view. The main benefits are related to the possibility of considering the trade-off between different features of the production system, such as the target saturation level and the actual load conditions for the machines. The application of this method to a real case allowed us to test the improvements obtained through the algorithm in respect to the initial solution in terms of the sum of associated costs. In addition the proposed solution is able to reduce the variability of the delivery lead times and to guarantee more flexibility in case of increased demand.

However some future research should be desirable to exactly evaluate the obtainable

advantages compared to other existing techniques. (Obviously, a comparison with ROC method, which is used to obtain the initial system configuration for the proposed approach, is incorrect.)

A possible extension of the model is the removal of the constraint of a fixed predetermined set of machines within the production system. Other possible future developments for the research are related to the following areas:

- testing alternative probability functions, and
- completely automatizing the parameters of self-regulation procedure.

In order to achieve the first objective, it is possible to modify similarity coefficients used by tech-ants by improving their memory and their comparison capacities. A further extension of the model could be the introduction of a mechanism allowing the exit from local minimum positions.

As far as the second area is concerned, future research on the applicability of the ant system paradigm should include some mechanisms of natural selection; for instance, rewarding the ants that produce better behavior can help to reinforce the system and set the most adequate number of agents.

REFERENCES

Ballakur, A., & Steudel, H. J. (1987). A within-cell utilization-based heuristic for designing cellular manufacturing systems. *International Journal of Production Research, 25*(5), 639-665.

Blum, C., & Sampels, M. (2004). An ant colony optimization algorithm for shop scheduling problems. *Journal of Mathematical Modeling and Algorithms, 3,* 285-308.

Burbidge, J. L. (1996). The first step in planning group technology. *International Journal of Production Economics, 43,* 261-266.

Cheng, C. H., Goh, C.-H., & Lee, A. (2001). Designing group technology manufacturing systems using heuristics branching rules. *Computers and Industrial Engineering, 40,* 117-131.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: Optimization by a colony of cooperating agents. *IEEE Transaction on Systems, Man and Cybernetics, Part B, 26*(1), 29-41.

Fazakerlay, G. M. (1974). Group technology: Social benefits and social problems. *The Production Engineer,* (October), 383-386.

Ferretti, I., Zanoni, S., & Zavanella, L. (2005). Production-inventory scheduling using ant system meta-heuristic. *International Journal of Production Economics.*

Foulds, L. R., & Neumann, K. (2003). A network flow model of group technology. *Mathematical and Computer Modeling, 38,* 623-635.

Gallagher, C. C., & Knight, W. A. (1973). *Group technology.* London: Butterworth.

Gallagher, C. C., & Knight, W. A. (1986). *Group technology: Production methods in manufacture.* New York: John Wiley & Sons.

Ham, I., & Yoshida, T. (1985). *Group technology: Application to product management.* Boston: Kluwer-Nijhoff.

Kamrani Ali, K., & Parsaei, H. R. (1994). A methodology for the design of manufacturing system using group technology. *Production Planning & Control, 5,* 450-464.

Kusiak, A. (1987). The generalized group technology concept. *International Journal of Production Research, 25*(4), 561-569.

Kusiak, A., & Cho, M. (1992). Similarity coefficient algorithms for solving the group technology problem. *International Journal of Production Research, 30*(11), 2633-2646.

Kusiak, A. (1987). The generalized group technology concept. *International Journal of Production Research, 25*(4), 561-569.

Malakooti, B., & Yang, Z. (2002). Multiple criteria approach and generation of efficient alternatives for machine-part family formation in group technology. *IEEE Transactions, 34,* 837-846.

Nagi, R., Harhalakis, G., & Proth, J.-M. (1990). Multiple routings and capacity considerations in group technology applications. *International Journal of Production Research, 28*(12), 2243-2257.

Nyman, L. R. (1992). *Making manufacturing cells work.* Dearborn, MI: Society of Manufacturing Engineers.

Optiz, H. (1970). *A classification system to describe work-pieces*. Elmsford, NY: Pergamon Press.

Pai, P.-F.F., & Lee, E. S. (2001). Adaptive fuzzy systems in group technology. *Computers and Mathematics with Application, 42*(1), 1393-1400.

Sarker, B. R. (1996). The resemblance coefficients in group technology: A survey and comparative study of relational metrics. *Computers and Industrial Engineering, 30*(1), 103-116.

Selim, H. S., Askin, R. G., & Vakharia, A. J. (1998). Cell formation in group technology: Review, evaluation and directions for future research. *Computers and Industrial Engineering, 34*(1), 3-20.

Shafer, S. M., & Meredith, J. R. (1990). A comparison of selected manufacturing cell formation techniques. *International Journal of Production Research, 28,* 661-673.

Shtovba, S. D. (2004) Ant algorithms: Theory and applications. *Programming and Computer Software, 31*(4), 167-178.

Singh, N., & Rajamani, D. (1996). *Cellular manufacturing systems—design, planning and control.* London: Chapman & Hall.

T'kindt, V., Monmarché, N., Tercinet, F., & Laugt, D. (2002). An ant colony optimization algorithm to solve 2-machine bi-criteria flowshop scheduling problem. *European Journal of Operational Research, 142,* 250-257.

Van Dyke Parunak, H. (1996a). *Industrial applications of multi- agent systems engineering principles from natural multi-agent systems.* Ann Arbor, MI: Industrial Technology Institute.

Van Dyke Parunak, H. (1996b). *Go to the ant engineering principles from natural multi-agent systems.* Ann Arbor, MI: Industrial Technology Institute.

Waghodekar, P. H., & Sahu, S. (1984). Machine-component cell formation in group technology: MACE. *International Journal of Production Research, 22*(6), 937-948.

Wang, J. F., Liu, J. H., & Zhong, Y. F. (2005). A novel ant colony algorithm for assembly sequence planning. *International Journal Advanced Manufacturing Technology, 25,* 1137-1143.

KEY TERMS

Ant Algorithm: Solution algorithm based on the simulation of self-organized colony of biologic ants according to the principles of the

ant system paradigm, aimed at solving different types of problems within a reasonable amount of time.

Cell Formation: Activity consisting of the definition of families of parts to be manufactured and groups of machines of the system by which these parts must be processed.

Cellular Manufacturing System: Productive system composed of cells, including resources able to process specific groups of items.

Group Technology: Manufacturing philosophy helping to manage diversity by identifying similarities in products and activities (carrying out similar activities, standardizing similar assignments, filing and tracking down in an efficient way information regarding recurrent problems).

Heuristic Approach: Methodology proposed to the resolution of a problem aimed at finding an acceptable solution, but in general not able to identify the optimal one.

Productive Ants: Agents with the aim to optimize the match between cell capacity and the load generated by the associated product families.

Similarity Coefficient: Coefficient measuring the number of common items for a couple of machines, adopted in several methods of cell formation.

Technologist Ants: Agents with the aim to optimize the match between cell capability and the technological requirements determined by the associated product families.

APPENDIX A: NOTATION

N = number of iterations of tech-ants
M = number of iterations of prod-ants
N_{tech} = number of tech-ants within the system
N_{prod} = number of prod-ants within the system
CS_{hj} = similarity coefficient between machine h and the group of machines j
$P_{mach,hj}$ = probability value to move machine h to cell j
β = parameter set by the user
N_{mach} = number of machines in a production department
Mem = size of ants' memory
SW = standard weight for machine h, corresponding to the minimum inter-cell moving cost
γ = coefficient set by the user, included between 0 and 1
C = fixed constant linked to cost of single out-of-cell operation
n_i = annual number of lots for product i
W = total number of realized products
TL_h = target level of saturation specified by the user for machine h
AL_h = actual level due to the job selected by the ant on machine h
α_h = weight of machine h linked to its capacity availability
Δ_{hi} = deviation of the workload on machine h from the target level (the job is supposed to be associated to the cell i, i.e., the original cell)
Δhj = deviation of the workload on machine h from the target level (the job is supposed to be associated to cell j, i.e., the original cell)
$P_{job,ij}$ = probability value to move a job from cell i to cell j

APPENDIX B: SIMULATION RESULTS

Variance Alalysis Output

Factor	Type	Levels	Values
Memory size	Fixed	3	1 2 3
Objective level	Fixed	3	1 2 3
Penalty	Fixed	3	1 2 3
Ratio tech-ants/prod-ants	Fixed	3	1 2 3

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Memory size	2	1.4551E+18	1.4956E+18	7.4780E+17	15.45	0.000
Objective level	2	1.6755E+18	1.6780E+18	1.3900E+17	17.34	0.000
Penalty	2	3.6004E+17	1.9786E+17	1.9893E+17	4.11	0.019
Ratio tech-ants/prod-ants	2	7.3381E+19	1.3392E+19	1.6696E+19	758.27	0.000
Memory size * Objective level	4	6.1884E+16	1.2471E+16	1.5618E+16	0.32	0.862
Objective level * Penalty	4	1.4550E+18	1.4551E+18	1.6378E+17	7.52	0.000
Objective level * Ratio tech-ants/prod-ants	4	1.4975E+18	1.4975E+18	1.7438E+17	7.74	0.000
Error	113	5.4686E+18	1.4686E+18	1.8394E+16		
Total	133	8.5355E+19				

Average % Reduction of Total Cost Compared to the Initial Configuration

Factor	Level	% Reduction
Memory size	1	46.6
	2	39.2
	3	49.2
Objective level	1	39.5
	2	44.9
	3	50.6
Penalty	1	46.6
	2	46.3
	3	42.1
Ratio tech-ants/prod-ants	1	3.1
	2	60.7
	3	71.5

Chapter XXXV
Agent-Oriented Modeling and Simulation of Distributed Manufacturing

Kuldar Taveter
University of Melbourne, Australia

Gerd Wagner
Bradenberg University of Technology at Cottbus, Germany

ABSTRACT

This chapter proposes an agent-oriented method for modeling and simulation of distributed production environments. The proposed method views a manufacturing enterprise as consisting of active entities—agents. The method makes use of the Radical Agent-Oriented Process (RAP) methodology introduced by Taveter and Wagner (2005) which is based on Agent-Object-Relationship (AOR) modeling. The chapter first presents the AOR Modeling Language and the RAP/AOR viewpoint modeling framework of the methodology. Thereafter its lays out principles of turning the modeling constructs of the RAP/AOR methodology into the implementation constructs of a simulation environment and briefly describes the simulation environment. The method is aimed at the creation of environments for modeling and simulation of distributed manufacturing.

INTRODUCTION

With the advent of virtual enterprises, new business models emerge. In them, a manufacturing enterprise should be capable of composing its manufacturing processes in a modular fashion so that if the factory receives an order at short notice, the satisfaction of which requires only a part of a full-length manufacturing process of the enterprise, the order will be scheduled and satisfied in a dynamic, flexible, and fast manner. One way to achieve this is to view a manufacturing enterprise as a collection of active entities—*agents*—so that each resource would be represented by an agent responsible for scheduling and performing its manufacturing operations. An agent is autonomous and does not know the decision logic of

the other agents as a rule. The decision logic is thus specified for each agent individually and not for the system of agents as a whole. Differently from conventional modeling approaches, including UML (OMG, 2003a, 2003b), this is closer to how real socio-technical systems—consisting of both human and technical components—operate. This is why our approach can be characterized as inspired by nature.

THE RAP/AOR METHODOLOGY

The Radical Agent-Oriented Process/Agent-Object-Relationship (RAP/AOR) methodology of simulation and software engineering, which was introduced by Taveter and Wagner (2005), is based on the Agent-Object-Relationship Modeling Language (AORML) proposed by Wagner (2003a) and the Business Agents' approach presented in Taveter (2004). The ontological foundation of the RAP/AOR concepts is provided by the Unified Foundational Ontology (UFO) proposed by Guizzardi and Wagner (2005). The UFO defines an ontological distinction between active and passive entities—that is, between agents and (non-agentive) objects of the real world. The agent metaphor subsumes *artificial* (software and robotic), *natural* (human and animal), as well as *social/institutional* agents (groups and organizations). We will subsequently describe AORML, which is used as the main graphical description for work products of RAP/AOR. Thereafter, we will introduce the RAP/AOR viewpoint modeling framework forming the core of the methodology.

The AOR Modeling Language

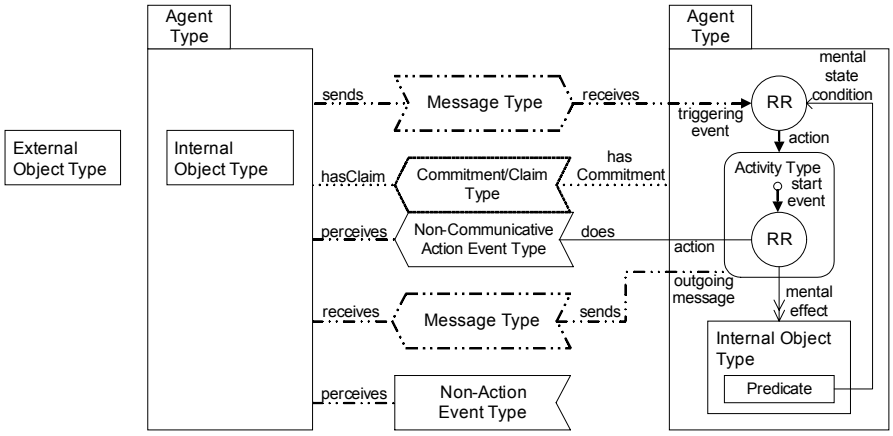
In AORML, an entity is an agent, an event, an action, a claim, a commitment, or an ordinary object. Only agents can communicate, perceive, act, make commitments, and satisfy claims. Objects are passive entities with no such capabilities. Besides human and artificial agents, AORML also includes the concept of *institutional agents,* which are composed of a number of other agents that act on their behalf. Organizations and organizational units are important examples of institutional agents.

There are two basic types of AOR models: *external* and *internal* models. An external AOR model adopts the perspective of an external observer who is looking at the (prototypical) agents and their interactions in the problem domain under consideration. In an internal AOR model, the internal (first-person) view of a particular agent to be modeled is adopted. While a (manufacturing) domain model corresponds to an external model, a design model (for a specific agent-oriented information system) corresponds to an internal model which can be derived from the external one. Since the use of external AOR models suffices for the purposes of simulation, in this chapter internal AOR models are treated only marginally.

An *external AOR diagram* specified by Figure 1 shows how the types and instances (if applicable) of institutional, human, and artificial (for example, software) agents of a problem domain can be represented, together with their internal agent types and instances and their beliefs about instances of "private" and external ("shared") object types. There may be attributes and/or predicates defined for an object type and relationships (associations) among agent and/or object types. A predicate, which is visualized as depicted in Figure 1, may take parameters. As in UML (OMG, 2003a, 2003b), an instance of a type is graphically rendered by a respective rectangle with the underlined name of the particular instance as its title.

As formulated in Wagner (2003a) and reflected by Figure 1, if an object type belongs exclusively to one agent or agent type, the corresponding rectangle is drawn inside of this agent or agent type rectangle. Otherwise, if the

Figure 1. The core mental state structure and behavior modeling elements of external AOR diagrams



object type rectangle is drawn outside of the respective agent or agent type rectangles, the focus agents have by default beliefs of the corresponding structure about its instances. However, Wagner (2003a) has emphasized that an external object type does not imply that all the agents connected by an association to it have the same beliefs about it, or in other words, that there is a common extension of it shared by all agents.

Sometimes a belief of an agent needs a different representation, using different attributes and predicates, because it does not directly correspond to the structure of an "external" object type, but rather to the agent's "personal" view of it. For such a case, we have extended AORML by the UML dependency arrow with the stereotype <<represents>> between the internal representation and the corresponding external object type. For example, there is an internal representation of the object type ProductionOrder within the agent CeramicsFactory.

Figure 1 shows that the graphical notation of AORML distinguishes between an *action event type* (an event that is created through the action of an agent, such as starting a machine) and a *non-action event type* (for example, types of temporal events or events created by natural forces). The graphical notation of AORML further distinguishes between a *communicative* action event (or *message*) type and a *non-communicative* (physical) action event type like providing another agent with a commodity.

Two kinds of *commitments* may occur between agents: commitments to perform actions of certain types, such as a commitment of the SalesDepartment of the CeramicsFactory towards a Customer to provide it with a product set, and commitments to see to it that some condition holds, such as a commitment of the ProductionDepartment towards the SalesDepartment to have a ProductionOrder completed. The former are called *to-do* commitments and the latter *see-to-it-that (stit)* commitments. A *stit*-commitment is used for modeling situations where one agent directly requests another agent to make true some proposition that is expressed in terms of predicates defined for object types. An achieve-modeling construct type denotes achieving—that is, making true—a proposition.

In an external AOR model, a *commitment* of agent a_1 towards agent a_2 to perform an action of a certain type (such as a commitment to pay for a product set) can also be viewed as a *claim* of a_2 against a_1 that an action of that kind will be performed. Figure 1 reflects that a *commitment/claim type* is coupled with the action event type whose instance fulfills the corresponding commitment (or satisfies the claim). Analogously, an achieve-construct type is coupled with the corresponding *stit*-commitment/claim type. An achieve-construct type and the *stit*-commitment/claim type coupled with it are visualized like an action event type and the commitment/claim type coupled with it shown in Figure 1 but drawn with a thick line.

In an external AOR model, there are four types of designated relationships between agents and action events: sends and receives are relationship types that relate an agent with communicative action events, while does and perceives are relationship types that relate an agent with non-communicative action events. A relationship of the perceives type also relates an agent to non-action events perceived by it. In addition, there are two types of designated relationships between agents and commitments/claims: hasCommitment and hasClaim. These designated relationship types are visualized with particular connector types as depicted in Figure 1.

As Wagner (2003b) has shown, mental state structure modeling in AORML can be defined as a UML Profile, that is, it is a conservative extension of UML class modeling.

The most important behavior modeling elements of AORML are *reaction rules.* As is shown in Figure 1, a reaction rule is visualized as a circle with incoming and outgoing arrows drawn within the rectangle of the agent type or instance whose reaction pattern it represents. Each reaction rule has exactly one incoming arrow with a solid arrowhead that specifies the *triggering event* type. In addition, there may be ordinary incoming arrows representing *mental state conditions* (referring to corresponding instances of other object types or to the predicates defined for them). There are two kinds of outgoing arrows: one for specifying *mental effects* (changing beliefs and/or commitments) and the other one for specifying the performance of (physical and/or communicative) *actions*. An outgoing arrow with a double arrowhead denotes a mental effect. An outgoing connector to an action event type denotes the performance of an action of that type. As Taveter and Wagner (2001) have shown, reaction rules are the most important type of business rules.

Reaction rules start activities. An *activity* is defined using workflow terminology as an uninterruptible amount of work that is performed in a non-zero span of time by an actor[1] (Eshuis, Jansen, & Wieringa, 2002). Each activity belongs to some *activity type,* which is visualized in the way depicted in Figure 1. An *activity type* (or *task* in Yu, 1995), like "Process production order," is defined as a prototypical job function in an organization that specifies a particular way of doing something by performing one or more elementary epistemic, physical, and communicative actions in a non-zero span of time by an agent.

It seems natural to allow specifying the start of an activity in the action part of a reaction rule as shown in Figure 1. In other words, an instance of an activity type is created by means of a reaction rule in response to perceiving an event. Dataflow through an activity in the course of its execution is represented as *input parameters* of the activity. In an external AOR diagram, the input parameters that are passed to the activity are defined in parentheses following the name of the activity type. Additionally, one can define for each activity type in terms of its input parameters the *goal* that its instances try to achieve.

There are activity border events of two types (*start-of-activity* and *end-of-activity*) implicitly associated with the beginning and end of each activity. An activity border event starts either a sub-activity or a subsequent activity, or triggers a reaction rule. As Figure 1 reflects, the *start-of-activity* event type is graphically represented by an empty circle with the outgoing arrow to the symbol of the sub-activity type or internal reaction rule. The *end-of-activity* event type is visualized by drawing a triggering arrow from the activity type symbol to either the symbol of the next activity type or to the symbol of the reaction rule triggered by an activity of the corresponding type.

Taveter (2004) has shown that AORML extended by activity modeling allows the representation of 16 out of 19 behavioral workflow patterns as defined in the workflow benchmark proposal of van der Aalst, ter Hofstede, Kiepuszewski, and Barros (2003). Examples of such behavioral patterns are the sequence and parallel split of activities and execution of activities in loops of various types.

An external AOR diagram of the kind shown in Figure 1 can be considered as a specification of a high-level state transition system where the state of an agent consists of two parts: its *mental state* (beliefs, memory of events, actions, and commitments/claims) and its *activity state*. Taveter (2004) has formalized the execution cycle of such an agent based on the semantic framework of *Knowledge-Perception-Memory-Commitment* (KPMC) agents proposed by Wagner and Schroeder (2000).

An external AOR diagram represented by Figure 1 can be decomposed into one or more diagrams of the following types:

- **AOR Agent Diagrams:** Depicting the agent types (and instances, if applicable) of the domain, certain relevant object types, and the relationships among them;
- **AOR Interaction Frame Diagrams:** Depicting the action event types and commitment/claim types that determine the possible interactions between instances of two agent types;
- **AOR Interaction Sequence Diagrams:** Depicting prototypical instances of interaction processes;
- **AOR Interaction Pattern Diagrams:** Focusing on general interaction patterns expressed by means of a set of reaction rules which define an interaction process type; and
- **AOR Activity Diagrams:** As specifications of parameterized behaviors at different levels of granularity that are expressed as flows of execution via sequencing of subordinate activities whose primitive elements are individual epistemic, physical, and communicative actions.

Examples of diagrams of most of the kinds defined above are presented in the case study of Taveter (2006).

An internal AOR model may comprise one or more diagrams of the following types, in addition to AOR agent diagrams:

- **AOR Reaction Frame Diagrams:** Depicting other agents (or agent types) and the action and event types in the internal perspective of an agent, as well as the commitment and claim types that determine the possible interactions with them;
- **AOR Reaction Sequence Diagrams:** Depicting prototypical instances of interaction processes in the internal perspective of an agent;
- **AOR Reaction Pattern Diagrams:** Focusing on the reaction patterns of the agent under consideration expressed by means of reaction rules; and

Table 1. The RAP/AOR viewpoint modeling framework

Viewpoint Models	***Viewpoint Aspect***		
Abstraction Level	*Interaction*	*Information*	*Behavior*
Conceptual Domain Modeling	AOR Agent Diagrams, UML Use Case Diagrams, AOR Interaction Frame Diagrams, AOR Interaction Sequence Diagrams	AOR Agent Diagrams	Goal-Based Use Case Models (Cockburn, 1997a, 1997b, 2001), Goal Models (Kuan, Karunasakera, & Sterling, 2005), AOR Interaction Pattern Diagrams, AOR Activity Diagrams
Platform-Independent Computational Design	UML Use Case Diagrams, AOR Reaction Frame Diagrams, AOR Reaction Sequence Diagrams, User Interface Design Models, Security Models, UML Class Diagrams, UML Interaction Diagrams	AOR Agent Diagrams	AOR Reaction Pattern Diagrams, AOR Activity Diagrams, UML State Machine Diagrams
Platform-Specific Implementation	UML Deployment Diagrams	UML Class Diagrams	UML Class Diagrams

- **AOR Activity Diagrams:** As specifications of parameterized behaviors in the internal perspective of an agent.

The RAP/AOR Viewpoint Modeling Framework

While in the previous section we provided an overview of the graphical notation used in the RAP/AOR methodology, in this section we describe the methodology itself. The core of the methodology is formed by the RAP/AOR viewpoint modeling framework described in Table 1. It is based on the six perspectives of agent-oriented business modeling proposed by Taveter (2004) which is, in turn, rooted in the ideas of the Zachman framework (Sowa & Zachman, 1992). The RAP/AOR viewpoint modeling framework is also well aligned with the Model-Driven Architecture (MDA, http://www.omg.org/mda/) framework of the Object Management Group (OMG). It consists of a matrix with three rows representing different abstraction levels and three columns representing the viewpoint aspects *interaction*, *information,* and *behavior.* Each cell in this matrix represents a specific viewpoint, such as *conceptual interaction modeling, computational information design,* or *behavior implementation*.

Normally one or more views are created for each viewpoint using the respective modeling language(s). A *view* is a diagram or a model of another kind, such as a tabular use case or a textual description. In the following, different viewpoints of the framework will be briefly described.

The *domain interaction viewpoint* (column 1 in Table 1) concerns the analysis and modeling of *active entities*, that is, of agent types and instances and relationships, as well as the *interactions* and *communication* between them. The domain interaction viewpoint comprises organization modeling. The purposes of organization modeling are to identify:

a. the organization(s) of the problem domain,
b. the relevant organizational units of which each organization to be modeled consists,
c. the *roles*[2] included by the organizational units, and
d. the types of relationships occurring between these agent types.

According to Zambonelli, Jennings, and Wooldridge (2001), among the five types of relationships that can be identified between institutional agent types and/or role types, control, benevolence, and dependency relationships are the most relevant ones to modeling interactions between agents. *Control* relationships identify the authority structures within an organization. *Benevolence* relationships identify agents with shared interests. *Dependency* relationships exist between agents because of resource restrictions where the depender depends on the dependee for the availability of a physical or an informational resource.

We represent types of organizational units and roles by AOR agent diagrams where different agent types may relate to each other through the relationships of *generalization* and *aggregation* in addition to relationships of the types described above. An important purpose of an agent diagram is to describe all *stakeholders* that are involved in the manufacturing processes to be modeled and simulated, and to give an overview of the manufacturing system viewed as a multi-agent system.

Additionally, we model the possible interactions between two (types of) agents by means of AOR interaction frame diagrams. Table 1 also lists diagrams of other types that can be used for modeling from the domain interaction viewpoint.

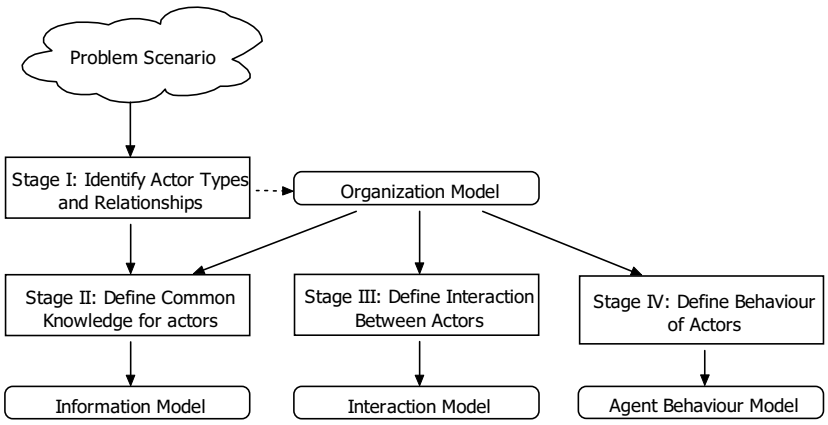
Representing the *domain information viewpoint* (column 2 in Table 1) for the focus organization(s) can be regarded as creating a *domain ontology* which provides a common framework of knowledge for the agents of the organization(s) and external agents connected to the organization(s). Each agent of the problem domain can see only a part of the ontology; that is, each agent views the ontology from a specific perspective.

The domain information viewpoint is described with the help of one view—AOR agent diagrams. In addition to describing agent types from the domain interaction viewpoint, an AOR agent diagram thus describes object types of the problem domain, as well as their relationships to agent types and with each other. The beliefs of an agent thus include its internal agents and all external agents related to it, in addition to the agent's "private" objects and shared objects that are related to it.

The *domain behavior viewpoint* (column 3 in Table 1) addresses the modeling of an agent's functionality (what functions the agent has to perform), as well as of the agent's behavior (when, how, and under what conditions work has to be done).

Actor types (or *agent role types*) are always characterized by *goals* because, as noted by Kueng and Kawalek (1997, p. 19): "Human activity is inherently purposeful." In a business or manufacturing domain, a human or an institutional agent acting in the role of a "customer" has a goal of having something accomplished. To achieve its goal, the agent uses some service provided by another agent. An agent's autonomy implied by a *benevolence* relationship between the service provider and a service requester means that the service provider performs the service requested if it is able to do so, but the service provider also *has an option to refuse the service request.* Even though the agent requesting the service may not explicitly communicate its goals to the service provider agent, the latter always "internalizes" the whole or a part of the customer's goal in an attempt to provide the service. For example, assuming

Figure 2. The modeling process according to the RAP/AOR methodology



that a customer—wholesaler—has a goal of reselling a set of ceramics products, the goal of the ceramics factory is to provide the wholesaler with a product set, which is of course a sub-goal of the factory's higher-level goal—to earn money through producing ceramic items. The ceramic factory tries to achieve this higher-level goal by "internalizing" as many customer goals as possible.

We model the "internalizations" of the goals of customers by service providers by using AOR activity diagrams. In addition to them, diagrams and models of the types listed in Table 1 can be used for the same purpose.

As explained earlier, an interacting system (or agent), as a subject in its domain, does not have an objective but a subjective view of the domain. This is reflected in RAP/AOR by a computational design model, in which the internal (subjective) perspective of the system to be built is adopted in contrast to the external (objective) perspective adopted in a conceptual domain model. For instance, in the transformation of a domain interaction model into an interaction design model for a specific agent, the objective term *action event* is mapped onto the two indexical subjective terms *action* (if performed by the agent under consideration) and *event* (if performed by other agents). Likewise, the objective term *message* is mapped onto the two subjective terms *incoming message* and *outgoing message.*

External models of the conceptual domain modeling level are thus transformed into internal models of the level of platform-independent computational design. In particular, AOR agent diagrams are refined into more detailed agent diagrams. Analogously, AOR interaction frame diagrams are turned into reaction frame diagrams, AOR interaction pattern diagrams into reaction pattern diagrams, and AOR activity diagrams into AOR activity diagrams represented from the perspective of a specific agent. However, as has been mentioned before, in this chapter we are interested in simulation of domain models rather than transforming them into design and implementation models.

The stages of conceptual domain modeling in the RAP/AOR methodology and the resulting models are represented in Figure 2. Stages I and III make up parts of conceptual interaction modeling. They result in the organization model described as an agent diagram and the interaction model represented as one or more

interaction frame diagrams. Stage II corresponds to conceptual information modeling and stage IV to conceptual behavior modeling. These stages respectively result in the information model, which is represented as an agent diagram and the agent behavior model in the form of AOR activity diagrams.

IMPLEMENTATION OF THE SIMULATION ENVIRONMENT

Taveter (2004) demonstrated that AORML extended by activities is the first agent-oriented modeling language where fully as well as partially specified domain models can be executed. On the other hand, Wagner and Tulba (2003) have shown that, with some minor extensions, AOR models can be used for a certain form of agent-based discrete event simulation, called *Agent-Object-Relationship Simulation* (AORS). An AORS system includes an environment simulator that is responsible to simulate external events, such as orders issued by customers, and in some cases, the causality laws of the physical environment. Using executable process models jointly with an AORS system described by Wagner and Tulba (2003) thus permits the creation of powerful simulation environments.

Taveter (2004) has also made clear that external AOR diagrams can be straightforwardly transformed into the programming constructs of the JADE agent platform. The Java Agent Development Environment (JADE, http://jade.cselt.it/) agent platform is a software framework to build software agent systems in the Java programming language for the management of networked information resources in compliance with the Foundation for Intelligent Physical Agents (FIPA, http://www.fipa.org/) specifications for interoperable intelligent multi-agent systems. In addition to providing an agent development model, JADE deals with all the aspects that are not peculiar to agent internals and that are independent of the applications, such as message transport, encoding, and parsing or agent lifecycle management. According to Bellifemine, Poggi, and Rimassa (2001), JADE offers the following features to the agent programmer:

- FIPA-compliant distributed agent platform which can be split onto several hosts;
- Java Application Programmer's Interface to send/receive messages to/from agents;
- library of FIPA interaction protocols, such as Contract Net, ready to be used; and
- graphical user interface to manage several agents from the same Remote Management Agent.

A JADE agent must be able to carry out several concurrent tasks in response to different external events. These tasks, which are termed *behaviors* in JADE, correspond to activities in RAP/AOR. All the behaviors of a JADE agent are implemented as instances of the Java object class jade.core.behaviors.Behavior. The base class Behavior has the predefined subclasses SimpleBehavior and CompositeBehavior. The first of them has been further divided into the subclasses OneShotBehavior and CyclicBehavior, while the second one has the subclasses SequentialBehavior and ParallelBehavior. As is reflected by Table 2, the classes OneShotBehavior, SequentialBehavior, and ParallelBehavior correspond to the respective activity types of RAP/AOR, while the class CyclicBehavior is used for implementing the execution cycle of a KPMC agent. When an activity type is mapped to behavior of JADE, an extension for the appropriate subclass of Behavior is defined and instantiated, and the resulting behavior object is then added to the agent behavior list. Before that, the jade.core.Agent

Table 2. Mapping of notions of AORML to the object classes and methods of JADE

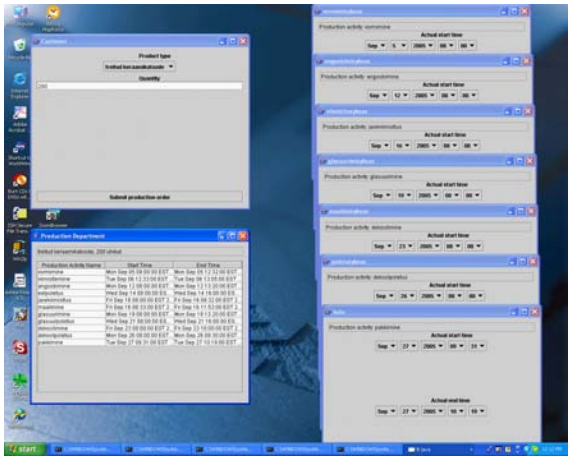
Notion of RAP/AOR	Object Class in JADE	Object Method of JADE (if applicable)
Object type	java.lang.Object	-
Agent type	jade.core.Agent	-
Elementary activity category	jade.core.behaviors. OneShotBehavior	-
Sequential activity category	jade.core.behaviors. SequentialBehavior	-
Parallel activity category	jade.core.behaviors. ParallelBehavior	-
Execution cycle of a KPMC agent	jade.core.behaviors. CyclicBehavior	-
Waiting for a message to be received	jade.core.behaviors. ReceiverBehavior	-
Starting the first-level activity	jade.core.Agent	public void addBehavior (Behavior b)
Starting a sub-activity	jade.core.behaviors. SequentialBehavior	public void addSubBehavior (Behavior b)
Starting a parallel sub-activity	jade.core.behaviors. ParallelBehavior	public void addSubBehavior (Behavior b)
Start-of-activity activity border event type	jade.core.behaviors. OneShotBehavior	public abstract void action()
Start-of-activity activity border event type	jade.core.behaviors. SequentialBehavior, jade.core.behaviors. ParallelBehavior	public abstract void onStart()
End-of-activity activity border event type	jade.core.behaviors.Behavior	public int onEnd()
Agent message	java.lang.acl.ACLMessage	-

class is extended for each agent type of the problem domain that has been modeled using RAP/AOR. The jade.core.Agent class exposes two methods, addBehavior(Behavior) and removeBehavior (Behavior), which allow management of the behaviors' queue of a specific JADE agent. As Table 2 reflects, by using these two methods within an instance of the jade.core.Agent class (or its extension), behaviors can be added to the agent whenever needed. Analogously, by invoking the method addSubBehavior(Behavior) within an instance of SequentialBehavior or ParallelBehavior, a sub-behavior can be added.

The functionality of a behavior is included in its action() method. The Behavior class also provides two placeholder methods, named onStart() and onEnd(). The functionality of a SequentialBehavior and ParallelBehavior is included in the method onStart() in place of action(). As is shown in Table 2, the action() and onStart() methods form counterparts of the *start-of-activity* event type, while the onEnd() method corresponds to the *end-of-activity* event type.

The mappings presented in Table 2 formed the basis for creating a simulation environment of the ceramics factory. The snapshot of the simulation environment is represented in Figure 3. The snapshot shows the user interfaces for the agent representing the Customer and the agents of the ProductionDepartment, and several ResourceUnits of the factory. Actual simulation by means of the simulation environment is discussed in Taveter (2006).

Figure 3. A snapshot of the simulation environment for the ceramics factory



RELATED WORK

Other similar agent-oriented modeling and simulation approaches have been used, for instance in the following research areas:

- **Biology:** For example, for investigating eco-systems or in population ethnology (especially with respect to ants and other insects) (Klügl, 2001);
- **Engineering:** For analyzing and designing complex socio-technical systems, such as Automatically Guided Vehicle Transport Systems (Raffel, Taveter, & Wagner, 2001);
- **Economics:** For example, in the simulation of auctions and markets (e.g., Trading Agent Competition—TAC, http://www.sics.se/tac/) and in the simulation of supply chains (Labarthe, Tranvouez, Ferrarini, Espinasse, & Montreuil, 2003); and
- **Social Sciences:** For example, the phenomena of social monitoring and norm-based social influence are studied in Conte and Dignum (2001), and the cooperation in teams is studied in Hales (2002).

Bandini, Manzoni, and Simone (2002) propose the multi-layered multi-agent situated system model (MMASS), a model for multi-agent systems situated in a physical environment. One of its applications is a crowding dynamics simulation model reported by Bandini, Manzoni, and Vizzari (2004).

Some well-known platforms for agent-based simulation are Swarm (http://www.swarm.org), SDML (Moss, Gaylard, Wallis, & Edmonds, 1998), Sesam (Klügl, 2001), and CORMAS (Bousquet, Bakam, Proton, & Le Page, 1998). A particularly interesting class of simulation systems is formed from international technology competitions, such as RoboCup (Noda, Matsubara, Hiraki, & Frank, 1998) and TAC (http://www.sics.se/tac/). Both RoboCup and TAC can be classified as interactive agent-based real-time simulation systems.

CONCLUSION

In this chapter, a method for agent-oriented modeling and simulation of distributed manufacturing has been put forward. We first provided an overview of the graphical AOR Modeling Language and the RAP/AOR methodology that the method is based on. After that, we described how the models obtained can be transformed into the constructs of the JADE agent platform for simulation.

Two particular strengths of the proposed agent-oriented modeling approach are its ability to model distributed systems in a straightforward and natural way, and the *executability* of partially as well as completely specified external AOR diagrams. The latter facilitates simulation. Another strength of our approach is the possibility to represent the models of the inter-

action, information, and behavior viewpoint aspects in just one integrated diagram at the desired level of granularity. This enables one to overcome the *model multiplicity problem* (Peleg & Dori, 2000), which states that to understand the system being studied and the way it operates and changes over time, the reader must concurrently refer to various models. Agent-oriented modeling and simulation may also serve as the first stages of agent-based or another kind of automation.

ACKNOWLEDGMENT

The author is thankful to Fernando Koch for drawing Figure 2.

REFERENCES

Bandini, S., Manzoni, S., & Simone, C. (2002). Heterogeneous agents situated in heterogeneous spaces. *Applied Artificial Intelligence, 16*(9-10), 831-852.

Bandini, S., Manzoni, S., & Vizzari, G. (2004). Situated cellular agents: A model to simulate crowding dynamics. *IEICE Transactions on Information and Systems, E87-D*(3), 669-676.

Bellifemine, F., Poggi, A., & Rimassa, G. (2001). Developing multi-agent systems with a FIPA-compliant agent framework. *Software—Practice and Experience, 31,* 103-128.

Bousquet, F., Bakam, I., Proton, H., & Le Page, C. (1998, June 1-4). Cormas: Common-pool resources and multi-agent systems. In A.P. Del Pobil, J. Mira, & M. Ali (Eds.), *Proceedings of Tasks and Methods in Applied Artificial Intelligence, the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems* (IEA/AIE-98), Castellon, Spain (LNCS 1416, pp. 826-837). Berlin: Springer-Verlag.

Cockburn, A. (1997a). Goals and use cases. *Journal of Object-Oriented Programming, 10*(5), 35-40.

Cockburn, A. (1997b). Using goal-based use cases. *Journal of Object-Oriented Programming*, *10*(7), 56-62.

Cockburn, A. (2001). *Writing effective use cases*. Reading, MA: Addison-Wesley.

Conte, R., & Dignum, F. (2001). From social monitoring to normative influence. *Journal of Artificial Societies and Social Simulation, 4*(2) (electronic journal). Retrieved October 11, 2005, from http://jasss.soc.surrey.ac.uk/4/2/7.html/

Eshuis, R., Jansen, D. N., & Wieringa, R. J. (2002). Requirements-level semantics and model checking of object-oriented statecharts. *Requirements Engineering Journal, 7*(4), 243-263.

Guizzardi, G., & Wagner, G. (2005, June 8/July 20). Towards ontological foundations for agent modeling concepts using the Unified Foundational Ontology (UFO). In P. Bresciani, B. Henderson-Sellers, G. Low, & M. Winikoff (Eds.), *Proceedings of the 6th International Bi-Conference Workshop on Agent-Oriented Information Systems* (AOIS 2004), Riga, Latvia/New York (LNAI 3508, 110-124, pp. 110-124). Berlin: Springer-Verlag.

Hales, D. (2002, July 15-16). Evolving specialization, altruism and group-level optimization using tags. In J. S. Sichman, F. Bousquet, & P. Davidsson (Eds.), *Proceedings of the 3rd International Workshop on of Multi-Agent-Based Simulation* (MABS 2002), Bologna, Italy (LNAI 2581). Berlin: Springer-Verlag

Klügl, F. (2001). *Multiagentensimulation*. Reading, MA: Addison-Wesley.

Kuan, P.P., Karunasakera, S., & Sterling, L. (2005, March 31-April 1). Improving goal- and role-oriented analysis for agent-based systems. *Proceedings of the 16th Australian Software Engineering Conference* (ASWEC 2005), Brisbane, Australia (pp. 40-47).

Kueng, P., & Kawalek, P. (1997). Goal-based business process models: Creation and evaluation. *Business Process Management Journal, 3*(1), 17-38.

Labarthe, O., Tranvouez, E., Ferrarini, A., Espinasse, B., & Montreuil, B. (2003, September 1-3). A heterogeneous multi-agent modeling for distributed simulation of supply chains. In V. Marik, D.C. McFarlane, & P. Valckenaers (Eds.), *Proceedings of the 1st International Conference on Industrial Applications of Holonic and Multi-Agent Systems* (HoloMAS 2003), Brisbane, Australia (LNCS 2744, pp. 134-145). Berlin: Springer-Verlag.

Moss, S., Gaylard, H., Wallis, S., & Edmonds, B. (1998). SDML: A multi-agent language for organizational modeling. *Computational and Mathematical Organization Theory, 4*(1), 43-70.

Noda, I., Matsubara, H., Hiraki, K., & Frank, I. (1998). Soccer server: A tool for research on multi-agent systems. *Applied Artificial Intelligence, 12*(2-3), 233-250.

OMG. (2003a, March). *Unified Modeling Language specification, version 1.5*. Retrieved October 11, 2005, from http://www.omg.org/cgi-bin/doc?formal/03-03-01/

OMG. (2003b, August). *Unified Modeling Language: Superstructure, version 2.0*. Retrieved October 11, 2005, from http://www.omg.org/cgi-bin/doc?ptc/2003-08-02

Peleg, M., & Dori, D. (2000). The model multiplicity problem: Experimenting with real-time specification methods. *IEEE Transactions on Software Engineering, 26*(8), 724-759.

Raffel, W. U., Taveter, K., & Wagner, G. R. (2001, October). Agent-oriented modeling of business rules and business processes—The case of automatically guided transport systems. *Proceedings of Verbundtagung Verteilte Informationssysteme auf der Grundlage von Objekten, Komponenten und Agenten* (verIS2001) (pp. 72-81). Bamberg, Germany.

Sowa, J. F., & Zachman, J. A. (1992). Extending and formalizing the framework for information systems architecture. *IBM Systems Journal, 31*(3), 590-616.

Taveter, K. (2004). *A multi-perspective methodology for agent-oriented business modeling and simulation.* PhD thesis, Tallinn University of Technology, Estonia.

Taveter, K. (2006). Application of RAP/AOR to the modeling and simulation of a ceramics factory. In J.-P. Rennard (Ed.), *Handbook of research on nature-inspired computing for economy and management.* Hershey, PA: Idea Group Reference.

Taveter, K., & Wagner, G. (2001, November 27-30). Agent-oriented enterprise modeling based on business rules. In H. S. Kunii, S. Jajodia, & A. Sølvberg (Eds.), *Proceedings of the 20th International Conference on Conceptual Modeling*, Yokohama, Japan (LNCS 2224, pp. 527-540). Berlin: Springer-Verlag.

Taveter, K., & Wagner, G. (2005). Towards radical agent-oriented software engineering processes based on AOR modeling. In B. Henderson-Sellers & P. Giorgini (Eds.), *Agent-oriented methodologies* (pp. 277-316). Hershey, PA: Idea Group Publishing.

van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and Parallel Databases, 14*(3), 5-51.

Wagner, G., & Schroeder, M. (2000). Vivid agents: Theory, architecture, and applications. *Journal of Applied Artificial Intelligence, 14*(7), 645-675.

Wagner, G. (2003a). The agent-object-relationship meta-model: Towards a unified view of state and behavior. *Information Systems, 28*(5), 475-504.

Wagner, G. (2003b, July 15). A UML profile for external AOR models. In F. Giunchiglia, J. Odell, & G. Weiss (Eds.), *Proceedings of the 3rd International Workshop on Agent-Oriented Software Engineering* (AOSE 2002), Bologna, Italy (LNCS 2585, pp. 1380149). Berlin: Springer-Verlag.

Wagner, G., & Tulba, F. (2003). Agent-oriented modeling and agent-based simulation. In P. Giorgini & B. Henderson-Sellers (Eds.), *Conceptual modeling for novel application domains* (LNCS 2814, pp. 205-216). Berlin: Springer-Verlag.

Yu, E. (1995). *Modeling strategic relationships for process reengineering*. PhD thesis, Department of Computer Science, University of Toronto, Canada.

Zambonelli, F., Jennings, N. R., & Wooldridge, M. (2001). Organizational abstractions for the analysis and design of multi-agent systems. In P. Ciancarini & M. Wooldridge (Eds.), *Agent-oriented software engineering* (LNCS 1957, pp. 127-141). Berlin: Springer-Verlag.

ENDNOTES

1 Henceforth, we use the terms "actor" and "agent" as synonyms.

2 A *role* can be understood as an "abstract characterization of the behavior of a social actor within some specialized context or domain of endeavor" (Yu, 1995) such as the role Seller.

Chapter XXXVI
Application of RAP/AOR to the Modeling and Simulation of a Ceramics Factory

Kuldar Taveter
University of Melbourne, Australia

ABSTRACT

This chapter describes the application of the RAP/AOR methodology proposed by Taveter and Wagner (2005, 2006) to the modeling and simulation of a real ceramics factory. The chapter addresses the modeling of the ceramics factory from the interaction, information, and behavior aspects of the methodology. The chapter also discusses the simulation of the manufacturing processes of the ceramics factory by executing the process models developed using the RAP/AOR methodology. The method is aimed at the creation of simulation environments and automation systems of distributed manufacturing.

INTRODUCTION

According to Tamm, Puusepp, and Tavast (1987), the so-called "modeling by simulation" is one of the most practical means of conceptual analysis of a problem domain, because it enables learning of and experimenting on the target system with complex internal dependencies, and trying out the influences of decisions of informational, technological, and organizational nature.

On the other hand, new business models emerge. The importance of subcontracting is emphasized by, for example, Zeng and Sycara (1999): "In manufacturing, managers face 'make or buy' decisions, i.e., the choice of making components/products in house or subcontracting them to outside sources ... These decisions are critical in today's highly competitive and dynamic business environment." On the European scale, subcontracting is often the only way how the countries newly admitted to the European Union, as well as the other candidate countries from Central and Eastern Europe, are able to participate in the European division of labor. Primarily for this reason the

case study of the current chapter deals with the Tallinn Ceramics Factory Ltd. (http://www.keraamikatehas.ee) located in Tallinn, Estonia. Introducing new business models is already acute at the Tallinn Ceramics Factory because a remarkable portion of the orders received by it are sub-contractual orders for mug handles and stove tiles for fireplaces from Sweden and other countries.

In this chapter, the *Radical Agent-Oriented Process* (RAP) methodology of agent-oriented modeling is employed. The RAP methodology was proposed by Taveter and Wagner (2005, 2006), and it is based on the agent-object-relationship (AOR) Modeling Language (AORML, http://aor.rezearch.info). In AORML, the agents in a problem domain are distinguished from the (non-agentive) objects. The agents' actions, event perceptions, commitments, and claims are explicitly included in the models.

It was demonstrated earlier by Wagner and Tulba (2003) that agent-oriented modeling by AORML lends itself easily to simulation. In this chapter, we first show how business and manufacturing processes of the ceramics factory can be modeled by making use of the RAP/AOR methodology. Thereafter we address the actual simulation of the manufacturing processes by means of the simulation environment described by Taveter and Wagner (2006).

MODELING OF THE CERAMICS FACTORY

In this section, the modeling methodology proposed by Taveter and Wagner (2005, 2006) is applied to the case study of the Tallinn Ceramics Factory. We decided to model the factory in an agent-oriented manner for two reasons. Firstly, since there are communicating and interacting agents (actors[1]) everywhere, it is the most natural way and in a sense truly nature-inspired. Secondly, an agent-oriented modeling approach lends itself easily to simulation. As has been shown by Taveter and Wagner (2006), the models of the problem domain worked out by following the RAP/AOR methodology can thus be quite straightforwardly turned into the implementation constructs of the simulation environment. It is important to emphasize here that agent-oriented modeling and simulation of a problem domain does not imply an automation solution based on software agents.

Principles of Reactive Scheduling

The core of manufacturing processes of any factory lies in the scheduling of production operations. In real life, scheduling is reactive as much as predictive. A general scheduling solution utilized in the modeling and simulation effort described in this chapter is based on the works by Ow, Smith, and Howie (1988), Smith, Ow, Muscettola, Potvin, and Matthys (1990), and Smith (1995) because the method proposed in them can be easily transformed into an agent-oriented one. Note that applying the scheduling method described in the works referenced effectively means re-engineering, that is, improving the existing manufacturing processes of the factory.

According to Smith et al. (1990), the factory scheduling problem can be defined as one of coordinating sequences of manufacturing operations for multiple orders so as to:

- obey the temporal restrictions of production processes and the capacity limitations of a set of shared resources (i.e., machines); and
- achieve a satisfactory compromise with respect to a myriad of conflicting preferential constraints (i.e., meeting due dates, minimizing work-in-progress, etc.).

Smith et al. (1990) distinguish between two kinds of scheduling:

- **Predictive Scheduling:** Concerns an ability to effectively predict factory behavior through the generation of production plans that reflect both the full complexity of the factory environment and the stated objectives of the organization; and
- **Reactive Scheduling:** Concerns an ability to intelligently react to changing circumstances, as the factory is a dynamic environment where unexpected events (for example, machine breakdowns, quality control inspection failures) continually force changes to planned activities.

Scheduling in the ceramics factory is modeled by using a mixture of the OPIS system (Smith et al., 1990) and the agent-based cooperative scheduling system (Ow et al., 1988). According to Smith (1995), the simplest reactive methods invoked in response to conflict and opportunity events in OPIS are the Right Shifter and Left Shifter, respectively. The *Right Shifter* implements a reactive method that resolves conflicts by simply "pushing" the scheduled execution times of designated operations forward in time. The *Left Shifter* provides a similar method that "pulls" operations backwards in time (that is, closer to execution) to the extent that current resource availability and temporal process constraints will permit.

We will next describe modeling of the ceramics factory from the interaction, information, and behavior aspects of the RAP/AOR viewpoint modeling framework defined by Taveter and Wagner (2005), considering the principles of reactive scheduling presented above.

The Interaction Aspect

The interaction aspect of the RAP/AOR viewpoint modeling framework concerns the analysis and modeling of *active entities*—of agent types and instances and relationships, as well as of *interactions* and *communication* between them.

Modeling from the interaction aspect comprises *organization modeling* and *interaction modeling*, which will be explained and illustrated in the following two sub-sections.

Organization Modeling

According to Ow et al. (1988), production scheduling decisions for large and/or complex manufacturing facilities are often not made by any single individual. Rather, a group of people may be identified in the organization that cooperate and share information to develop and manage a production schedule. Because of the routine nature of the scheduling task, this group has usually adopted some organization structure to make the decision-making process efficient. As a drastic change to this organization structure is not desired, the AOR agent diagram of the ceramics factory's organization structure, shown in Figure 1, reflects the existing factory. However, in line with the principles of reactive scheduling, the organization structure has been complemented with some generalized agent types.

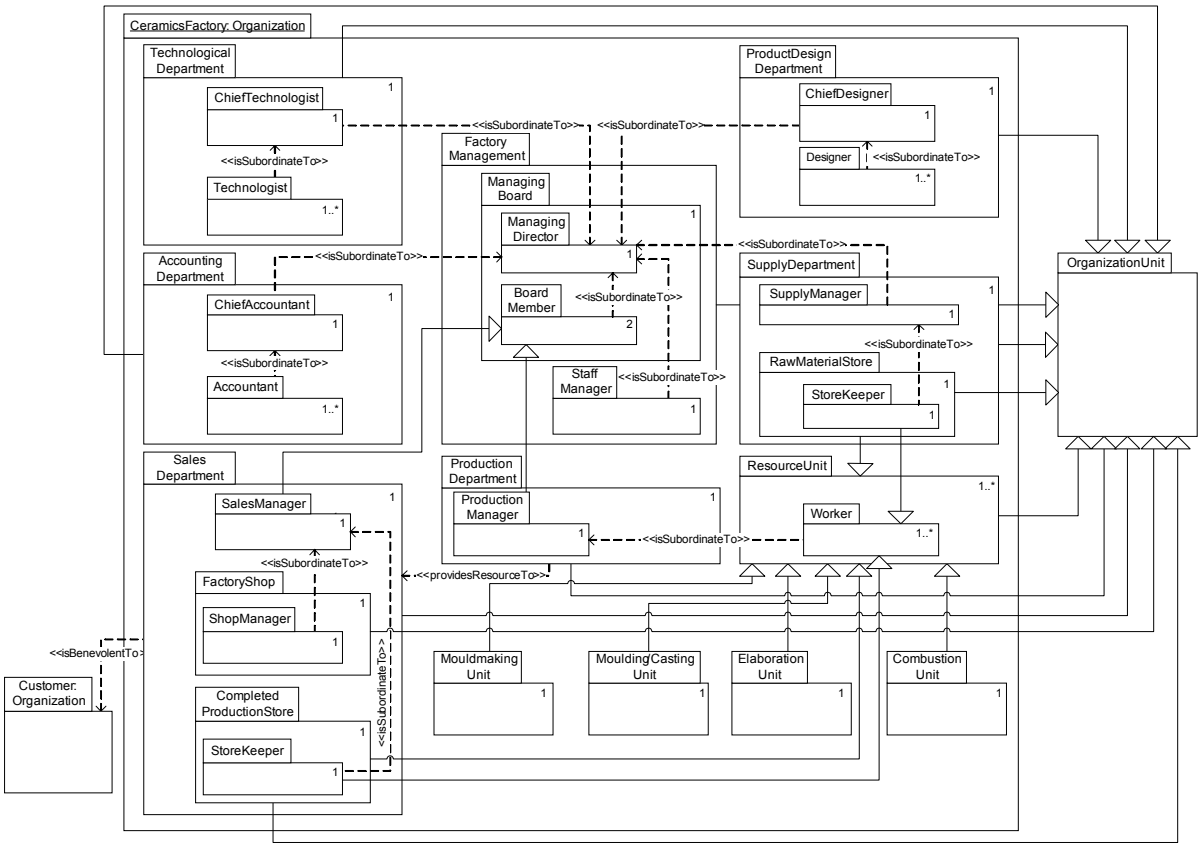
The agent diagram of Figure 1 represents the institutional agent instance CeramicsFactory which belongs to the agent type Organization. The agent CeramicsFactory consists of instances of the following subtypes of the institutional agent type OrganizationUnit, representing departments and other internal units of the factory: FactoryManagement, TechnologicalDepartment, AccountingDepartment, SalesDepartment, ProductionDepartment, ProductDesignDepartment,

SupplyDepartment, and ResourceUnit. The ResourceUnit has subtypes MouldmakingUnit, Moulding/CastingUnit, ElaborationUnit, and CombustionUnit, as well as RawMaterialStore and CompletedProductionStore. The number of instances of an agent type is shown in the top right corner of the box denoting the corresponding agent type. Since the factory depends on its customers, there is a benevolence (isBenevolentTo) relationship between the internal agent type SalesDepartment of the CeramicsFactory and the external agent type Customer.

The institutional agent type FactoryManagement includes the internal institutional agent type ManagingBoard. In the same way, the institutional agent type SalesDepartment includes the internal institutional agent types FactoryShop and CompletedProductionStore, and the institutional agent type SupplyDepartment includes the institutional agent type RawMaterialStore. Several institutional agent types modeled in Figure 1 include human role types, for example, ChiefTechnologist, Accountant, and Designer.

Within the ceramics factory, like within any other organization, there is a hierarchy of roles where one role is subordinate to another role. For example, in Figure 1 there is a control (isSubordinateTo) relationship between the human role types BoardMember and StaffManager on one hand and the human role type ManagingDirector on the other. The human role

Figure 1. The organization model of the ceramics factory



type BoardMember forms a supertype of the human role types SalesManager and ProductionManager. The human role types ManagingDirector and BoardMember are included by the internal institutional agent type ManagingBoard which reflects the fact that the ManagingDirector, as well as the SalesManager and ProductionManager, belong to the managing board of the factory by virtue of their offices.

There is also an isSubordinateTo relationship between several other human role types in Figure 1, such as, for example, between the internal agent types StoreKeeper and SalesManager of the SalesDepartment. A typical pattern of internal human agent types within an institutional agent type representing a department of the factory consists of the human role type of the department head, such as ChiefTechnologist, ChiefAccountant, SalesManager, ChiefDesigner, and SupplyManager, who has one or more human role types subordinated to it. All the human role types represented in Figure 1 are subtypes of the human role type EmployeeOfCeramicsFactory, which, in turn, forms a subclass of the agent type Person. For the sake of clarity of Figure 1, these agent types are not represented in it.

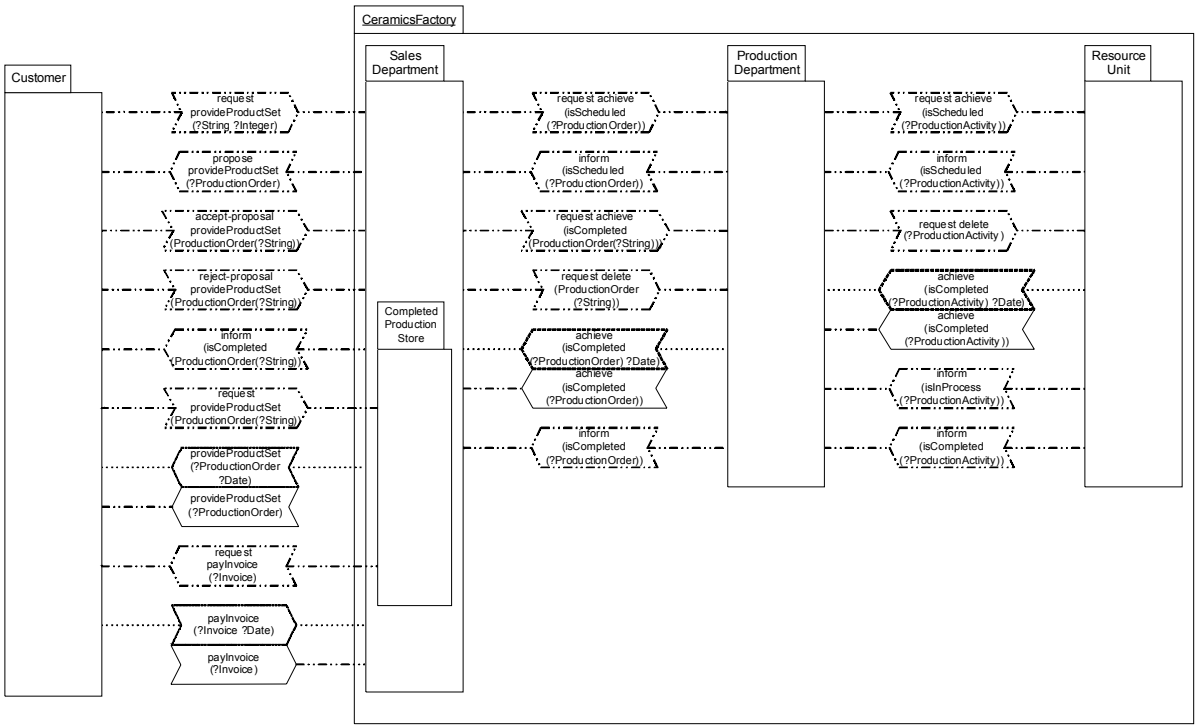
The institutional agent types ProductionDepartment and ResourceUnit respectively include the human role types ProductionManager and Worker where instances of Worker are subordinated to the instance of ProductionManager. The institutional agent type ResourceUnit has been introduced to enable the modeling and simulation according to the principles of reactive scheduling. In the ceramics factory to be modeled, there is no real agent type corresponding to ResourceUnit, even though workers effectively form teams according to their specialties.

Interaction Modeling

Interactions between a service requester and provider are modeled in the interaction frame diagram in Figure 2. Most of the communicative action event types represented in Figure 2 are prefixed by one of two *functions* of message types[2]: request, by which a sender requests the receiver to perform a communicative or physical action or both of them, and inform, which identifies a communicative action performed in reply to a request message or independently. In addition, there are message types prefixed by the functions propose, accept-proposal, and reject-proposal with obvious meanings.

The first communicative action event type in the interaction frame in Figure 2 between the agent type Customer and the internal agent type SalesDepartment of the CeramicsFactory represents a request by the Customer to provide it with the product set that is identified by the product code (?String) and quantity required (?Integer). The following three communicative action event types of the interaction frame being studied in Figure 2 comply with the isBenevolentTo relationship between the agent types SalesDepartment and Customer in the organization model of Figure 1. They model a proposal by the SalesDepartment to provide the Customer with the product set according to the production order created by the SalesDepartment and its acceptance or rejection by the Customer. The instance of the production order, which includes a specific due time, is described by the data element ?ProductionOrder of the corresponding communicative action events. If the proposal is accepted, the SalesDepartment commits on behalf of the CeramicsFactory towards the Customer to provide it by the due time with the product set defined by the production

Figure 2. The interaction model of the CeramicsFactory



order. A commitment/claim of this type is satisfied by an action event of the type provideProductSet(?ProductionOrder), which is coupled with the corresponding commitment/claim type. After the product set has been produced, the SalesDepartment first informs the Customer about the completion and the Customer then issues to the CompletedProductionStore (an internal institutional agent of the SalesDepartment) a request to release the product set identified by the corresponding ProductionOrder. The CompletedProductionStore provides the Customer with the product set in question and also sends to the Customer the invoice (?Invoice). This is accompanied by creating for the SalesDepartment a claim against the Customer that it would pay for the product set according to the invoice by a certain date. The claim is satisfied by actual paying for the product set by the Customer.

The starting point for creating the interaction frame in Figure 2 between the internal agent types SalesDepartment and ProductionDepartment is the dependency relationship (providesResourceTo) between them, as shown in Figure 1. The ProductionDepartment thus provides the SalesDepartment with the resources needed for selling the products of the factory. The first communicative action event type of the interaction frame models a request by the SalesDepartment to the ProductionDepartment to schedule the production order described by the data element

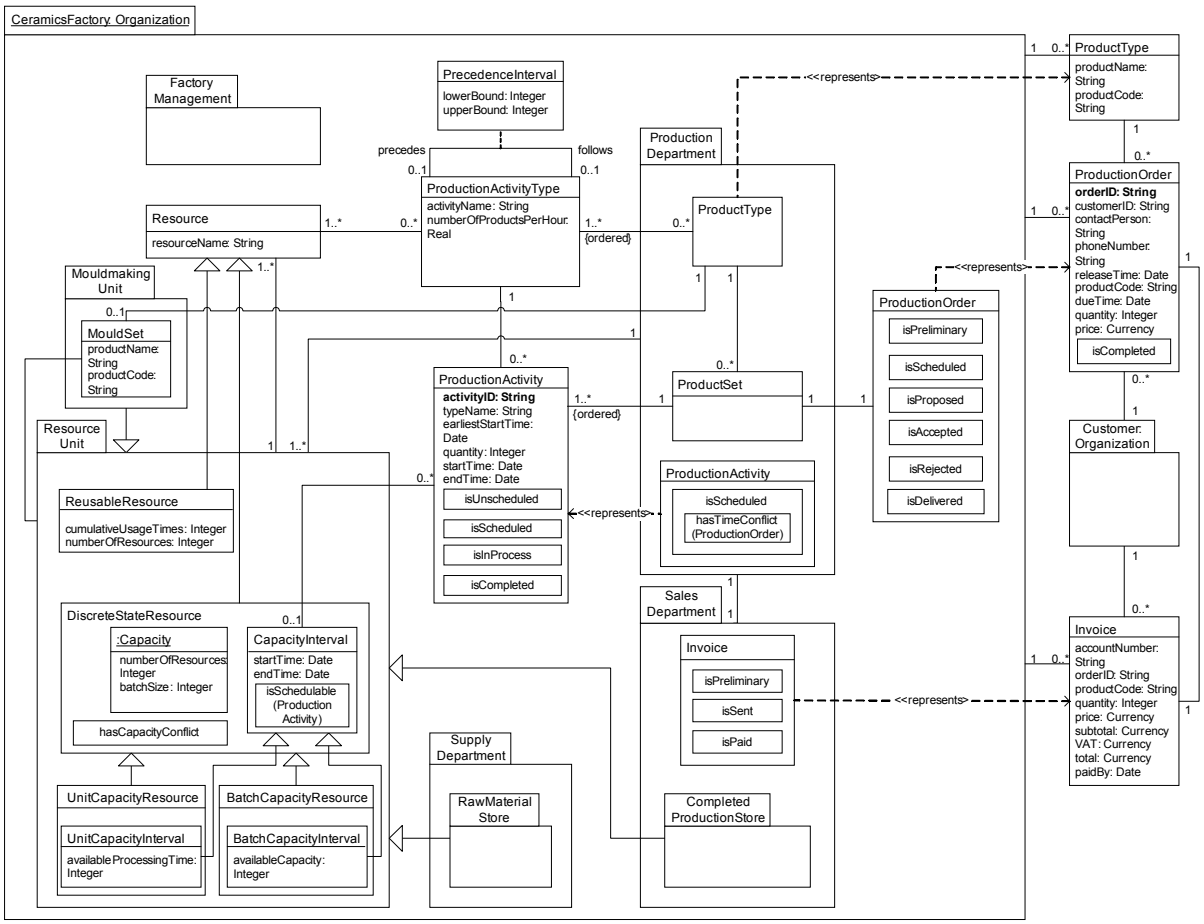
?ProductionOrder of the action event. Since neither scheduling a production order nor producing a product set according to it can be immediately perceived by the SalesDepartment, they are modeled as making true the respective status predicates isScheduled and isCompleted of the corresponding instance of ProductionOrder. After the ProductionDepartment has returned the scheduled production order to the SalesDepartment, it receives from the SalesDepartment a request to either have the production order completed or to delete it. In the first case, a *stit*-commitment/claim of the type (achieve(isCompleted(?ProductionOrder) ?Date) is formed between the ProductionDepartment and SalesDepartment. The satisfaction of this commitment/claim is expressed by the corresponding achieve-construct type.

The interaction frame between the agent types ProductionDepartment and ResourceUnit in Figure 2 is largely determined by the isSubordinateTo relationship between their internal agent types Worker and ProductionManager which is reflected by their comprising organization units. This means that a ResourceUnit schedules and performs a production activity as requested by the ProductionDepartment and reports to the latter. The first communicative action event type between the agent types ProductionDepartment and ResourceUnit models a request by the ProductionDepartment to schedule the production activity that is described by the data element ?ProductionActivity of the action event. In addition to initial scheduling of a production activity, a request of this type is also sent if a time conflict in the schedule is detected within the ProductionDepartment.[3] The second message type between the same agent types models the confirmation of the scheduling by the ResourceUnit. The third message type, which represents a request to delete the scheduled production activity described by ?ProductionActivity, is used only if the production order including the production activity to be deleted has been rejected by the Customer. Since the completion of a production activity cannot be directly perceived, it is modeled through the achieve-construct type achieve(isCompleted(?ProductionActivity)) between the agent types ResourceUnit and ProductionDepartment. The achieve-construct type is coupled with the corresponding *stit*-commitment/claim type because the completion of a production activity is preceded by the formation of the corresponding commitment/claim. Communicative action event types inform(isScheduled(?ProductionActivity)), inform(isInProcess (?ProductionActivity)), and inform(isCompleted (?ProductionActivity)) serve to inform the ProductionDepartment about the status changes of the production activity described by ?ProductionActivity. A message of the type inform(isCompleted (?ProductionActivity)) may trigger rescheduling of a production order by "pushing" all the production activities included by it forward or backward in time when the production activity is completed later or earlier than scheduled, respectively.

The Information Aspect

Representing the information aspect of the RAP/AOR viewpoint modeling framework for the focus organization(s) can be regarded as creating ontology. The latter provides a common framework of knowledge for the agents of the organization(s) and external agents connected to the organization(s). The ontology—information model—of the ceramics factory has been developed according to the principles of the OZONE scheduling ontology laid out by Smith and Becker (1997). The information model of the ceramics factory is represented as an agent diagram in Figure 3. In addition to de-

Figure 3. The information model of the ceramics factory



scribing agent types, it describes object types of the ceramics factory, as well as their relationships to agent types and with each other.

In the information model of the ceramics factory depicted in Figure 3, the concept DEMAND of the scheduling ontology by Smith and Becker (1997) is represented by the object type ProductionOrder. It is shared between the agent CeramicsFactory and the agent type Customer. A ProductionOrder is characterized by a number of attributes, the most important ones being releaseTime, dueTime, productCode, and quantity, and by the status predicate isCompleted. The attributes releaseTime and dueTime are respectively the earliest and latest time when the production activities for producing the product set specified by the ProductionOrder can start and end, respectively. The attributes productCode and quantity respectively specify the type and number of the products in the product set requested. The internal representation of the object type ProductionOrder within the agent CeramicsFactory satisfies one of the following status predicates: isPreliminary, isScheduled, isProposed, isAccepted, isRejected, or isDelivered.

There is a shared object type Invoice connected to ProductionOrder, Customer, and CeramicsFactory. It includes a number of attributes like orderID, productCode, quantity, price, subtotal, VAT, and total. In addition, its internal representation within the SalesDepartment has the status predicates isPreliminary, isSent, and isPaid.

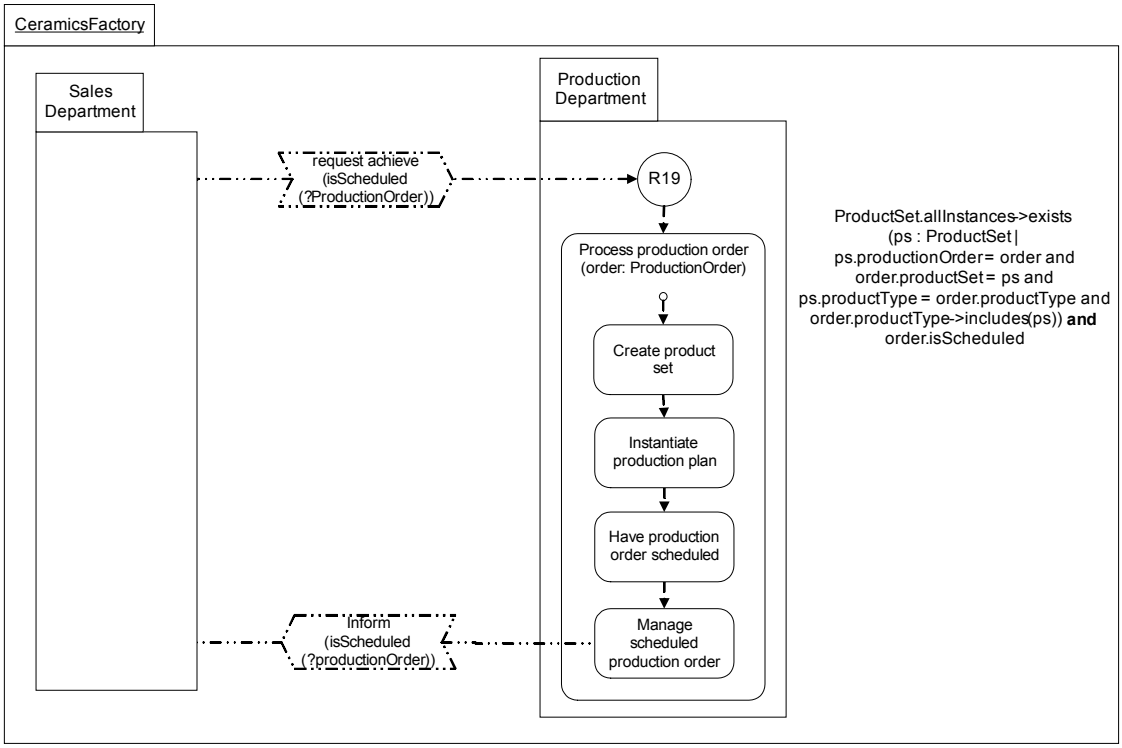
In Figure 3, the type of the products requested is modeled by the association between the shared object types ProductionOrder and ProductType. An instance of ProductType is identified by its attributes productName (for example, "coffee cup Kirke") and productCode (for example, "22882"). The internal representation of the object type ProductType within the agent type ProductionDepartment differs from its base object type by a number of relationships to other object types. Among them, an ordered sequence of instances of ProductionActivityType associated with an instance of ProductType defines the product type in question. Specific sets of products to be produced to satisfy production orders are represented as instances of the object type ProductSet. The latter corresponds to the concept PRODUCT of the scheduling ontology by Smith and Becker (1997). Each ProductSet references an ordered sequence of instances of ProductionActivity which define the set of processing steps required to produce the product set. There are associations of the type PrecedenceInterval between instances of ProductionActivityType. Each association specifies the lower bound and upper bound of the temporal separation between production activities of two consecutive types. In conformance with Smith (1989), the associations of the type PrecedenceInterval are intended to provide a basis for describing generic manufacturing processes, defining sets of possible production activity sequences.

Analogously to the OZONE scheduling ontology proposed by Smith and Becker (1997), the information model of the ceramics factory adopts an activity-centered modeling perspective. In the center of the ontology represented in Figure 3 is thus the object type ProductionActivity, corresponding to the concept ACTIVITY of the scheduling ontology by Smith and Becker (1997). An object of the type ProductionActivity can have the status isUnscheduled, isScheduled, isInProcess, or isCompleted. An instance of ProductionActivity is characterized by the following attributes: activityID, typeName, earliestStartTime, quantity, startTime, and endTime. The identifier attribute activityID contains the identifier of the production activity which is automatically assigned to it upon creation of the corresponding object. The attribute typeName repeats the value of the attribute activityName of the ProductionActivityType associated with the activity. The action of scheduling the corresponding production activity results in determining values for the attributes startTime and endTime. The attribute earliestStartTime indicates the earliest time at which the given production activity can be started, considering the endTime of the previous activity scheduled or the releaseTime of the ProductionOrder in case of its first production activity. Each instance of ProductionActivity belongs to some ProductionActivityType. This is represented by the corresponding many-to-one relationship in Figure 3. An instance of ProductionActivityType is characterized by the name of the activity type (activityName) and the average speed of performing an activity of the corresponding type (numberOfProductsPerHour). The latter includes the time required for setting up the resources before a production activity of the given type can actually start. The object type ProductionActivity has a specific internal representation within the agent type ProductionDepartment. It refines the status predicate isScheduled by the internal intentional predicate hasTimeConflict(ProductionOrder),

because a time conflict between scheduled activities is detected within the ProductionDepartment. Taveter (2004) has defined the intentional predicate hasTimeConflict(ProductionOrder) as an Object Constraint Language (OCL) (OMG, 2003a, 2003b) operation attached to the object type ProductionActivity.

Another important notion in a scheduling ontology is that of a resource. In the information model depicted in Figure 3, it is represented as the object type Resource which corresponds to the concept RESOURCE of the scheduling ontology by Smith and Becker (1997). Each institutional agent of the type ResourceUnit has knowledge about objects of at least one of the proprietary subtypes ReusableResource and DiscreteStateResource. A resource described by an instance of ReusableResource, like a set of ceramic moulds, is a resource whose capacity becomes available for reuse after the production activity to which it has been allocated finishes. As Figure 3 reflects, an instance of ReusableResource is characterized by two attributes: the cumulativeUsageTimes and the numberOfResources. The cumulativeUsageTimes specifies the total amount of resource uses permitted (i.e., the number of times that the use of a mould set is permitted for moulding or casting. A resource described by an instance of DiscreteStateResource, like a worker or a machine or their combination, is a resource whose availability is a function of some discrete set of possible state values (i.e., *idle* and *busy*). Each instance of DiscreteStateResource consists of the internal nameless object instance :Capacity and instances of the internal object type CapacityInterval. The :Capacity object specifies the numberOfResources and batchSize. The latter is the number of products that the resource can process simultaneously. The capacity of a resource is represented as an ordered sequence of instances of CapacityInterval, describing, for example, work shifts. Each interval indicates the production activities that are anticipated to be consuming capacity within its temporal scope and the capacity that remains available (Smith, 1989). For each CapacityInterval, the startTime and endTime of the interval are specified. The specializations of CapacityInterval, not shown in Figure 3, are WorkMonth, WorkWeek, and WorkShift. Their implementations form parts of the simulation environment of the ceramics factory that has been described by Taveter and Wagner (2006). Successful scheduling results in attaching a CapacityInterval to one or more instances of ProductionActivity. In order to determine whether a CapacityInterval can be allocated to the given ProductionActivity, the object type CapacityInterval possesses the intentional predicate isSchedulable(ProductionActivity). The object type CapacityInterval has the subtypes UnitCapacityInterval and BatchCapacityInterval which are included by the respective two subtypes of DiscreteStateResource – UnitCapacityResource and BatchCapacityResource. A resource described by an instance of UnitCapacityResource, like a worker, can process only one product at a time—that is, its batchSize is 1, whereas a resource described by an instance of BatchCapacityResource, like a kiln, can process simultaneously up to batchSize products. While the available capacity of a capacity interval described by an instance of UnitCapacityInterval is characterized by the attribute availableProcessingTime (i.e., per work shift), the available capacity of a capacity interval described by an instance of BatchCapacityInterval is characterized by the number of products (availableCapacity) that the resource is capable of processing at a time. The intentional predicate isSchedulable(ProductionActivity) for the object types UnitCapacityInterval and

Figure 4. An incomplete behavior model of the manufacturing process type "Process production order"



BatchCapacityInterval is defined by Taveter (2004) using OCL.

The Behavior Aspect

The behavior aspect of the RAP/AOR viewpoint modeling framework addresses the modeling of an agent's functionality (what functions the agent has to perform), as well as of the agent's behavior (when, how, and under what conditions work has to be done).

The behavior aspect comprises motivation modeling. Goals and sub-goals can be captured by AOR activity diagrams. In the AOR activity diagram of Figure 4, an activity of the type "Process production order" is started by reaction rule R19 in response to receiving a message containing a request for scheduling. As shown in Figure 4, an activity of the type "Process production order" consists of sequential subactivities of the types "Create product set," "Instantiate production plan," "Have production order scheduled," and "Manage scheduled production order."

For each activity type represented in Figure 4 can be defined the goal that its instances try to achieve. The figure shows the goal that has been defined for the outermost activity type "Process production order" by means of OCL in terms of its input parameter order. Input parameters defined for activity types represent the dataflow through the activities.

Subsequently, initial AOR activity diagrams are elaborated on by introducing into them *behavioral constructs* by means of reaction rules. Taveter (2004) has shown that AORML

extended by activity modeling allows the representation of 16 out of 19 behavioral workflow patterns as defined in the workflow benchmark proposal of van der Aalst, ter Hofstede, Kiepuszewski, and Barros (2003). In the behavior model represented in the external AOR diagram in Figure 5, the activity type "Have production order scheduled" modeled in Figure 4 has been complemented by the behavioral constructs of the types "For-each loop" and "Sequence." In addition, the agent type ResourceUnit and the elementary epistemic, physical, and communicative actions that are included by activities of the types "Request scheduling" and "Register scheduling" have been specified. For scalability reasons, only one of the activity types represented in Figure 4 is refined in Figure 5.

The behavioral construct of the type "For-each loop" represented in Figure 5 specifies that upon the start of an activity of the type "Have production order scheduled," its sub-activity of the type "Have production activity scheduled" is performed for each instance of the object type ProductionActivity for which the pre-condition productSet.productionOrder = order and isNextActivity(order) represented in OCL evaluates to true. The pre-condition limits the set of production activities for which the sub-activity is performed to the ones associated with the instance of ProductionOrder that is identified by the value of the input parameter order. In addition, the expression isNextActivity(order) ensures the scheduling of production activities in the correct order. When all sub-activities of the type "Have production activity scheduled" have ended, the enclosing activity of the type "Have production order scheduled" also ends.

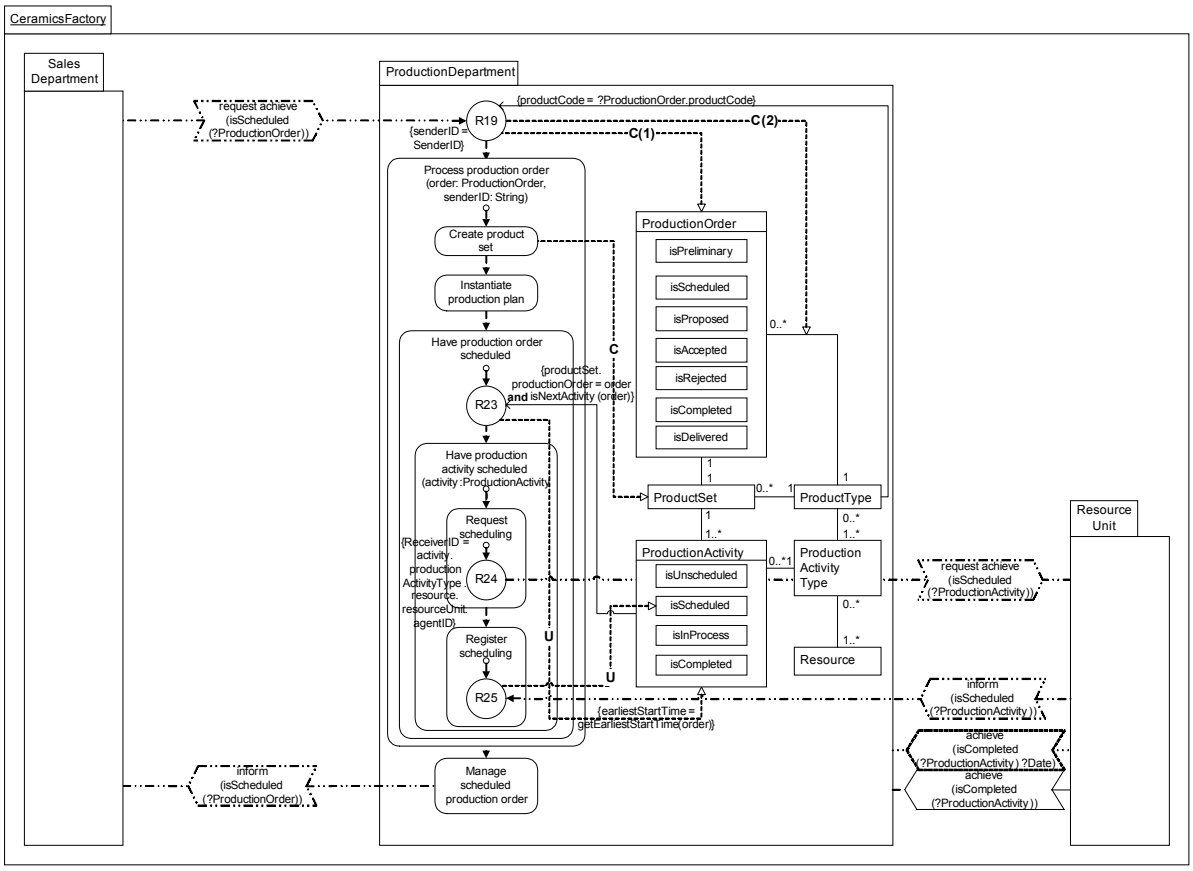
The sub-activity "Request scheduling" shown in Figure 5 sends to the corresponding agent ResourceUnit a request to schedule the ProductionActivity, while the sub-activity "Register scheduling" waits for and registers the scheduling of the production activity by the ResourceUnit. When a time conflict is detected within the ProductionDepartment, a production order is rescheduled in a similar way which results in "pushing" all the production activities included by it forward in time.

SIMULATION

Taveter (2004) demonstrated that AORML extended by activities is the first Agent-Oriented Modeling Language where fully as well as partially specified behavior models by external AOR diagrams can be executed. For example, in our case study, conceptual AOR behavior models represented in Figures 4 and 5 can be executed, even though neither of them is completely specified. This facilitates iterative "modeling by simulation" where models can be executed at any stage of behavior modeling. Taveter and Wagner (2006) describe a simulation environment created for the ceramics factory by transforming external AOR diagrams into the programming constructs of the JADE (Java Agent Development Environment, http://jade.cselt.it/) agent platform.

For performing simulation experiments, an on-site survey was first performed at the Tallinn Ceramics Factory. In the survey, the average speeds of performing production activities of different types, as well as the minimal precedence intervals required between the activities, were found out and recorded. The results of the survey, along with the batch size for each resource type and the number of resources, are presented in Table 1. The values in Table 1 were assigned to the corresponding object fields of the simulation environment. In particular, the average speed of performing a production activity of a given type is assigned to the numberOfProductsPerHour attribute of the cor-

Figure 5. An external AOR diagram representing the manufacturing process type "Process production order"



responding ProductionActivityType object, and the minimal precedence interval required between two production activities is assigned to the lowerBound attribute of the corresponding PrecedenceInterval object. Please notice that the numberOfProductsPerHour attribute is not needed for instances of BatchCapacityResource that describe kilns and their secondary resources[4]—burners—because kilns operate at regular intervals.

The simulation environment lends itself to both predictive and reactive scheduling. Table 2 represents a production schedule for producing a product set of the type "Moulded ceramic product 22882." The production schedule reflects that kilns have a specific work cycle because of the requirements for cleanliness and safety—they are in operation on Mondays, Wednesdays, and Fridays. Table 2 shows the start and end times of production activities before and after detecting two time conflicts within the ProductionDepartment. As the table reflects, the scheduled execution times of the "Initial elaboration" and "Painting" production activities have been "pushed" forward in time because their preceding production activities have taken more time than had been initially scheduled. In a similar manner, reactions to the

Table 1. The parameter values for production activities to be performed for producing a product set of the type "Moulded ceramic product 22882"

Production Activity	**Types of Resource(s) Required**	**Batch Size**	**Number of Resources**	**Average Speed (number of products per hour), if applicable**	**Minimal Precedence Interval (hours) between this and the next activity, if applicable**
Moulding	Moulder	1	1	44	24
Initial elaboration	Elaborator	1	16	23	72
Engobe painting	Engobe painter	1	1	37.5	-
Initial combustion	Kiln for initial combustion	1000	1	-	24
	Burner	1	2	-	-
Elaboration	Elaborator	1	16	23	-
Painting	Painter	1	8	7.5	-
Glazing	Glazier	1	1	37.5	-
Post-glazing combustion	Kiln for post-glazing combustion	1000	4	-	24
	Burner	1	2	-	-
Decoration	Painter	1	8	12.5	-
Post-decoration combustion	Kiln for post-glazing combustion	1000	4	-	24
	Burner	1	2	-	-
Packaging	Packer	1	2	125	-

Table 2. A schedule for the production process of the "Moulded ceramic product 22882" product set before and after the right shift

Production Activity	**Start Time**	**End Time**	**New Start Time**	**New End Time**
Moulding	Mon Aug 29 08:00	Mon Aug 29 12:32	Mon Aug 29 08:00	Mon Aug 29 12:53
Initial elaboration	Tue Aug 30 12:33	Tue Aug 30 13:05	Tue Aug 30 12:54	Tue Aug 30 13:26
Engobe painting	Mon Sep 05 08:00	Mon Sep 05 13:20	Mon Sep 05 08:00	Mon Sep 05 13:42
Initial combustion	Wed Sep 07 08:00	Wed Sep 07 16:00	Wed Sep 07 08:00	Wed Sep 07 16:00
Elaboration	Fri Sep 09 08:00	Fri Sep 09 08:32	Fri Sep 09 08:00	Fri Sep 09 08:37
Painting	Fri Sep 09 08:33	Fri Sep 09 11:53	Fri Sep 09 08:38	Fri Sep 09 11:58
Glazing	Mon Sep 12 08:00	Mon Sep 12 13:20	Mon Sep 12 08:00	Mon Sep 12 13:20
Post-glazing combustion	Wed Sep 14 08:00	Wed Sep 14 16:00	Wed Sep 14 08:00	Wed Sep 14 16:00
Decoration	Fri Sep 16 08:00	Fri Sep 16 10:00	Fri Sep 16 08:00	Fri Sep 16 10:00
Post-decoration combustion	Mon Sep 19 08:00	Mon Sep 19 09:30	Mon Sep 19 08:00	Mon Sep 19 09:30
Packaging	Tue Sep 20 09:31	Tue Sep 20 10:19	Tue Sep 20 09:31	Tue Sep 20 10:19

changes in the number of available resources could be simulated. Note that because of the requirement to warrant a homogeneous quality of ceramic products in a product set, a production activity once started on a product set should be finished on the same day.

CONCLUSION

In this chapter, the application of the RAP/AOR methodology proposed by Taveter and Wagner (2005, 2006) has been described by using a full-length and real-life case study of a ceramics factory. The models that were created demonstrated the ability to overcome the *model multiplicity problem* (Peleg & Dori, 2000) which states that to understand the system being studied and the way it operates and changes over time, the reader must concurrently refer to various models. Even more importantly, the chapter also explained how the models obtained were used for simulation of the ceramics factory and its manufacturing processes. The RAP/AOR methodology can be applied more generally for modeling and simulation of manufacturing processes of enterprises of different industries. Simulations of this kind enable the creation and analysis of production schedules, and try out the influences of changes in the capacities of resources. Moreover, agent-oriented modeling and simulation by means of the RAP/AOR methodology also serves as the first stage of creating a distributed automation system that is able to adapt to disturbances concerning orders and resources. Such systems also facilitate the integration of manufacturing processes of different enterprises.

ACKNOWLEDGMENT

The author would like to extend his gratitude to Reet Hääl, the former managing director of the Tallinn Ceramics Factory, for enabling us to use the factory as a case study.

REFERENCES

Austin, J. (1962). *How to do thing with words.* Oxford, UK: Clarenson Press.

OMG. (2003a, March). *Unified Modeling Language specification, version 1.5.* Retrieved October 11, 2005, from http://www.omg.org/cgi-bin/doc?formal/03-03-01/

OMG. (2003b, August). *Unified Modeling Language: Superstructure, version 2.0.* Retrieved October 11, 2005, from http://www.omg.org/cgi-bin/doc?ptc/2003-08-02/

Ow, S. P., Smith, S. F., & Howie, R. A. (1998, May 3-5). CSS: A cooperative scheduling system. In M. D. Oliff (Ed.), *Proceedings of the 2nd International Conference on Expert Systems and the Leading Edge in Production Planning and Control*, Charleston, SC (pp. 43-56). New York: North-Holland.

Peleg, M., & Dori, D. (2000). The model multiplicity problem: Experimenting with real-time specification methods. *IEEE Transactions on Software Engineering*, *26*(8), 724-759.

Smith, S. (1989). *The OPIS framework for modeling manufacturing systems*. Technical *report* CMU-RI-TR-89-30, Robotics Institute, Carnegie Mellon University, USA.

Smith, S., Ow, P. S., Muscettola, N., Potvin, J. Y., & Matthys, D. (1990, July 15-18). OPIS: An opportunistic factory scheduling system. *Proceedings of the 3rd International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems* (IEA/AIE'90) (Vol. 1, pp. 268-274), Charleston, SC.

Smith, S. (1995). Reactive scheduling systems. In D. E. Brown & W. T Scherer (Eds.), *Intelligent scheduling systems* (pp. 155-192). Boston: Kluwer.

Smith, S. F., & Becker, M. A. (1997, March). An ontology for constructing scheduling systems. *Working Notes of the 1997 AAAI Spring Symposium on Ontological Engineering* (pp. 120-129), Stanford, CA.

Tamm, B. G., Puusepp, R., & Tavast, R. (1987). *Analiz i modelirovanije proizvodstvennyh sistem* (*Analysis and modeling of production systems*). Moscow: Finansy i Statistika.

Taveter, K. (2004). *A multi-perspective methodology for agent-oriented business modeling and simulation.* PhD thesis, Tallinn University of Technology, Estonia.

Taveter, K., & Wagner, G. (2005). Towards radical agent-oriented software engineering processes based on AOR modeling. In B. Henderson-Sellers & P. Giorgini (Eds.), *Agent-oriented methodologies* (pp. 277-316). Hershey, PA: Idea Group Publishing.

Taveter, K., & Wagner, G. (2006). Agent-oriented modeling and simulation of distributed manufacturing. In J.-P. Rennard (Ed.), *Handbook of research on nature-inspired computing for economy and management.* Hershey, PA: Idea Group Reference.

van der Aalst, W. M. P., ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. *Distributed and Parallel Databases, 14*(3), 5-51.

Wagner, G., & Tulba, F. (2003). Agent-oriented modeling and agent-based simulation. In P. Giorgini & B. Henderson-Sellers (Eds.), *Conceptual modeling for novel application domains* (LNCS 2814, pp. 205-216). Berlin: Springer-Verlag.

Zeng, D. D., & Sycara, K. (1999). Dynamic supply chain structuring for electronic commerce among agents. In M. Klusch (Ed.), *Intelligent information agents: Agent-based information discovery and management on the Internet* (pp. 232-249). Berlin: Springer-Verlag.

ENDNOTES

1. Henceforth, the terms "actor" and "agent" are used as synonyms.
2. These functions are actually *speech acts* (Austin, 1962).
3. This situation is represented by the same interaction frame because interaction frame diagrams do not model the order in which action events of specified types occur.
4. The allocation of a secondary resource accompanies the allocation of a resource of another type.

Chapter XXXVII
Building Distribution Networks Using Cooperating Agents

Neil Urquhart
Napier University, Scotland

ABSTRACT

This chapter examines the use of emergent computing to optimize solutions to logistics problems. The chapter initially explores the use of agents and evolutionary algorithms to optimise postal distribution networks. The structure of the agent community and the means of interaction between agents is based on social interactions previously used to solve these problems. The techniques developed are then adapted for use in a dynamic environment planning the despatch of goods from a supermarket. These problems are based on real-world data in terms of geography and constraints. The author hopes that this chapter will inform researchers as to the suitability of emergent computing in real-world scenarios and the abilities of agent-based systems to mimic social systems.

INTRODUCTION

The design and optimisation of urban distribution networks used to facilitate the delivery of goods to households is a non-trivial problem. Such problems exist within an urban environment where routes must take into account a complex real-world road network. These problems are not solved by building single routes in isolation, but require networks consisting of routes each of which must satisfy local constraints pertaining to that route. The network should also satisfy global constraints that may conflict with local constraints. The principle tools to be used are: agents coordinated by economic and social metaphors, and evolutionary algorithms. Unlike much previous work in this area, which has utilised datasets based on Euclidean distances between arbitrary points, the problems in this chapter are based on real-world street topologies.

This chapter initially reviews existing work in the areas of scheduling and routing, with particular emphasis on the use of emergent

computing (EC). It subsequently describes the use of EC methods, notably evolutionary algorithms and agents to solve a real-world urban routing problem, specifically the distribution of post within urban areas in the City of Edinburgh.

The techniques utilised for the postal problem are then developed for use in planning the despatch of goods from supermarkets to individual houses, this being a dynamic problem with fluctuating demand. Finally, conclusions are drawn as to the effectiveness of EC-based techniques on the problems discussed.

REVIEW OF EXISTING WORK

Multi-vehicle routing is a combinatorial problem that normally involves the division of work (deliveries) between vehicles, as well as the optimisation of each vehicle's route. Traditionally such problems have used one of two representations: they either utilise a permutation-based approach or represent the problem as a network of arcs. The former representation is traditionally used to solve the travelling salesman problem (TSP); in this case the size of the search space is n!/2n. Alternatively the graph-based approach may be utilised, the graph being based on real-world street topology, with deliveries being achieved through the traversal of a set of arcs.

Techniques used to solve routing problems may be subdivided into three main categories: exhaustive algorithms, heuristics, and meta-heuristics. Exhaustive algorithms seek to evaluate the entire search space; they have the property of always producing an optimal solution, but given the size of the search spaces, the time taken can be prohibitive. Such methods may be hybridised with techniques such as linear programming (Ladányi, Ralphs, & Trotter, 2001). The approach taken used branch and cut techniques combined with linear programming. The time cost was overcome by distributing the problem over multiple CPUs.

A number of researchers have opted to solve vehicle routing problems with the use of heuristics. Many of the simpler heuristics such as 2-opt (Croes, 1958) are applied iteratively to solve the problem in small stages. Heuristics such as the Lin-Kernighan heuristic (Lin & Kernighan, 1973) have been used with success to solve the TSP (Cook, Cunningham, Pulleyblank, & Schrijver, 1998).

Meta-heuristic techniques have been used to solve the vehicle routing problem (VRP); these include ant-colony optimisation (Gambardella & Dorigo, 2000), simulated annealing (Czech & Czarnas, 2002), and evolutionary algorithms. Considerable research into solving the travelling salesman problem using EAs has been undertaken (Freisleben & Merz, 1996); Tamaki, Kita, Shimizu, Maekawa, & Nishikawa, 1994; Homaifar, Guan, & Liepins, 1993; Whitley, Starkweather, & Shaner, 1991; Mathias & Whitley, 1992). Investigation into the use of EAs to solve the vehicle routing problem has also been undertaken, notably in Thangiah (1999), Blanton and Wainright (1993), Thangiah, Vinayaamoorthy, and Gubbi (1993), and Homaifar et al. (1993). The integration of the Lin-Kernighan heuristic and an evolutionary algorithm has been explored in Freisleben and Merz (1996), and Baraglia, Hidalgo, and Perego (2002). The problem of routing garbage collection using EAs is explored in Bousonvile (2001); the authors use an arc-routing EA to construct routes for a real-world garbage collection problem. Garbage collection is essentially an arc-routing-based problem, each street requiring collection being represented by an arc on the graph. Work on the construction of postal delivery routes has been discussed in Urquhart, Ross, Paechter, and Chisholm (2002a, 2002b) and Urquhart, Paechter, and Chisholm (2001). The earlier work concentrated on the

use of a representation known as street-based routing (SBR) designed for use within an evolutionary algorithm. Because SBR groups deliveries geographically, it also forms a convenient unit of exchange between routes when optimising a delivery network. Agent-based approaches to multi-vehicle routing problems have also been examined by Hoen and La Poutré (2003), who explore an approach whereby agents representing different companies compete through auctions to win deliveries. Agents are not committed to deliveries once they have been won; they may de-commit if more profitable work is available. The authors of Bachem, Hochstättler, and Malich (1996) attempt to solve the VRP using a simulated trading heuristic. Using their heuristic it costs an agent credit to remove a customer, but an agent gains credit for accepting a customer.

THE CONSTRUCTION OF POSTAL DISTRIBUTION NETWORKS

Problem Description

Christmas cards are delivered directly to households within the City of Edinburgh by charities that organise the collections and deliveries to raise funds. The city is sub-divided into areas, each of which requires a network of delivery rounds; each round should attempt to satisfy the following constraints:

1. The maximum distance for each round must be less than a set limit, typically 3km.
2. The maximum number of delivery points (households) per route should not exceed a predefined limit, typically 250.
3. The number of delivery rounds should be minimised.

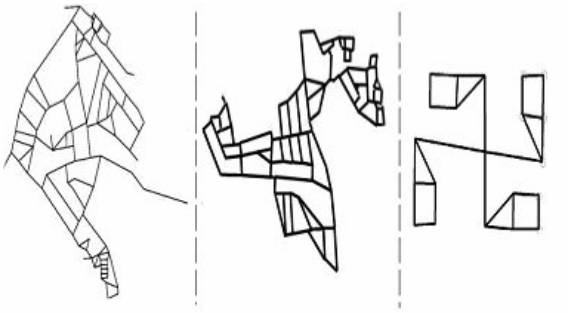
Note that constraints 1 and 2 may be deemed hard constraints and the rest soft constraints.

The test areas used within this study are the Moredun and Fairmilehead housing areas located in the south of the city. The Moredun area contains 1,770 households arranged within 165 street sections. This area combines a mixture of tower blocks (with over 100 dwellings per building) and semi-detached villas. The Fairmilehead area encompasses 1,592 delivery points over 153 street sections. The topology and distance values were derived from data supplied by the British Ordnance Survey under the Edina/JISC agreement. The Windmill dataset (see below) was an artificially created data set consisting of 202 delivery sections spread over 28 street sections. The topology of the street sections may be seen in Figure 1.

An informal study of the problem revealed that a social system had evolved when constructing manual solutions; a coordinator initially divides up work and allocates it to a number of operatives for delivery, each operative having one delivery round. The operatives then informally exchange work so as to optimise their individual workloads. The solution proposed seeks to model this existing social system by using a coordinator agent that allocates work to a community of routing agents who optimise individual routes. The routing agents are allowed to carry out transactions that redistribute work between them.

Each agent represents the interests of one human delivery operative (adhering to maximum length and maximum capacity constraints), whilst contributing towards the community's goal of minimising the overall cost of the network in terms of distance and number of operatives used. Each agent has an objective to undertake deliveries up to the maximum allowed, whilst minimising the distance covered (and cost incurred).

Figure 1. Street topology of the datasets used in this chapter (left to right): the Moredun dataset, the Fairmilehead dataset, and the Windmill dataset



Building Networks Using a Community of Agents

Agents must be able to redistribute work (deliveries); this will be facilitated by means of a first-price sealed bid auction administered by the coordinator agent. In order to participate within the community, each agent requires a number of abilities:

- to compare and evaluate items of work (deliveries),
- to communicate with other agents, and
- to optimise their route.

These requirements are met as follows:

- **Item 1:** Each agent will use a valuation function to allocate a value to any item of work, owned or not. This value represents the degree to which that item of work would contribute to the agent's objectives. By generating such values an agent can take part in an auction or decide which items of work are of least value to it and should be given to other agents.
- **Item 2:** Agents can communicate via the coordinator who is responsible for administering auctions.
- **Item 3:** In this case the algorithm used by the agents is SBR-EA, an EA-based routing algorithm (see the following).

The chosen representation is a constant length permutation of street section identifiers. Both sides of each two-sided section have a gene within the chromosome; for instance, if there were just two street sections and both sides of each section required a delivery, there would be four genes.

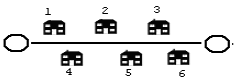
Street-Based Routing Using an Evolutionary Algorithm

SBR exploits the geographical groupings of houses (delivery points) that exist within streets. Each route is represented by a permutation of street sections. Each street being a section of roadway between two junctions, each section can contain 0 or more delivery points. To construct a route the street sections are supplied to a route builder that adds them in sequence to the route.

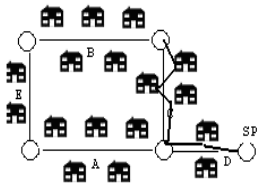
Because each street section may contain a number of delivery points, it is necessary to determine the order in which these points should be visited when building a route. The order of deliveries is determined by applying one of three delivery patterns. The choice of delivery pattern for each section is determined during the evaluation process and is not included in the chromosome structure. The available patterns are:

- deliver each side separately,
- deliver to both sides in one operation whether by crossing backwards and forwards repeatedly over the road, and
- deliver to one side in its entirety and then the other side.

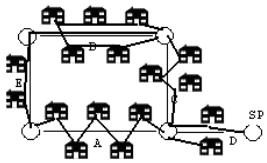
Figure 2. A simple SBR example based on the chromosome DCCBBEAAD (please note that as street E is a single-sided section, it only appears once)



Delivery Pattern 1- deliver to each side separately (e.g. 1-3 then 4-6) by traversing the street section twice within the route. Delivery Pattern 2- deliver to both sides crossing the street as required (e.g. 1-4-2-5-3-6 in the above example). Delivery pattern 3- Deliver to one side and then the other (e.g. 1-2-3-6-5-4 in the above example).



The first gene in our example is D followed by CC. Street D is visited (pattern 1 is applied) then street C (pattern 2 is applied).



The route builder interprets the rest of the chromosome. Patterns are applied as follows:
Street D pattern 1
Street C pattern 2
Street B pattern 3
Street E pattern 1
Street A pattern 2
Street D pattern 1

The delivery pattern used depends on where the section is to be placed within the route. By altering the delivery patterns, the route may be optimised for a delivery person travelling on foot or a delivery vehicle. This routing algorithm is known as SBR-EA; a full description may be found in Urquhart et al. (2001). See Figure 2 for an example of how the SBR route builder decodes chromosomes.

The recombination operator used within SBR-EA creates new genotypes based on two parents, these being selected using a tournament of size 2. A string of genes from parent 1 which exist between two random points are copied to the child, allowing the preservation of a contiguous section of route. Subsequently those genes in parent 2 which have not already been copied are each copied to the first available space in the child chromosome. This scheme is based on the OX recombination operator (Davis, 1985). A mutation is defined as the selection of a single gene and its movement to another random point within the chromosome. This will produce a change within the phenotype route. All of the child genotypes created at a particular generation are subject to the possibility of mutation as determined by the mutation rate. Parameters used with SBR-EA are given in Table 1.

Constructing Static Distribution Networks Using Agents

The coordinator undertakes an initial allocation of work (street sections) to agents using a grouping EA that produces groups of adjacent streets that meet the maximum deliveries constraint. Each group is allocated to an agent. The initial number of agents is determined by dividing the total number of deliveries by the maximum allowed per agent. The agents then commence optimising their routes using their copy of SBR-EA.

When the agents have all executed their copy of SBR-EA, the coordinator requests that the agent with the greatest violation of the hard

Table 1. The EA parameters used for the SBR-EA

Parameter	Value
Population Size	175
No of children/generation	popSize*0.5
Tournament size	2
Mutation rate	1/chromo_len
Recombination rate	0.5
No of evaluations per run	500000

Figure 3. The bidding logic used by agents

```
function bid(Street work){
 bid = avgDist from work to existing work
 if adding street means total delivery > MAXDELS)
then
   bid = bid + (totalDeliveries *2)
 if (previous owner of work = thisAgent) then
   decline to bid
 if (agent already > MAXLEN) then
   decline to bid
 if (current dist + dist to new work > MAXLEN)  then
   decline to bid
 }
```

constraints returns a street section for reallocation. The section is then auctioned; the agent submitting the lowest bid has the section allocated to it.

The bidding logic used by agents is shown in Figure 3; each bid is based on the average distance from the street section under consideration to streets currently allocated to that agent. The closer the street section under consideration to the other work owned by the agent, the lower the bid value submitted. Penalties are added to the bid if the addition of the new street section causes the agent to break any hard constraints. The logic declines to bid if the agent estimates that the addition of the street section will violate the constraints or if the agent was the previous owner of the street section. In the event of all agents declining to bid for street section, this is taken as a consensus decision by the community to introduce a new agent. The new agent is created by and receives half of the workload of the agent with the largest workload in the community. The auction for the unallocated section is then held again with the enlarged community.

The system was tested with maximum round lengths of 2, 2.5, and 3km; the results obtained using the agents are shown in Table 2. Note that in the cases of the 2.5 km and 2km problems, the maximum distance in any of the routes is greater than 99% of the maximum length capacity. Efficient solutions should be those that produce rounds that use as much of the available delivery capacity as possible (see Table 2 for the percentages of delivery capacities used). This suggests that the system is capable of producing efficient solutions. As expected the number of delivery routes in the solution increases as the maximum length decreases.

Results obtained on the Moredun dataset are shown in Table 2, and results for the Fairmilehead dataset are shown in Figures 5a and 5b. In Figure 5a we see that the system finds it difficult to meet the maximum length constraint for every round when the maximum length is less than 3km. When the maximum length is increased beyond 4km, the system produces solutions in which the maximum round length constraint is no longer violated. Figure 5b shows the use of each agent's delivery and route length capacities over the same set of runs. These results are based on the average of 20 runs at each maximum length setting.

The bidding mechanism does not address the requirement to minimise extra sorts in its initial form. Figure 4 demonstrates how work is exchanged within the system. Note that in this case, two additional agents are added during the run (agents K and L). A vertical line represents each transaction; alphanumeric identifiers for the street section being exchanged are shown. Numbers at the start and end of the transaction represent the length of the agent's route after agent has used its EA to update its route. It may be noted that transactions fre-

quently happen in sequences that allow the transfer of all the street sections that make up a complete street from one agent to another. What is interesting about this phenomenon is that this behaviour was not programmed explicitly into the system.

Controlling the Number of Agents within the Community

The initial bidding mechanism described previously may be described as selfish as it does not submit a bid for any item of work that the agent

Figure 4. The run chart for the 2km problem: the vertical lines represent the agents, the horizontal lines represent transactions. Each transaction represents the result of one auction.

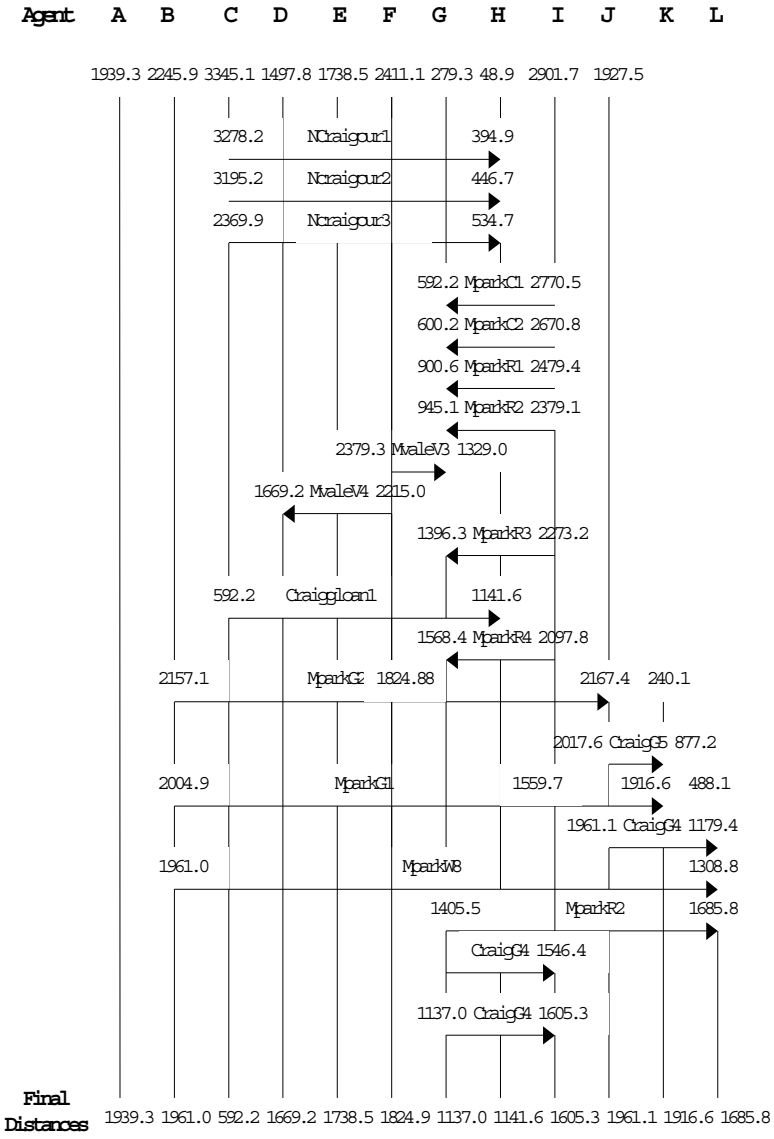


Table 2. A summary of the initial results obtained after undertaking 20 runs on the Moredun dataset

Max length constraint	Avg round length	% of max length constraint	Max round length	% of max length constraint	Avg No of round
3km	2166.9	72%	2996	99%	9.1
2.5km	1993.8	80%	2491.5	99%	10
2km	1589.5	80%	1995.7	99%	12.4

estimates will conflict with its own objectives. If all the agents decline to bid for a street section, the system creates a new routing agent even if the existing agents could cope with the workload simply by exchanging street sections. This can be demonstrated by creating a simple dataset, known as the Windmill dataset, which is illustrated in Figure 1. It was noted that the agent community described in the previous section consistently produced solutions that contained too many rounds. The reason for this was the constrained nature of this particular dataset.

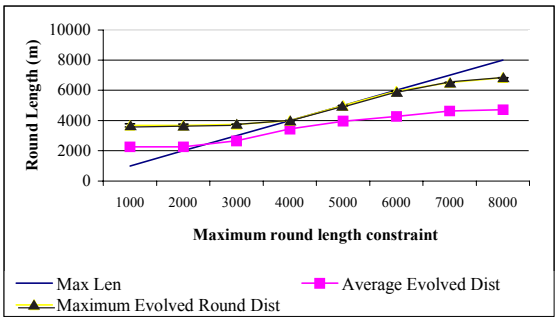
If the bidding logic is altered to always submit a bid, regardless of whether or not a hard constraint is broken, then the street sections are divided amongst the initial set of agents, although some agents may be violating hard constraints. Using this relaxed strategy the system stabilises with adjacent street sections being allocated to the same agent. Having used this relaxed bidding strategy to allocate surplus work to the most appropriate agents, the agents may then adopt the selfish strategy to prompt repair transactions to repair those routes that violate constraints. If another agent was required, then it would be added when each member of the community declines to bid for a street section. This will not happen until an interim solution has been found using the relaxed bidding strategy. The concept of switching between relaxed and selfish bidding strategies is referred to as dynamic bidding.

If the system is allowed to exchange work using the relaxed bidding strategy, eventually a state is reached whereby the same piece of work is continually cycled between two agents. When such a cycle is detected, the bidding strategy of agents may then be reset to the original selfish strategy. Allowing each agent to keep a trading list of those street sections

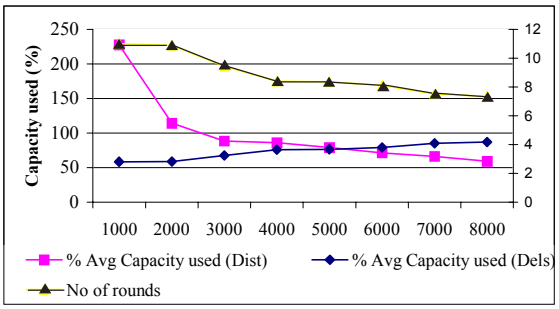
Table 3. Results obtained with the single-stage bidding and dynamic bidding on the Moredun dataset

	Selfish Bidding			2 Stage Bidding		
Max. Length Constraint	2K	2.5K	3K	2K	2.5K	3K
Avg. round Length	1583.2	1980.4	2175.0	1553.6	1955.9	2149.7
Std. Deviation (as a % of the average)	7.4%	7.7%	8.0%	6.1%	6.3%	7.4%
Over-length routes	0	0	0	0	0	0
Avg. extra sorts	7.1	7.0	4.4	8.4	6.8	4.0
Avg. supervised routes	4.6	4.5	4.4	4.7	4.8	4.3
% of supervised routes	37.4	45.7	48.4	37.8	47.8	47.3
Avg. No of routes	12.2	9.9	9.1	12.5	10.2	9.1

Figures 5a and 5b. Results obtained with the Fairmilehead dataset over a range of maximum delivery values



(a)



(b)

surrendered by the agent facilitates this change in strategy. The trading list is similar in concept to the tabu list used in Glover (1989, 1990) in that they both keep a form of record of recent actions. In a tabu list this information is used to prevent the algorithm re-examining solutions too frequently. In this context the trading list is used to indicate when the system has found an interim relaxed solution. A comparison of the two-stage strategy and the original selfish strategy may be seen in Figures 6(a) and 6(b). Although Figures 6(a) and 6(b) show a dramatic improvement on the Windmill dataset, dynamic bidding does not affect performance on the larger Moredun dataset as shown in Tables 3 and 4. Note that the number of routes required for routing around the Windmill dataset drops as a result of the modified bidding strategy. A side effect is the rising in average round lengths, although this is to be expected given the relationship between quantity of routes and round length (round lengths increase as the number of routes decreases).

Table 4. Significance tests for Table 3

Max. Length Constraint	T-test comparison (round length)	T-test comparison (rounds per solution)
2K	0.5	0.6
2.5K	0.4	0.8
3K	0.3	0

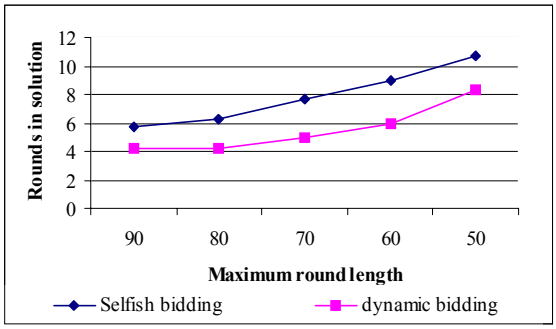
DYNAMIC DISTRIBUTION NETWORKS

This section sets out to examine the applicability of the techniques developed previously, to the optimisation of home delivery services offered by many supermarkets in the United Kingdom and elsewhere. Solutions to such problems may be optimised in terms of providing the fastest possible delivery service or providing the delivery service at a low cost. These two objectives are not always compatible; it may be difficult to provide a fast delivery service for a low cost.

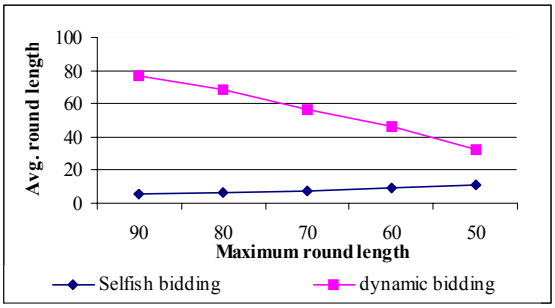
Problem Description

The problem discussed is a generic problem based on the services offered by a number of UK supermarkets. The problem is spread over

Figures 6a and 6b. Results obtained on the Windmill dataset



(a)



(b)

a period of time (0 <t< T); at each time interval t a number of deliveries will be added to the system. Deliveries are made from the supermarket by a number of vehicles; each vehicle collects a load, consisting of a number of items for delivery from the supermarket, and delivers those items before returning to the supermarket to collect another set of deliveries. In this case each vehicle has the capacity to undertake up to 50 deliveries on a route, subject to a maximum length constraint, before returning to the supermarket. The system must allocate deliveries to vehicles, and assign a departure time and route to each vehicle to allow it to make its deliveries and return for the next load.

The number of deliveries q added to the system each tick t is determined as follows:

$$q = \sin\frac{t}{20} * 2$$

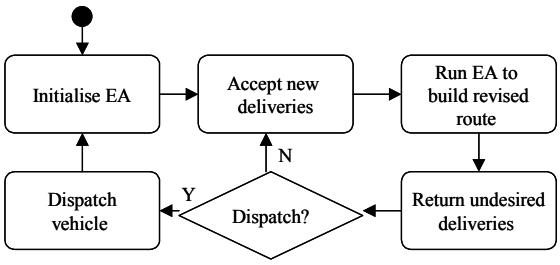
If q <=0 then a random value in the range 0..3 is used. This gives rise to a pattern of deliveries that represents periods of high and low demand. The control mechanism used to allocate work to the agents must manage the dynamic nature of the problem.

The Agent Architecture

The starting point for our solution was the agent architecture previously described. Each agent incorporates a version of SBR-EA that has been modified to remove those patterns that are specific to walking couriers. Each agent represents one vehicle; an important difference in the dynamic problem is that each agent must construct several routes during the course of the problem-solving process, a new route being required each time the vehicle returns to base and departs with another load of deliveries.

When a delivery is requested, the delivery is added to the appropriate SBR street section. If this street section is already allocated to an agent, then the delivery will be automatically incorporated within that agent's route. If the street section has no other deliveries, it is placed in a list of unallocated street sections awaiting allocation to an agent. Items in the unallocated list are distributed to agents by means of the auction system described previously. The lifecycle of an agent may be described as follows (see also Figure 7). The agent initialises its SBR-EA; it then accepts deliveries in the form of street sections or as extra deliveries to street sections currently

Figure 7. The lifecycle of an agent



incorporated in the route constructed by that agent. Each item in the unallocated list is offered for auction at each tick. The agents use the selfish bidding strategy, resulting in some items remaining in the unallocated list. Returned items are placed in the unallocated list. Although an agent will have work allocated to it and removed from it whilst it is constructing its route, the population is not re-initialised, but has the appropriate genes removed or added to it. The population is only re-initialised after a set of deliveries has been dispatched.

If allocating deliveries directly to a street section already incorporated within a route results in a constraint violation, the agent returns one or more street sections to the coordinator as described previously. If a delivery is required to a street section that has not already been incorporated into a route, then that delivery section is added to a list of unallocated street sections. Having processed all of the new street sections added at that particular time, agents are then allowed an opportunity to submit bids for those items contained within the surplus list.

The agent will continue accepting and rejecting street sections, and evolving a route for period of time, until it is decided to dispatch the deliveries allocated to it. Once an agent has dispatched a vehicle, it cannot re-dispatch for a given period of time to allow for the vehicle to make the deliveries and return to the supermarket.

The crucial decision that determines whether our solution will be optimised in terms of minimising delivery time or minimising cost is at what stage the agent should dispatch the deliveries. If the agent waits until the vehicle is full, this may result in deliveries having an unacceptably long delivery time. Departing too frequently will incur a greater travelling distance. To assist with determining the rate at which vehicles should depart, we have used an artificial currency s, which allows deliveries to "buy" the services of a delivery vehicle. In many organisations the prime motivation in decision making is the economic effects of the decision, more specifically the potential cost implications. The use of an economics metaphor should also make the system more understandable to users, allowing them to interact with the system and make more effective use of it.

When a critical mass of deliveries has grouped together, they will have enough spending power to buy the services of a vehicle to undertake their dispatch.

Each delivery within the system is allocated a subsidy s, for a given delivery h is initially calculated as:

s = dist(superMarket,h)* 0.1

where

dist(i,j) returns the distance from location i to location j

With each tick t, the delivery is in the system awaiting dispatch so the amount of subsidy allocated to it increases as follows:

s = s * subsidy_rate

where

subsidy_rate is a value in the range 1..2

Figure 8. Results obtained with the Moredun dataset

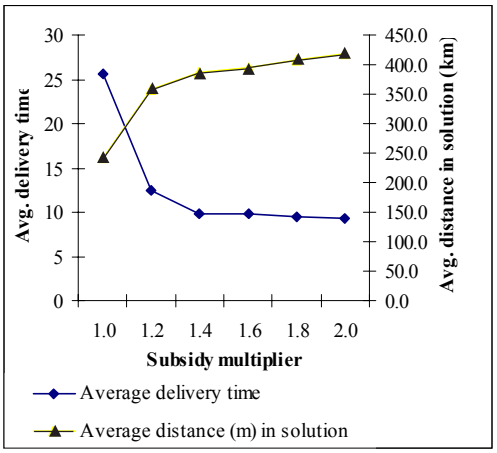
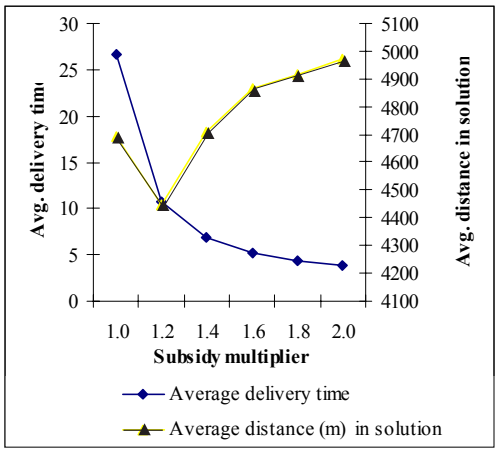


Figure 9. Results obtained with the Fairmilehead dataset



As the SBR representation clusters deliveries together into street sections, those sections with multiple deliveries will have a higher combined subsidy and are more likely to be delivered sooner. The longer a delivery is in the system awaiting dispatch, the more likely it is to be included within a delivery run, as the amount of subsidy allocated to it will increase. When an agent has to remove some work (due to an over length route or too many deliveries), the street section selected is that which attracts least subsidy.

The cost of dispatching a set of deliveries is calculated as the cost per km of running each vehicle. In this case the cost of delivering is 1,000 units of currency per km. For differing problems other costing factors may be taken into consideration, such as a fixed daily cost per vehicle.

Results for Dynamic Distribution Networks

Runs were carried out with subsidy rate in the range 1.0 to 2.0. Each problem runs for t = 250 using the demand pattern outlined earlier. In this instance, four agents were used to simulate four delivery vehicles. In each case the actual CPU time taken for each unit of time t was three seconds. For larger problems the agents may be distributed across multiple CPUs to allow concurrent evolution. The Fairmilehead and Moredun datasets described previously were used for testing.

Because of the non-deterministic nature of the EA, the results presented are the average of 20 individual runs with each setting of the subsidy rate. Reference to Figures 7 and 8 shows that the average time between a delivery being added to the system and being delivered decreases as the subsidy rate is increased. The best improvement happens in the range 1.2 to 1.4, with less improvement happening above 1.4. In Figures 8 and 9, the average delivery time initially increases with a subsidy rate of 1.2 and decreases in the range 1.2 to 2, suggesting that the best results may be obtained by varying the subsidy in the range 1.2 to 2.

The total distance covered by deliveries for the same set of runs is shown in Figures 8 and

9. As expected the total distance covered rises in proportion to the subsidy rate. The greatest improvement is again found in the range 1.2 to 1.4. When the subsidy rate is increased beyond 1.6, the total distance travelled increases, but with little corresponding reduction in average delivery time. Examination of these results suggests that a low value of the subsidy produces a solution that will be cheaper in terms of distance, but resulting in longer delivery times, whilst a higher subsidy rate will result in shorter delivery times but a higher cost.

CONCLUSION

This chapter has illustrated the means by which tools such as evolutionary algorithms, agents, and economic metaphors may be used to optimise distribution networks. The postal distribution problem was tackled using a community of agents incorporating SBR-EA to build a route around each copy of the algorithm. There exists a problem with combining the entire network into a single chromosome, in that a single mutation or crossover is likely to affect several or even all of the routes. The agent-based system overcomes these difficulties, each route being optimised by a separate agent. Those agents who are satisfying all of the hard constraints may simply evolve their route and never interact with other agents whilst the remainder of the problem is being solved. Implicit within the agent community is the dynamic valuation of work objects; the bidding value for any street section (the work object in this instance) varies as the bidding agent changes its route. This means that at any time it is possible to select an item of work and then rank the agents in the order that the agents desire the item of work.

The addition of dynamic bidding further improves the efficiency of the system, allowing it to cope with problems that are tightly constrained such as the Windmill dataset. Within the agent-based system, there does not exist a method for removing a delivery round. In solving optimisation problems it is often necessary to relax the problem by lessening (or removing) some of the constraints on it, allowing relaxed solutions to be constructed. Such solutions may then be used as a starting point to search for a solution once the original constraints have been re-imposed. By implementing dynamic bidding, with a relaxed stage that allows agents to disregard the maximum length and maximum deliveries constraints, the problem is relaxed in a manner that does not result in the wholesale destruction of a route, but simply in the re-allocation of work amongst agents.

The agent-based architecture is then applied to the problem of optimising the despatch times of vehicles in a supermarket delivery system. Unlike the postal problem this is a continuous dynamic problem. The arrival of goods into the problem and their subsequent departure upon delivery gives rise to an element of supply and demand. The artificial currency metaphor is introduced with the ability for goods to "pay" for delivery. It is demonstrated that the type of solution produced (optimised for speed or optimised for cost) may be determined by altering the subsidy rate. Given that such systems are likely to be used by individuals with no background in computing or AI, the ability to control the output in this manner makes their likely acceptance in the real world much higher.

REFERENCES

Bachem, A., Hochstättler, W., & Malich, M. (1996). The simulated trading heuristic for solving vehicle routing problems. *Discrete Applied Mathematics, 65,* 47-72.

Baraglia, R., & Hidalgo, J., & Perego, R. (2002) A hybrid heuristic for the travelling salesman problem. *IEEE Transactions on Evolutionary Computation, 5*(6), 613-622.

Blanton, J. L., & Wainright, R. L. (1993). Multiple vehicle routing with time and capacity constraints using genetic algorithms. In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 452-459). San Mateo, CA: Morgan Kaufmann.

Bousonville, T. (2001). Local search & evolutionary computation for arc routing in garbage collection. In L. Spector (Ed.), *Proceedings of the Genetic and Evolutionary Computation Conference 2001* (pp. 1137-1143). San Francisco: Morgan Kaufmann.

Cook, W., Cunningham, W., Pulleyblank, W., & Schrijver, A. (1998). The travelling salesman problem in combinatorial optimization. In *Cominatorial optimization* (pp. 241-271). New York: John Wiley & Sons.

Croes, G. A. (1958). A method for solving travelling salesman problems. *Operations Research, 6,* 791-812.

Czech, Z., & Czarnas, P. (2002). Parallel simulated annealing for the vehicle routing problem with time windows. In F. Vajda & N. Podhorszki (Eds.), *Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-Based Processing* (pp. 376-383). IEEE Press.

Davis, L. (1985). Applying adaptive algorithms to epistatic domains. *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 162-164).

Ferber, J. (1999). *Multi-agent systems: An introduction to distributed artificial intelligence.* Reading, MA: Addison-Wesley.

Freisleben, B., & Merz, P. (1996). New genetic local search operators for the travelling salesman problem. In H. Voigt, W. Ebeling, I. Bechenberg, & H. Schwefel (Eds.), *Proceedings of the 4th Conference on Parallel Problem Solving from Nature (PPSNIV)* (LNCS 1141, pp. 890-899). Berlin: Springer-Verlag.

Gambardella, L., & Dorigo, M. (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS Journal on Computing, 12*(3), 237-255.

Hoen, P., & La Poutré, J. (2003). A decommitment strategy in a competitive multi-agent transportation setting. In P. Faratin, D. Parkes, & J. Rodriquez-Aguilar (Eds.), *Proceedings of Agent-Mediated Electronic Commerce V* (pp. 1010-1011). Berlin: Springer.

Homaifar, A., Guan, S., & Liepins, G.E. (1993). A new approach on the travelling salesman problem by genetic algorithms. In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 460-466). San Mateo, CA: Morgan Kaufmann.

Ladányi, L., Ralphs, T. K., & Trotter, L. E. (2001). Branch, cut, and price: Sequential and parallel. In D. Naddef & M. Juenger (Eds.), *Computational combinatorial optimization* (pp. 223-260). Berlin: Springer.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the travelling salesman problem. *Operations Research, 21,* 498-516.

Mathias, K., & Whitley, D. (1992). Genetic operators, the fitness landscape and the travelling salesman problem. In R. Manner & B. Manderick (Eds.), *Proceedings of Parallel Problem Solving from Nature II (PPSN 2)* (pp. 219-228). Amsterdam: Elsevier.

Tamaki, H., Kita, H., Shimizu, N., Maekawa, K., & Nishikawa, Y. (1994). A comparison study of genetic codings for the travelling salesman problem. In Z. Michalewicz (Ed.), *Proceedings of the 1st (IEEE) Conference on Evolutionary Computing (ICEC'94)* (pp. 1-6). IEEE Press.

Thangiah, S. R. (1999). A hybrid genetic algorithm, simulated annealing and tabu search heuristic for vehicle routing problems with time windows. In L. Chambers (Ed.), *Practical handbook of genetic algorithms, complex coding systems* (Vol. III, pp. 371-381). CRC Press.

Thangiah, S., Vinayaamoorthy, R., & Gubbi, A. (1993). Vehicle routing with time deadlines using genetic and local algorithms. In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 506-515). San Mateo, CA: Morgan Kaufmann.

Urquhart, N., Ross, P., Paechter, B., & Chisholm, K. (2002a). Improving street-based routing using building block mutations. In S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf, & G. Raidl (Eds.), *Proceedings of the Applications of Evolutionary Computing Workshops* (pp. 334-342).

Urquhart, N., Ross, P., Paechter, B., & Chisholm, K. (2002b). Solving a real-world routing problem using multiple evolutionary algorithms. In J. J. Mercelo, P. Adamidis, H.-G. Beyer, J.-L. Fernandez-Villacanas, & H.-P. Schwefel (Eds.), *Proceedings of Parallel Problem Solving from Nature (PPSN VII)* (LNCS 2939, pp. 871-882). Berlin: Springer.

Urquhart, N., Paechter, B., & Chisholm, K. (2001). Street-based routing using an evolutionary algorithm. In E. J. W. Boers, S. Cagnoni, J. Gottlieb, E. Hart, P. L. Lanzi, G. R. Raidl, et al. (Eds.), *Proceedings of the Applications of Evolutionary Computing Workshops* (pp. 495-504). Berlin: Springer.

Whitley, D., Starkweather, T., & Shaner, D. (1992). The travelling salesman and sequence scheduling: Quality solutions using genetic edge recombination. In L. Davis (Ed.), *The handbook of genetic algorithms* (pp. 350-372). New York: Van Nostrand Reinhold.

Chapter XXXVIII
Games, Supply Chains, and Automatic Strategy Discovery Using Evolutionary Computation

Tim Gosling
University of Essex, UK

Nanlin Jin
University of Essex, UK

Edward Tsang
University of Essex, UK

ABSTRACT

The use of evolutionary computation is significant for the development and optimisation of strategies for dynamic and uncertain situations. This chapter introduces three cases in which evolutionary computation has already been used successfully for strategy generation in the form of work on the Iterated Prisoner's Dilemma, Rubinstein's alternating offers bargaining model, and the simple supply chain model. The first two of these show how evolutionary computation has been applied to extensively studied, well-known problems. The last of these demonstrates how recent statistical approaches to evolutionary computation have been applied to more complex supply chain situations that traditional game-theoretical analysis has been unable to tackle. The authors hope that the chapter will promote this approach, motivate further work in this area, and provide a guide to some of the subtleties involved in applying evolutionary computation to different problems.

INTRODUCTION

The use of evolutionary computation is important in the development of strategies for dynamic, uncertain situations or for any situation where a simple strategy has many parameters to tune. While game theory and theories of equilibrium are highly effective tools for the

analysis of various problems, they suffer from being unable to deal with the increased complexity and uncertainty inherent in many real-life situations. Strategies requiring a large number of parameters to be tuned cannot effectively be optimised by hand both because those numbers may be so large, but primarily because interactions between the parameters are often difficult to understand.

One such problem is that of supply chains and what strategies should be used by participants to operate effectively within them. Tackling this problem is important because trading electronically will become increasingly important in the future, and a need will exist, if it does not already, for many of the transactions to be handled fully automatically (He, Jennings, & Leung, 2003; Sandholm, 1999; Walsh, 2001). Even relatively simple supply chain scenarios prove difficult to analyse, and it is usually necessary to resort to domain knowledge in order to develop strategies. While this approach to strategy creation is capable of producing good solutions, it is difficult to foresee how they will respond in unexpected situations, guarantee robustness, and ensure maximum effectiveness in the face of change. Furthermore even an effective hand-crafted solution is likely to require a large number of parameters to be tuned, and doing this manually could well prove impossible either because the number of parameters is so large or because they interact in a way that is difficult to understand.

Evolutionary computation (EC) gives us the potential to address these issues. By defining the supply chain environment, or indeed any other environment, in terms of a reasonable strategy representation scheme and practical strategy evaluation mechanism, EC is able to evolve strategies and/or good parameter sets to tackle the problem.

In this chapter we will be looking at three different strategy generation problems and how EC can be used to tackle them. The first of these, Iterated Prisoner's Dilemma (IPD), introduces strategy generation using EC and shows how different algorithms have been used to tackle the same problem. The second problem, Rubinstein's alternating offers bargaining model (RAOBM), is used to demonstrate that EC can find a known optimal strategy. The final problem is defined by the simple supply chain model (SSCM). For the SSCM we show how EC can be used to tackle a far more complex strategy evolution problem by using a supporting strategy framework. In each case we examine why a particular EC algorithm is most appropriate, while discussing past efforts and presenting recent work.

GAME THEORY

Game theory has been highly successful in its application to situations such as the Prisoner's Dilemma (PD) and Rubinstein's bargaining game, along with many others. By starting from a notion of rationality and often complete information, it has proven invaluable and provided a good indication of how to behave in different situations. Since its initial formulation various theories of equilibrium have been posited to help explain how and why certain outcomes do (or should) occur within a game. Some of these, along with other terms, will be referred to during the course of this chapter; we briefly recap these now.

- **Dominant Strategy:** A strategy that yields superior results regardless of the opponent's move.
- **Dominant Strategy Equilibrium:** The outcome of a game reached when all players have a dominant strategy and play it.

- **Nash Equilibrium:** The set of possible results reached by players playing the best possible strategy in response to their opponents move.
- **Sub Perfect Game Equilibrium (SPE):** The result of a game if each player moves such that a Nash Equilibrium strategy is played at each sub game, avoiding the worst possible outcomes at each stage of a game.
- **Evolutionary Stable Strategy (ESS):** A strategy that dominates the population and cannot suffer from invasion by other (mutant) strategies.

Game theory, as stated in the introduction, while highly successful in analyzing different situations, has difficulty dealing with problems containing a considerable degree of uncertainty or of a highly dynamic nature (essentially the same thing). If a problem cannot effectively be captured, its subsequent analysis by game theory is not possible. While ESS can help explain why and under what conditions a particular strategy may become dominant within a population, it cannot tell us what that strategy may be without the associated prior game analysis. Evolutionary computation, by comparison, offers a way to develop strategies from scratch and discover which (if any) of these are dominant; provided that a good strategy representation scheme and evaluation method are used.

EVOLUTIONARY COMPUTATION AND PBIL

As is described in earlier chapters, evolutionary computation covers a wide range of powerful problem solving tools, or evolutionary algorithms (EAs), that have been inspired by nature and make use of the concept of natural selection to improve a population of solutions. Three of these algorithms are considered in this chapter, genetic algorithms (Mitchell, 1998), genetic programming (Banzhaf, Nordin, Keller, & Kaufmann, 1998), and population-based incremental learning (Baluja, 1994; Sebag & Ducoulombier, 1998).

The last of these, PBIL, is a relatively new statistical approach to EC that combines the concept of a GA with that of reinforcement learning techniques (such as neural networks).

A PBIL algorithm may make use of the same solution representation as a GA; however, instead of a population of solutions, PBIL makes use of a probability distribution. The probability distribution represents the likelihood of a solution string's elements (or alleles) of taking on a particular value. Test solutions are generated from this distribution, evaluated, and used to reinforce the distribution; good solutions increase the likelihood of their elements' values recurring in the future and the reverse for bad solutions. Like GA, PBIL may make use of mutation to help increase solution diversity and forms of elitism to focus the search (Gosling 2005). While quite new, PBIL has already proven useful for various types of problem solving (Sukthankar, Baluja, & Hancock, 1998; Inza et al., 1999). The basic operation of a PBIL algorithm is shown in Figure 1.

Figure 1. Basic operations of PBIL

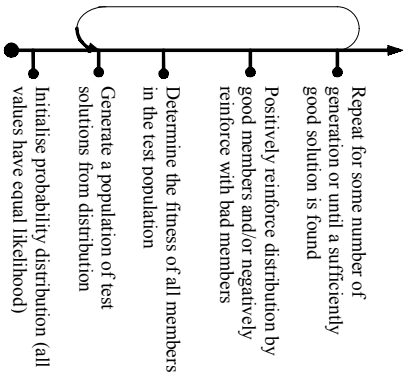
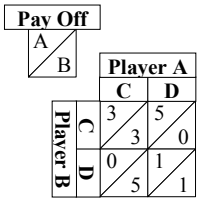


Figure 2. Prisoner's Dilemma payoff table



While discussing EAs throughout this chapter, two key features will recur. First, the solution representation used by the algorithm is critical to the success of the algorithm in tackling the problem; a good representation should reduce the search space as far as possible and limit or remove the possibility of invalid solutions being generated to avoid a combinatorial explosion. Secondly, the evaluation mechanism must successfully distinguish the quality of different solutions, but at the same time be computational efficient.

ITERATED PRISONER'S DILEMMA

Introduction

Iterated Prisoner's Dilemma is an extension of the well-known Prisoner's Dilemma (PD) game. In PD two players, A and B, play two possible moves, cooperate or defect, simultaneously. The combined choices determine each player's score. IPD is PD played over some number of rounds, the scores accumulating. The payoff table for PD is shown in Figure 2.

The sole Nash Equilibrium for PD is the play (defect, defect). If the number of rounds is known, this is also the best play for IPD. If, however, the number of rounds is unknown, an incentive exists for each player to cooperate in order to avoid uncertain future punishments from the other for defecting. How to play under these circumstances is open to debate; cooperate is desirable, as the long-term payoff for the players would be better but the strategy should not be open to exploitation.

EC has previously been used to study IPD strategies. In 1987 Axelrod first did so, running a competition in which strategies were learned using a GA-based system. Axelrod (1987) concluded that tit-for-tat (TFT) was the dominant strategy for IPD. This result sparked some debate with various other experiments and analysis by other researchers, some of whom considered this conclusion incorrect (Binmore, 1994, 1998; Linster, 1992; Kraines & Kraines, 1993). Further EC-based work is discussed below in relation to IPD strategy representation.

We now introduce recent experiments conducted into IPD using a similar setup to Axelrod. This work aimed to discover how effective population-based incremental learning (PBIL) was for strategy generation using the more established genetic algorithms for comparison (Gosling, Jin, & Tsang, 2004, 2005).

Representing IPD Strategies

When developing a strategy, it is important to determine what the players know about the situation, history of play, each others' behaviour, and how, generally, they should respond to that information (probabilistically or deterministically). The answer to these questions has a profound effect on the nature of the strategies that can be produced, how they can be represented, and subsequently on the type of EAs that may be used to evolve them.

In answering these questions for IPD, a considerable degree of variation is possible. For instance Nowak and Sigmund (1992, 1993) dealt with players that responded probabilistically to a memory of only the last round of play. The

strategy representation scheme consisted of four variables, the chance of defection given each possible payoff. Evolving strategies using this representation led to a dominant Pavlov strategy (Kraines & Kraines, 1993) occurring within the population.

Crowley (1996) dealt with varying player memories and IPD strategies based on sets of hierarchical rules. The rules essentially pattern matched different situations within the player memory with more explicit rules having greater precedence. For small rule sets and low memory, Crowley found that rules similar to TFT would evolve, but in the case of longer memory and large possible rule sets, he noted that far more complex strategies emerged. Crowley argued that these more complex strategy hierarchies might provide some indication of how cooperation evolves within nature.

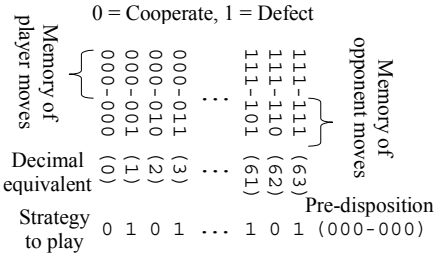
Further examples of the diversity of representations possible when using EAs are Fogel (1993), who evolved finite state machines in an attempt to probe the necessary conditions for cooperation to emerge, and Jang, Whigham, and Dick (2004), who determined the effect on behaviour of players with no memory that used fixed sequences of moves with varying lengths.

In all of the above cases, a variation on GA was used to evolve strategies. This was possible since the types of strategies selected and their representations could be thought of as fixed-length strings—an ordered set of variables that required optimisation. Since GAs are designed for the optimisation of such strings, they could be applied successfully to these situations. GP, by comparison, makes use of variable size tree representations of a solution and so is unsuitable in this instance.

PBIL, however, can make use of the same representation schemes as a GA which should make it equally applicable.

In recent work we examined the relative effectiveness of PBIL and GA in the context of IPD. For our experiments we selected the classic Axelrod representation. This representation is based upon players responding deterministically to a memory of only the last three rounds of play. Since both players make one move per round and there are only two possible moves (cooperate or defect), a complete player memory comprises six elements each of two possible values. This memory can be translated into six binary bits, with 1 representing defect and 0 cooperate. Six bits can be arranged in only 64 possible ways, so with consistent play, a player has only 64 possible responses to its memory. We therefore use a 64-bit string to represent the player's strategy, one bit for each possible memory configuration. Since at the beginning of the game a player would have no past memory, an additional six bits is provided to represent a player's starting memory or predisposition. Thus the total strategy representation is 70 bits in length. This is shown in Figure 3.

Figure 3. Axelrod IPD representation



GA vs. PBIL Comparison

In order to compare GAs and PBIL, two systems were set up and run independently. Both algorithms were used to produce a set of strategies that were then compared in a tournament.

Comparing the GA and PBIL approaches in this way was not straightforward. While the

two approaches used the same representation and both used tournaments to evaluate their strategies, each operate in very different ways. The PBIL test population cannot be directly compared to the GA's population for instance. For the GA, the entire population is evaluated, and essentially all (or at least much) of that information used for the ongoing evolutionary process. With a PBIL implementation, only one or two members of the test population are used to update the probability distribution, so for a large population, much of the evaluation information would be lost. Comparing GA and PBIL algorithms based on similar population sizes and number of generation would therefore be unfair.

To achieve a fairer comparison the PBIL system here made use of a relatively large pool of generated test strategies. Each of these test strategies was played against members of a smaller evaluation population. The best scoring member from the evaluation population was then used to reinforce the probability distribution. In this way members of the evaluation population receive high-quality evaluations, and fewer evaluations are performed per generation. Comparing the GA and PBIL can now be done on the basis of the number of evaluations (IPD games played) used by each. The formula below shows how the populations and generations of each system may be balanced to allow the comparison:

GAevals & PBILevals - Total number of solution evaluations
GApop - The GA population size
GAgens - The number of generations the GA runs for
PBILpop - The population size of the PBIL system
PBILtestpop - The testing pool size of the PBILsystem
PBILgens - The number of generation the PBILsystem runs

$$GA_{evals} \quad GApop \quad GApop \quad 1 \quad GAgens$$
$$PBIL_{evals} \quad PBILpop \quad PBILtestpop \quad PBILgens$$
$$GA_{evals} \quad PBIL_{evals}$$

Table 1. Best GA and PBIL configurations

Type	**GA**	**PBIL**
Population Size	100	5
Learning Pool	NA	99
Mutation Rate	0.007	Not Used
Learning Rate	NA	0.025
Generations	300	6000
Data Points	300	300

In the graphs that follow, the GA and PBIL strategies were compared at time steps equivalent in terms of number of evaluations. The interval between time steps (in PBIL generations) can be found by the following:

$$ComparisonOutputInterval = PBILgens / GAgens$$

Results and Conclusion

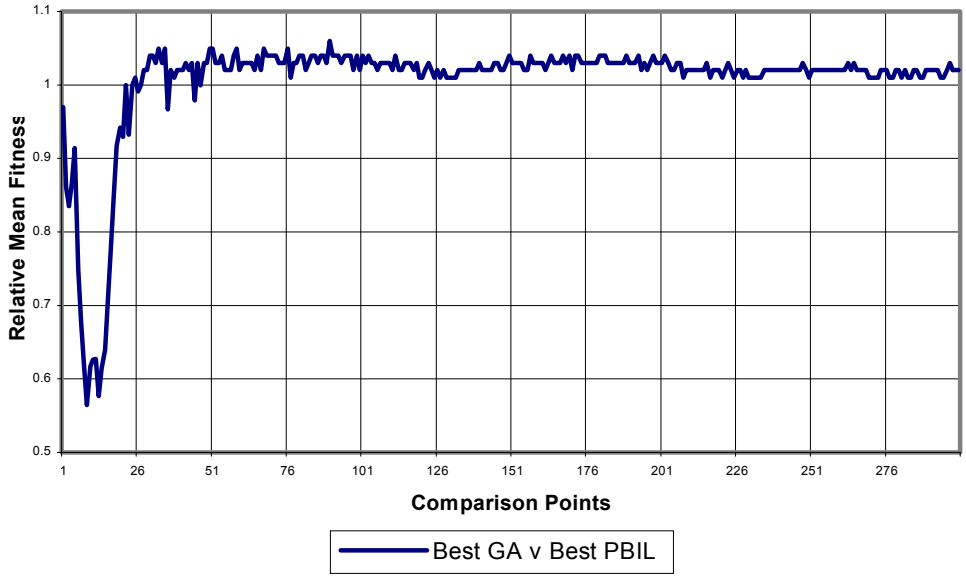
While an extensive set of comparisons was made, the best PBIL and GA parameter sets used are listed in Table 1.

In order to provide a comparison of relative effectiveness over time, the experiments were repeated 100 times for each set of parameters and the resulting strategies played against one another.

PBIL consistently performed slightly worse against the best GA configuration, although in the early stages it does a little better than the GA.

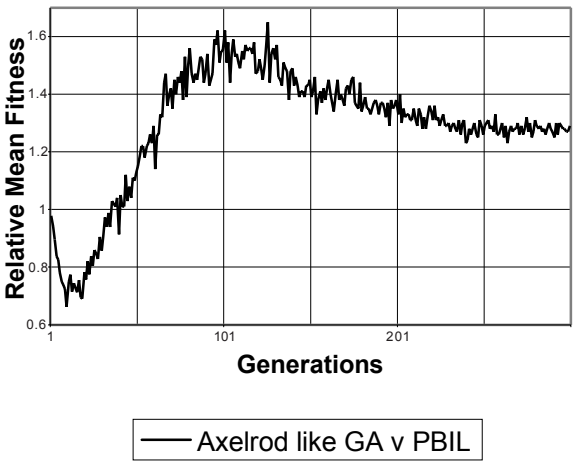
Under low population conditions however, a PBIL with an identical number of evaluations does better than GA. Reproducing similar GA conditions to those used by Axelrod, for example, shows PBIL to have superior performance to GA.

Figure 4. Mean GA strategy fitness/mean PBIL strategy fitness over time; PBIL starts off well but ultimately does slightly worse than GA; value greater than one indicates superior GA performance



The reason for this appears to be that the theoretically infinite population of PBIL is able to overcome the shortcomings of a lack of solution diversity within a small GA population.

Figure 5. PBIL vs. GA under Axelrod-like conditions; PBIL rapidly performs better than GA; values greater than one indicate superior PBIL performance



The conclusion of this work was that GA in general is slightly superior to standard PBIL for strategy generation in this context. PBIL, however, is more effective when only a small population size or a small number of evaluations are possible.

Efforts to improve PBIL further (Gosling et al., 2005), using a novel mutation operator, have allowed it to compete favourably with GA in this context under all conditions.

The use of EAs in this and other work supports the idea that EC is effective at the generation of game strategies.

RUBINSTEIN'S BARGAINING GAME

Introduction

Rubinstein's alternating offer bargaining model (RAOBM; Rubinstein, 1982) is a simple economic complete information game that involves

the division of some quantity, a unit pie for instance, between two players. The aim of the game is for the two players (A and B) to come to a mutual agreement about how to divide the pie. The game proceeds in rounds: in each round one player is able to make an offer for how much he or she should receive, and his or her opponent is able to accept or reject that offer. If the offer is rejected, another round occurs with the rejecting player making the next offer. This continues until agreement is reached. To add incentive for the players to agree, each is subject to a discount factor, thus how much each receives is reduced by the effects of waiting to obtain it. Since this is a complete information game, the players each know their and their opponent's discount factors. Game theoretic analysis of this game provides a Sub Game Perfect Equilibrium as follows (see Rubinstein, 1982; Muthoo, 1999 for proof):

$DisA$ = Player A discount factor (makes first offer)

$DisB$ = Player B discount factor (responds first)

$$GetsA = \frac{1 - DisB}{1 - DisA \times DisB}$$

$$GetsB = 1 - GetsA$$

If PlayerA, the first player to make a move, uses the above formula to calculate his or her offer (*GetsA*), there is no rational reason (from a game theory point of view) for the opponent, PlayerB, to reject it (and obtain *GetsB*).

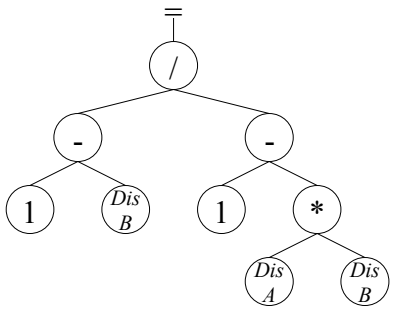
A considerable body of work exists studying the RAOBM, both in its traditional form and under various alternative conditions. When the model is altered such that the players have incomplete information (Rubinstein, 1985; Ausubel, Crampton, & Deneckere, 2002; Fatima, Wooldridge, & Jennings, 2001, 2005) or are boundedly rational or irrational (Kreps, 1990; Myserson, 1991), the scope for individual strategies increases dramatically. EC has also been applied to the RAOBM, comparisons being made to the Sub Game Perfect Equilibrium under various conditions (Binmore, Piccione, & Samuelson, 1998; Fatima et al., 2003; Jin & Tsang, 2005).

While games such as RAOBM have been extensively studied and solutions are known for various conditions, other games can prove too complex for traditional analysis. EC is of use in studying such games, both to obtain an idea of equilibriums that may exist for the game and to provide a reasonable playing strategy. To have confidence in this idea, we now introduce recent work that compared the game theory results for RAOBM with game playing strategies evolved using genetic programming (Jin & Tsang, 2005). The aim of this work was to establish if GP could be used to effectively approximate the game's SPE and so provide a case for its use in tackling problems that are too difficult to analyse with traditional game theoretic approaches.

Representing RAOBM Strategies

The first step in tackling RAOBM with EC was the same as with the IPD problem above, that is, one of representation. While RAOBM is a complete information game, the aim of approximating the SPE (and so finishing in the first round) leaves players with little knowledge. Assuming a unit pie, the players only know one another's discount factors and who starts first. With this in mind the objective becomes one of finding a representation that can make use of this information to come up with a good first offer from the starting player (PlayerA) that would be accepted (hopefully) by the second player (PlayerB). Essentially what we are looking for is a mechanism that allows the evolution of formulae that can tie the discount factors together to generate an offer (and accept/reject threshold).

Figure 6. Example of a GP structure showing how the game's SPE would be represented



GA and PBIL are not easily able to do this as they deal with fixed-length string representations of problems. Representing solutions is possible, for instance by allowing evolution of the offer/threshold directly, but these may lack generality (as in this case).

Genetic programming (GP) by contrast is able to evolve variable size tree structures that may represent formulae directly. Instead of defining the meaning of a string, the use of GP requires the selection of an appropriate symbol set to use within the tree structures. In general keeping the set of symbols as simple as possible is the best strategy, allowing evolution to do the rest. In the case of the RAOBM, we used the set of non-terminal symbols [+,-,/,*] and terminal symbols [*ADis*, *BDis*, 1, -1] (*ADis* and *BDis* being the PlayerA and PlayerB discount factors respectively). This symbol set is simple but provides sufficient flexibility for evolution of formulae, like that of the game's SPE, to occur (see the following).

Because the roles of PlayerA and PlayerB within the RAOBM game are slightly different, two separate populations are maintained for PlayerA and PlayerB strategies. To determine the fitness of a tree structure within a given population, it is evaluated and used to play games against all of the structures in the opposing population. The resulting accumulated payoff is used as the structure's fitness.

The use of two distinct populations is known as a co-evolutionary approach and is appropriate when competing strategies have to operate under different conditions from one another. In this case PlayerA and PlayerB strategies would evolve such that PlayerA represents the SPE shown above, while PlayerB would evolve a correspondingly different structure that would accept the value generated by PlayerA.

Table 2. GP configuration parameters

Parameter	Value
Nodes	Non-Terminal (+,-,*,/) Non-Terminal (1,-1,DisA,DisB)
Population Size	100 * 2 (200)
Generations	300
Initial Tree Depth	5
Maximum Nodes	50
Mutation Rate	0.01-0.5
Crossover rate	0-0.1

Table 3. ROABM results, GP approximates the game SPE well

Sets of discount factors (*ADis*,*BDis*)	SPE (*GotA*)	GP – Experimental Average of Player A	Standard Deviation of Player Average
(0.1, 0.4)	0.625	0.9101	0.0117
(0.4,0.1)	0.9375	0.9991	0.0054
(0.4,0.4)	0.7143	0.8973	0.0247
(0.4,0.6)	0.5263	0.509	0.0096
(0.4,0.9)	0.1563	0.1469	0.1467
(0.5,0.5)	0.6667	0.6745	0.0271
(0.9,0.4)	0.9375	0.9107	0.0106
(0.9,0.6)	0.8696	0.8	0.1419
(0.9,0.9)	0.5263	0.5065	0.1097
(0.9,0.99)	0.0917	0.1474	0.1023

Experiments and Results

Using the system of representation and co-evolution described above, experiments were run using the following parameters' GP:

The experiments were run 100 times each on 10 sets of discount factors; the average score of the best performing individuals from the final generation were used to determine how closely the population had converged towards the SPE (see Table 3). As can be seen the SPE was approximated reasonably well in most cases. It may be observed that this approximation began to break down where the discount factors tended towards the extreme.

Conclusion

GPs' more complex representation scheme can be used effectively to approximate game theoretically derived equilibriums.

The GP-derived approximation tends to break down when conditions are extreme.

While GP does not provide an exact match for the theoretically derived SPE for this game, it does provide a reasonable approximation in most cases. This tends to suggest that GP would be useful in studying other problems that require a more complex representation.

THE SIMPLE SUPPLY CHAIN MODEL

Introduction

At present, various electronic marketplaces, auctions, and negotiation systems exist. In the near future, full electronic supply chains will be possible and indeed desirable to improve efficiency (He et al., 2003; Sandholm, 1999; Walsh, 2001).

This situation, however, presents a problem. While humans are good at negotiations and situation analysis, they are less able to handle large volumes of information and numbers of transactions. What is needed is a computer-based system or strategy for handling these situations. The strategy does not need to be the perfect negotiator, although it must be competent, but it must be able to deal with negotiations more rapidly than a human operator could. As has been stated, while traditional economic approaches are effective in analysing simple games, they fail to tackle the more dynamic problems faced in supply chain situations and as such cannot be made full use of. The application of knowledge and experience to develop strategies is possible, but suffers from uncertainty about how robust these strategies would be, especially in unusual circumstances, and how to optimise them for maximum effect.

Making use of evolutionary computation within this domain is reasonable, given its application to other economics problems. As we have shown, provided we can define a reasonable strategy representation, it should be possible for an EA to evolve an effective and robust solution.

To begin tackling the supply chain problem, it is first necessary to model the supply chains we are interested in more precisely. A system such as the Simple Supply Chain Model (SSCM) provides one such way, and we will introduce this shortly.

Having accomplished this, the next task, as discussed earlier, is to develop a system of representation for possible solutions and a framework within which that representation may be used and evaluated. We also need to consider what sort of evolutionary algorithm would be appropriate for the learning process and how it should be applied.

It should be noted that considerable effort has gone into using EC and other techniques for negotiation and bargaining with computers. The Trading Agent Competition (Wellman, Greenwald, Stone, & Wurman, 2000) for example partially inspired the SSCM. Some examples of work in negotiation are Sandholm and Vulkan (2002), Fatima et al. (2000), and Bartolini, Preist, and Jennings (2005), while Fatima et al. (2005b) provide a comparison of evolutionary and game-theoretic approaches to bargaining.

The SSCM

The simple supply chain model (Gosling, 2003a; Gosling & Tsang, 2003b) has been developed to allow the specification of a simple supply chain starting state. To this end it models three different types of participants in the supply chain—customers, suppliers, and middlemen.

Customers have requirements that they wish to be fulfilled. These requirements are for a set of goods at some maximum price within a certain timeframe. Customers require the use of a middleman to obtain these sets of goods, and so have knowledge of some set of middlemen and a maximum outbound communication capability.

Suppliers are able to supply goods at some minimum price and some maximum quantity over the course of the scenario being modelled. They sell via the middlemen and so have a known set of middlemen along with a maximum outbound communications allocation.

Middlemen are responsible for matching up sets of requirements to available products in an attempt to make a profit. They are defined purely in terms of their known suppliers, customers, and a maximum outbound communications capacity. These are the focus of study here.

The SSCM defines supply chains in terms of these different participants, the set of products, the amount of time available for deals to be

struck, and the communication scheme used by the participants to interact. The SSCM does not impose restrictions on the way in which participants may attempt to resolve the chain, only the way in which the chain is initially set up and the means by which communication can occur.

Representation and Evaluation

Determining a representation scheme and evaluation mechanism for SSCM strategies is challenging simply because the number of possibilities are large. To reduce the scope somewhat, first we define different SSCM scenarios that restrict further the conditions the participants may face under a given SSCM instantiation, and secondly concern ourselves primarily with the middleman strategy and assume simple strategies on the part of the customers and suppliers. These restrictions may be relaxed later.

In the simplest scenario we assert that there is one supplier per product, that suppliers are passive and have no knowledge of the middlemen to begin with, and that they will not initiate contact with a middleman once known. Middlemen have no prior knowledge of customers but know of each of the suppliers they may need to fulfil a customer's requirements. Customers know only of one middleman each and initiate contact sometime prior to their earliest cut-off point for obtaining goods. These restrictions simplify the middleman strategy both by removing the need to mitigate the effects of customers attempting to find deals elsewhere and reducing the choice of suppliers. In more complex scenarios, these restrictions have been relaxed.

With this first scenario as a starting point, it is possible to begin defining a strategy representation.

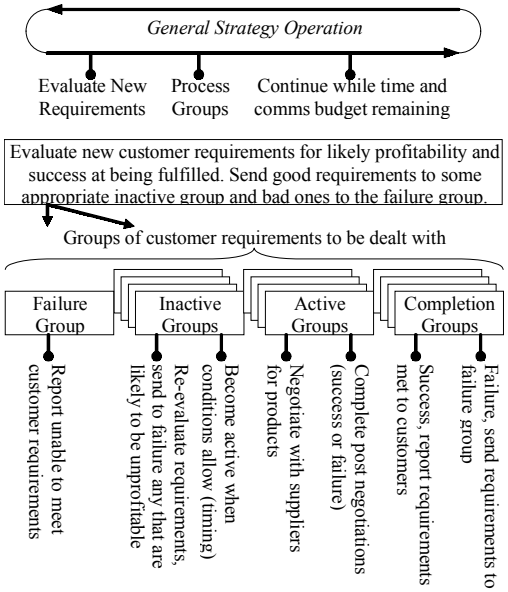
Initially we consider how the evaluation of any resultant strategy should be undertaken. The problem maps well into a multi-agent environment, and so it makes sense to build a market simulation system within which the participants can be configured in line with the SSCM and use their strategies to attempt to resolve the chain. When the chains' run time is up, the effectiveness of each strategy can be assessed. Since this process is likely to be computationally expensive and/or time consuming, it is reasonable to assume the total number of participants within the system will be limited. For this reason it would not be possible to evaluate many strategies simultaneously. From the discussion of IPD, strategy generation above this would suggest that PBIL would be superior to GA under these conditions since it is effective at leveraging small test populations for learning.

A second consideration is the complexity of the strategies to be used. The initial reaction is that GP would provide the flexibility required to define a complex SSCM strategy, and this would certainly be the case. The problem with this approach however comes in two parts: firstly defining a symbol set of sufficient subtlety and complexity to capture the various aspects of a participant's role is difficult; secondly having defined such a set, ensuring that viable strategies' results are problematic. While evolution is powerful, the representation must provide some guidance for it to stand a good chance of success. In each case it seems reasonable to provide a basic strategy framework within which the algorithm can evolve the control aspects of the strategy. This removes the problem of wholly invalid strategies being developed and helps reduce the complexity of the symbol set. The downside of this is that multiple elements within the framework would need to be evolved simultaneously, and the complexity of how to combine these multiple elements would additionally complicate the use of a GP algorithm. To simplify the strategy

problem further, the framework can be extended with reasonable control elements the parameter of which may then be evolved by an algorithm. If this approach is taken far enough, it is possible to remove the need for GP altogether and evolved the parameters directly. This is what we have done here, building on Matos, Sierra, and Jennings's (1998) bargaining work in particular for the agent negotiation elements. With the strategy representation reduced to a fixed-length string of parameters, it is possible to use PBIL or a GA. As stated, a GA would have difficulties under the limited population size available (as indeed would have GP), so we elect to use PBIL in this instance.

Having selected the algorithm and approach to be taken, it is necessary to outline the SSCM Strategy Framework, its evolvable parameters, and the market simulation system with which it will be used.

Figure 7. SSCM strategy basic framework



SSCM Strategy Framework

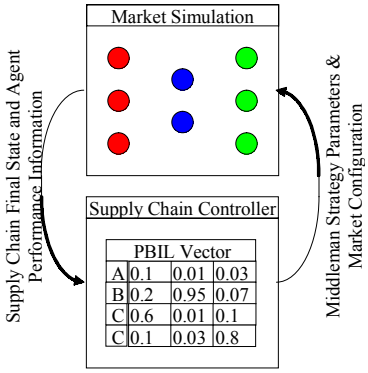
The SSCM strategy framework (SSF; Gosling, 2003c) is based around the idea of grouping together customer requirements and handling them as a conglomerate. Incoming customer requirements are first evaluated and then assigned to a group depending on some set of characteristics. The possible groupings include one for handling requirements the system has deemed impossible to fulfil or unprofitable. Requirement groups start in an inactive state (in which requirements are continually re-evaluated), progress to becoming active (during which supplier negotiations are undertaken), and finally move to a completion state for reporting back to the customers. The basic outline of this process is shown in Figure 7.

Primary parameters within the SSF are those relating to the evaluation of customer requirements, the dispersal of requirements to groups, and the negotiation mechanism used with the supplier. For example, the negotiation process, based on work by Matos et al. (1998), requires a set of 14 parameters for each product type under consideration. These parameters control estimates for likely values of products, tactics used for negotiation, and importance weighting for those tactics. Other parameters include control for how quickly groups should become active and what requirements should be accepted.

Market Simulation System

The SSCM market simulation system (SMSS) provides an environment within which the SSF may be used and under which the parameters are evolved. The SMSS consists of two core components, an agent-based supply chain simulator and a market controller. The market controller maintains a PBIL vector that provides strategy configuration parameters to the supply chain agents. Further, the controller sets up the

Figure 8. Market simulation system operation



supply chain and evaluates the performance of agents once completed. This information can be used to reinforce the PBIL vector for future generations of supply chain players. This process is briefly outlined in Figure 8.

The controller is able to configure the supply chain in such a way as to provide a different environment in which to evolve strategies. Examples of different environments are ones where the available goods are scarce or customer budgets are very limited.

The result of the SMSS is the controller's final PBIL vector state—this should contain an effective strategy for the environment presented.

Results and Conclusion

The main focus of experimentation with the SMSS was to determine if strategies could be evolved within the environment presented and what the limits of adaptability were.

It was found that effective strategies emerged within the SMSS and, as expected, that substitution of those strategies into new environments leads them to adapt to the new conditions, suggesting no universal strategy is optimal across all conditions.

Having determined that strategies could evolve within the SMSS, we then probed the limits of the system by adjusting the environment in such a way that it became difficult for middlemen to make a profit. This was accomplished by increasing the stubbornness of suppliers negotiating over prices. These efforts led to the determination of an adaptation boundary for this parameter beyond which the system was unable to evolve effective strategies. Further work suggested that using pre-evolved strategies close to that boundary condition would allow for adaptation under the harsher conditions.

While these results have proven interesting, the question of how to analyse them further has proven to be one of considerable importance, visualisation has certainly helped, but obtaining definitive evidence of why the strategies have adapted to the environment in a certain way has proven more difficult. To this end, analysis of the results ideally requires the development of further analytical tools, and this is currently the focus of much effort.

Overall EC has proven effective for evolving strategies in the complex, dynamic environment offered by the SSCM, and the SSF and SMSS have proven an effective way of harnessing the power of PBIL to this end.

CONCLUSION

This chapter has introduced evolutionary computation in the context of two well-known games (IPD and RAOBM) and the more complex SSCM. For these games we have shown that EC is able to evolve effective strategies that equate to the known equilibriums. For the SSCM we have shown that with careful consideration it is possible to evolve successful strategies within a strategy framework and supply chain simulation system. Since game theory cannot

effectively deal with the uncertainties inherent in situations like the SSCM, we assert that EC, used appropriately, provides a good alternative for this problem and other complex real-life problems.

On a cautionary note, while EC is effective for strategy generation, care must be taken with the design of a good representation, the selection of an appropriate algorithm, and the choice of a reasonable evaluation scheme. A further consideration is that of analysis. As the disagreement over stable IPD strategies demonstrates, results may still be open to interpretation. In the context of the SSCM, reaching a full understanding of the results is an issue.

Finally, evolutionary computation has many advantages for the generation or optimisation of strategies in challenging environments; this approach has had a successful beginning, but its future depends on carefully considered application.

REFERENCES

Ausubel, L. M., Crampton, P., & Deneckere, R. J. (2002). Bargaining with incomplete information. *Handbook of game theory* (Vol. 3). Amsterdam: Elsevier Science.

Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. Genetic algorithms and simulated annealing, *Research Notes in AI,* 32-41. San Francisco: Pitman/ Morgan Kaufmann.

Banzhaf, W., Nordin, P., Keller R., & Kaufmann M. (1998). *Genetic programming: An introduction.* San Francisco: Morgan Kaufmann.

Bartolini, C., Preist, C., & Jennings, N.R. (2005). A software framework for automated negotiation. *Software engineering for multi-agent systems III: Research issues and practical applications* (pp. 213-235). Berlin: Springer Verlag.

Binmore, K. (1994). *Game theory and the social contract I, playing fair.* Cambridge, MA: MIT Press.

Binmore, K. (1998). *Game theory and the social contract II, just playing.* Cambridge, MA: MIT Press.

Binmore, K., Piccione, M., & Samuelson, L. (1998). Evolutionary stability in alternating-offer bargaining games. *Journal of Economic Theory, 80,* 257-291.

Baluja, S. (1994). *Population-based incremental learning—A method for integrating genetic search-based function optimisation and competitive learning.* Technical Report No. CMU-CS-94-163, Carnegie Mellon University, USA.

Crowley, P. H. (1996). Evolving cooperation: Strategies as hierarchies of rules. *BioSystems, 37,* 67-80.

Faratin, P., Sierra, C., & Jennings, N. R. (2000). Using similarity criteria to make negotiation trade-offs. *Proceedings of the 4th International Conference on Multi-Agent Systems (ICMAS-2000)*, Boston (pp. 119-126).

Fatima, S.S., Wooldridge M., & Jennings, N. R. (2001). Optimal negotiation strategies for agents with incomplete information. *Proceedings of the 8th International Workshop on Agent Theories, Architectures and Languages (ATAL)*, Seattle (pp. 53-68).

Fatima, S. S., Wooldridge, M., & Jennings, N. R. (2003). Comparing equilibria for game-theoretic and evolutionary bargaining models. *Proceedings of AAMAS 2003, Workshop on Agent-Mediated Electronic Commerce V.*

Fatima, S. S., Wooldridge M., & Jennings, N. R. (2005). Bargaining with incomplete infor-

mation. *Annals of Mathematics and Artificial Intelligence.*

Fatima S. S., Wooldrige M., & Jennings N. R. (2005b). A comparative study of game theoretic and evolutionary model of bargaining for software agents. *Artificial Intelligence Review, 23*(2).

Fogel, D. B. (1993). Evolving behaviours in the Iterated Prisoner's Dilemma. *Evolutionary Computation, 1*(1), 77-97.

Gosling, T. (2003a). The simple supply chain model and evolutionary computation. *Proceedings of the Congress on Evolutionary Computation 2003 (CEC2003)*.

Gosling, T., & Tsang, E. (2003b). *The simple supply chain model (SSCM).* Technical Report CSM-392, Department of Computer Science, University of Essex, UK.

Gosling, T. (2003c). *The scenario one strategies.* Technical Report CSM-394, Department of Computer Science, University of Essex, UK.

Gosling, T., Jin, N., & Tsang, E. (2004). *Population-based incremental learning versus genetic algorithms: Iterated Prisoner's Dilemma.* Technical Report CSM-401, Department of Computer Science, University of Essex, UK.

Gosling, T., Jin, N., & Tsang, E. (2005). Population-based incremental learning with guided mutation versus genetic algorithms: Iterated Prisoner's Dilemma. *Proceedings of the Congress on Evolutionary Computation 2005* (CEC2005).

He, M., Jennings, N.R., & Leung, H. (2003). On agent-mediated electronic commerce. *IEEE Transactions on Knowledge and Data Engineering, 15*(4), 985-1003.

Inza, I., Merino, M., Larrañaga, P., Quiroga, J., Sierra, B., & Girala, M. (1999). *Feature subset selection by population-based incremental learning. A case study in the survival of cirrhotic patients treated with TIPS.* Technical Report No. EHU-KZAA-IK-1/99, University of the Basque Country, Spain.

Jang, D., Whigham, P. A., & Dick, G. (2004). On evolving fixed pattern strategieI for iterated Prisoner's Dilemma. *Proceedings of the 27th Conference on Australasian Computer Science* (pp. 241-247).

Jin, N., & Tsang, E. (2005). Co-evolutionary strategies for an alternating-offer bargaining problem. *Proceedings of CIG2005.*

Kraines, D., & Kraines, V. (1993). Learning to cooperate with Pavlov—An adaptive strategy for the ItePated prisoner's Dilemma with noise. *Theory and Decision, 35,* 107-150.

Kreps, D. (1990). *Game theory and economic modelling* (pp. 123-128). Oxford, UK: Oxford University Press.

Linster, B. (1992). Evolutionary stability in the repeated Prisoners' Dilemma played by two-state Moore machines. *Southern Economic Journal,* 880-903.

Nowak, M. A., & Sigmund, K. (1992). Tit-for-Tat in a heterogeneous population. *Nature, 355*(6357), 250-253.

Nowak, M. A., & Sigmund, K. (1993). A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game. *Nature, 364*(6432), 56-58.

Matos, N., Sierra, C., & Jennings, N. R. (1998). Determining successful negotiation strategies: An evolutionary approach. *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS-98)*.

Mitchell, M. (1998). *An introduction to genetic algorithms.* Cambridge, MA: MIT Press.

Muthoo, A. (1999). *Bargaining theory with applications.* Cambridge: Cambridge University Press.

Myerson, R. (1991). *Game theory: Analysis of conflict* (pp. 399-403). Cambridge, MA: Harvard University Press.

Rubinstein, A. (1982). Perfect equilibrium in a bargaining game. *Econometric, 50*(1), 97-110.

Rubinstein, A. (1985). A bargaining model with incomplete information about time preferences. *Econometric, 53*(5), 1151-1172.

Sandholm, T. (1999). Distributed rational decision making. In G. Weiss (Ed.), *Multi-agent systems: A modern approach to distributed artificial intelligence* (pp. 201+). Cambridge, MA: MIT Press.

Sandholm, T., & Vulkan, N. (2002). Bargaining with deadlines. *Proceedings of the National Conference on Artificial Intelligence (AAAI)* (pp. 44-51).

Sebag, M., & Ducoulombier, A. (1998). Extending population-based incremental learning to continuous search spaces. *Proceedings of the 5th Conference on Parallel Problems Solving from Nature* (pp. 418-427).

Sukthankar, R., Baluja, S., & Hancock, J. (1998). Multiple adaptive agents for tactical driving. *Applied Intelligence.*

Walsh, W. E. (2001). *Market protocols for decentralized supply chain formation.* Doctoral thesis, University of Michigan, USA.

Wellman, M. P., Greenwald, A., Stone, P., & Wurman, P. R. (2000). The 2001 trading agent competition. *Proceedings of the 14th Conference on Innovative Applications of Artificial Intelligence.*

Chapter XXXIX
Applications of Neural Networks in Supply Chain Management

Ioannis Minis
University of the Aegean, Greece

Nikolaos Ampazis
University of the Aegean, Greece

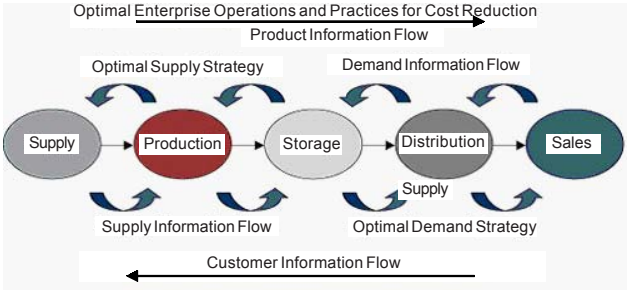
ABSTRACT

This chapter focuses on significant applications of self-organizing maps (SOMs), that is, unsupervised learning neural networks in two supply chain applications: cellular manufacturing and real-time management of a delayed delivery vehicle. Both problems require drastic complexity reduction, which is addressed effectively by clustering using SOMs. In the first problem, we cluster machines into cells and we use Latent Semantic Indexing for effective training of the network. In the second problem, we group the distribution sites into clusters based on their geographical location. The available vehicle time is distributed to each cluster by solving an appropriate non-linear optimization problem. Within each cluster an established orienteering heuristic is used to determine the clients to be served and the vehicle route. Extensive experimental results indicate that in terms of solution quality, our approach in general outperforms previously proposed methods. Furthermore, the proposed techniques are more efficient, especially in cases involving large numbers of data points. Neural networks have and will continue to play a significant role in solving effectively complex problems in supply chain applications, some of which are also highlighted in this chapter.

INTRODUCTION

The supply chain of both manufacturing and commercial enterprises comprises a highly distributed environment, in which complex processes evolve in a network of companies. Such processes include materials procurement and storage, production of intermediate and final products, warehousing, sales, and distribution (see Figure 1). The role of the supply chain in a company's competitiveness is critical, since the supply chain directly affects customer ser-

Figure 1. The flow of decisions and information in the supply chain



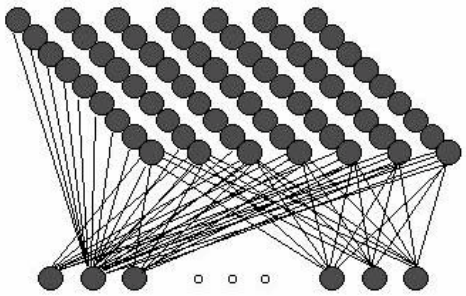
vice, inventory and distribution costs, and responsiveness to the ever-changing markets. Furthermore, this role becomes more critical in today's distributed manufacturing environment, in which companies focus on core competencies and outsource supportive tasks, thus creating large supply networks. Within this environment there are strong interactions of multiple entities, processes, and data. For each process in isolation, it is usually feasible to identify those decisions that are locally optimal, especially in a deterministic setting. However, decision making in supply chain systems should consider intrinsic uncertainties, while coordinating the interests and goals of the multitude of processes involved.

Nature-inspired computing offers effective tools for both modeling and managing operations in the uncertain environment of the supply chain, especially since the associated computational techniques are capable of handling complex interdependencies. As a result, these computational techniques may form the basis for the development of optimization methods and systems that integrate effectively the various objectives of the supply chain.

This chapter presents significant applications of artificial neural networks, a major technique of nature-inspired computing, in supply chain management. Specifically, we use self-organizing maps (SOMs) to reduce the complexity of problems in two supply chain applications: manufacturing shop design and real-time distribution management. Both share the requirement for drastic complexity reduction, which is addressed effectively by clustering using SOMs. Clustering is the problem of partitioning a set of N patterns in their feature space into K clusters, so that patterns that belong to a given cluster are more similar to each other than to the rest of the patterns. The majority of clustering algorithms are based on the minimization of the distance between each pattern and the centroid of each cluster. Neural networks have been used for clustering applications (Kamgar-Parsi, Gualtieri, & Devaney, 1990) and were shown to outperform conventional iterative techniques, such as the K-means algorithm (Jain & Dubes, 1988), in terms of both solution quality and speed when the clusters are well defined. In particular, SOMs have been shown to be very efficient clustering techniques in a variety of applications (Chen, Mangiameli, & West, 1995).

The remainder of this chapter is structured as follows. The second section overviews self-organizing maps. The third section presents an important clustering application, that is, the decomposition of a manufacturing shop in manufacturing cells. The fourth section presents another clustering application in distribution, a downstream process of the supply chain. The

Figure 2. A two-dimensional self-organizing map



fifth section discusses the extensions of the proposed concepts—clustering and neural networks—to other supply chain operations and processes.

SELF-ORGANIZING MAPS

SOMs are unsupervised learning neural networks, originally introduced by Kohonen (1989). This type of neural network is usually a two-dimensional lattice of neurons all of which have a reference model weight vector (see Figure 2). As a result of the SOM training algorithm, the reference vectors (otherwise known as codebook vectors) are fitted to a set of input vectors by approximating the model of the data distribution in the high-dimensional feature space. Therefore, the model vectors of neighboring units gradually learn to represent similar input data vectors.

SOMs are very well suited to organize and visualize complex data in a two-dimensional display, and by the same effect, to create abstractions or clusters of that data. The flexibility and visualization capabilities of a SOM make it an excellent tool for clustering applications. The projection of high-dimensional data points onto a 2-D neural map allows for the exploration and identification of non-trivial associations between these points, in a high-dimensional space. In addition, SOM is a computationally efficient clustering method.

The training of the SOM is achieved through a competitive learning process, which consists of two steps applied iteratively. In the first step each input vector is compared to all the neurons' codebook vectors. The neuron *s* that has its codebook vector at the shortest geometric distance to an input vector becomes the winner for that input vector. In the second step, each winning neuron and its surrounding neurons—that is, neurons within a neighborhood N_s—gradually change the value of their codebook vectors in an attempt to match the input vector which has won. This cycle of competition and learning processes is repeated. At each cycle the size of the neighborhood of the winning neuron is decreased. The whole process terminates when each codebook vector has reached a satisfactory approximation of its corresponding input vector.

The steps of the SOM algorithm can be summarized as follows:

- **Step 1:** Initialize
  - weights to small random values
  - neighborhood size $N_s(0)$ to be large (but less than the number of neurons in one dimension of the array)
  - parameter functions $a(t)$ and $\sigma^2(t)$ to be between 0 and 1
- **Step 2:** Present an input pattern *x* through the input layer and calculate the Euclidian distance between the input vector and each weight vector:

$$d_j(t) = \left\| x(t) - w_j(t) \right\| = \sqrt{\sum_{i=1}^{n} (x_i(t) - w_{ij}(t))^2} \quad (1)$$

- **Step 3:** Select the neuron with minimum distance to be the winner *s*
- **Step 4:** Update the weights connecting the input layer to the winning neuron and its neighboring neurons (neurons *k*) according to the learning rule

$$w_k\,(t+1) = w_k\,(t) + c[x(t) - w_k\,(t)] \quad (2)$$

where $c = a(t)\ exp\ (-\| r_i - r_s \|/\sigma^2\ (t))$ for all neurons *j* in $N_s\ (t)$, and $r_i - r_s$ is the physical distance (number of neurons) between neuron *i* and the winning neuron *s*

- **Step 5:** Continue from Step 2 for *T* epochs; then decrease neighborhood size, $a(t)$ and $\sigma^2(t)$: Repeat until weights have stabilized.

Kohonen (1989) proved that the SOM algorithm always converges to a solution—that is, each of the winner weight vectors of the map converges to the mean of the data vectors for which it has been a winner, in a finite number of steps.

MANUFACTURING CELL FORMATION

Background

The first application of supply chain management to be considered is related to production (or manufacturing) system design—that is, to the second link of the chain of Figure 1 (see also Ampazis & Minis, 2004). Manufacturing systems are notoriously complex to design and operate due to high dimensionality (e.g., thousands of make items and hundreds of work centers), multiple and dependent operations per make item, uncertain demand, as well as uncertainty in important system inputs and states (e.g., demand and work center failures, respectively). In an effort to reduce complexity, extensive research and implementation efforts have focused on cellular manufacturing, in which the production resources of a manufacturing system are arranged into manufacturing cells, and each cell been dedicated to the production of one or more part-families with similar production requirements. Immediate benefits of cellular manufacturing include: (a) significant reduction in material handling, achieved by confining much of the material handling effort within the cells; (b) reduction of set-up times and related costs, resulting from the production similarities of the parts within the same part family; (c) simplified shop floor planning, scheduling, and control, stemming from the decomposition of the shop to subsystems of reduced order (see Kusiak, 1987; Wemmerlov & Higher, 1989).

Extensive research has focused on the problem of disaggregating a manufacturing shop into cells over the last three decades. A comprehensive review and classification of the literature in this area can be found in King, Joines, and Culbreth, (1996). A significant class of methods transforms the part-machine incidence matrix of a shop. In this matrix each row represents a part and each column a machine; each entry is one or zero depending on whether the corresponding part uses the corresponding machine or not. The goal is to rearrange the rows and columns of this matrix in order to obtain a block diagonal form. The machines and parts corresponding to each block of the diagonal define the corresponding manufacturing cell and part-family. Notable transformation techniques include the bond energy algorithm (Schweitzer, McCormick, & White, 1972), rank order clustering (King, 1990), and the direct clustering algorithm (Milner & Chan, 1982). Other techniques aggregate the rows and columns of the part-machine incidence matrix based on similarity measured by various metrics (e.g., Seifoddini & Gupta, 1990). Clustering of

machines to cells and parts to part-families has also been performed by using artificial neural networks (Malave, Ramachandran, & Lee, 1992; Chen, Chang, Vohra, & Chen, 1990; Cheng & Chen, 1995; Kusiak & Chung, 1991), and simulated annealing (Narendran & Venugopal, 1992). Both linear and non-linear integer programming approaches have also been developed (Gunasingh & Kashkari, 1990; Boctor, 1991).

It is noted that beyond cell formation, neural networks have been used in various planning and control activities in manufacturing. Garetti and Taisch (1999), Smith (1999), and Smith and Gupta (2000) present extensive reviews that cover a broad spectrum of applications ranging from demand forecasting to shop floor scheduling and control.

Approach

Consider the shop's incidence matrix, each row of which corresponds to a make item (or part) and each column to a machine. The element p_{ij} = 1 if machine j is used to manufacture part i, and 0 otherwise. Thus, the dimension of the matrix is pxm, where p is the number of parts and m is the number of machines in the system. In realistic applications, p is in the order of 10^3 or 10^4, and m is in the order of 10^2. This high dimensionality results in increased computational complexity, irrespective of the clustering algorithm employed.

A way to reduce the dimensionality and thus to improve significantly the efficiency of the clustering algorithm is to project the data onto a lower-dimensional orthogonal subspace where most of the data variance is concentrated in the new subspace's axes. To do this we use latent semantic indexing (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990), which is based on the singular value decomposition (SVD) of the part-machine incidence matrix. latent semantic indexing (LSI) is well suited for sparse matrices, such as the part-machine incidence matrix, and preferable to other similar methods such as principal component analysis (PCA).

Latent Semantic Indexing

LSI constructs a linear mapping from the space spanned by the original machine vectors to a reduced dimensional subspace. This mapping is based on the SVD of the original pxm part-machine matrix A of m, p-dimensional, machine vectors:

$$A = U \sum V^T \quad (3)$$

where the orthogonal matrices U and V contain the left and right singular vectors of A and the diagonal matrix $\sum$ contains its singular values. LSI achieves reduction in the dimensionality of the data by retaining only the k-largest [($k < r$ = rank(A)] singular triplets of the decomposition of A, which means that all data vectors a^i (columns of A) are projected onto a k-dimensional subspace spanned by the left singular vectors corresponding to the k-largest singular values via the transformation

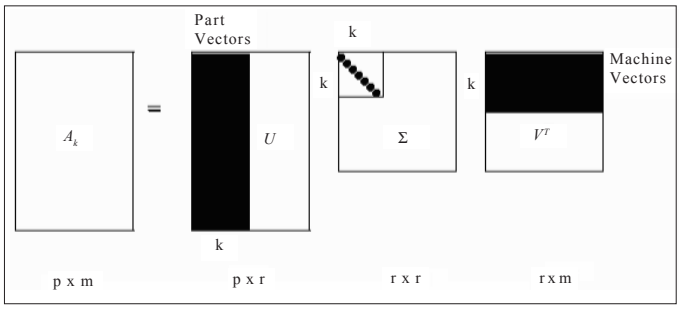
$$\hat{a^i} = (a^i)^T U_k \Sigma_k^{-1} \quad (4)$$

where U_k is of size pxk and contains these k singular vectors, and $\sum_k$ is of size kxk and contains the k largest singular values in its diagonal. In this sense the rows of V are considered as the LSI representations of the machine vectors, and by an analogous argument, the rows of matrix U_k are considered as the LSI representations of the part vectors (Figure 3).

The Proposed Algorithm

The results of the LSI processing of the part-machine incidence matrix are used as the input to the SOM algorithm of Section 2 to cluster machines into cells.

Figure 3. Singular Value Decomposition of the part-machine incidence matrix A



The steps of the proposed algorithm can be summarized as follows:

- **Step 1:** Utilize the Lanczos method (Berry, 1992) to obtain the SVD of the original sparse *pxm* part-machine incidence matrix *A*, keeping the *k* largest singular components
- **Step 2:** Store the *mxk* matrix V_k whose rows are the LSI representations of the original machine vectors
- **Step 3:** Use the rows of matrix V_k as input data vectors to a SOM of fixed topology in order to directly cluster the machines
- **Step 4:** Train the SOM until convergence and identify the corresponding machine cells by locating the best mathing unit (BMU) for each machine vector on the map
- **Step 5:** Evaluate the result of clustering by measuring the quality of clustering (QoC) criterion defined as:

$$QoC = \left(1 - \frac{\sum_{l=1}^{C} \sum_{i \in l,\, j \notin l} x_{ij}}{\sum_{l=1}^{C} \sum_{i \in l,\, j \in l} x_{ij}}\right) \qquad j \in l : \max\left(\sum_{i \in l} x_{ij}\right) \tag{5}$$

where *C* denotes the number of machine cells (clusters) and x_{ij} denotes the number of operations that machine *j* performs within machine cell *i*. The above equation is similar to the well-known Grouping Efficacy criterion (Kumar & Chandrasekharan, 1990), and calculates the percentage of operations performed outside any machine cell. It is clear that 0 <=*QoC*<= 1. In the ideal grouping *QoC*=1, decreasing values of *QoC* signify decreasing solution quality.

Experimental Investigation

We have evaluated the proposed method for three artificial problems of various sizes—small (100×22 part-machine incidence matrix), medium (576×54 matrix), and large (1152×108 matrix). All experiments were performed in the MATLAB programming language using the SOM MATLAB Toolbox.

We present here only the results of the large-scale problem which are representative of all three cases investigated. The shop considered in this problem consists of 18 mutually exclusive clusters. The aim of the experiment was to determine whether the proposed method finds the unique solution that exists. Figure 4 shows the *QoC* obtained with the proposed technique utilizing a one-dimensional *SOM* with 18 nodes and, as well as the *QoC* obtained by

Figure 4. Quality of clustering vs. LSI dimensions for the 1152×108 part-machine incidence matrix problem

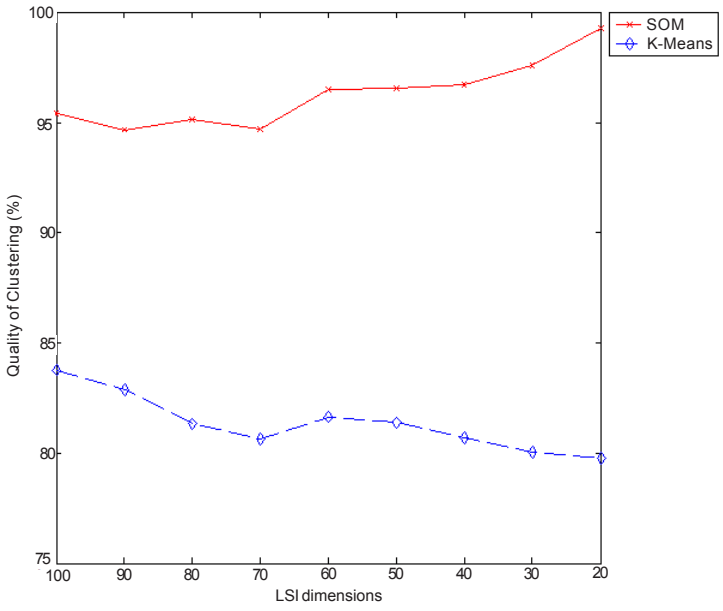


Figure 5. Training time (in seconds) vs. LSI dimensions for the 1152×108 part-machine incidence matrix problem

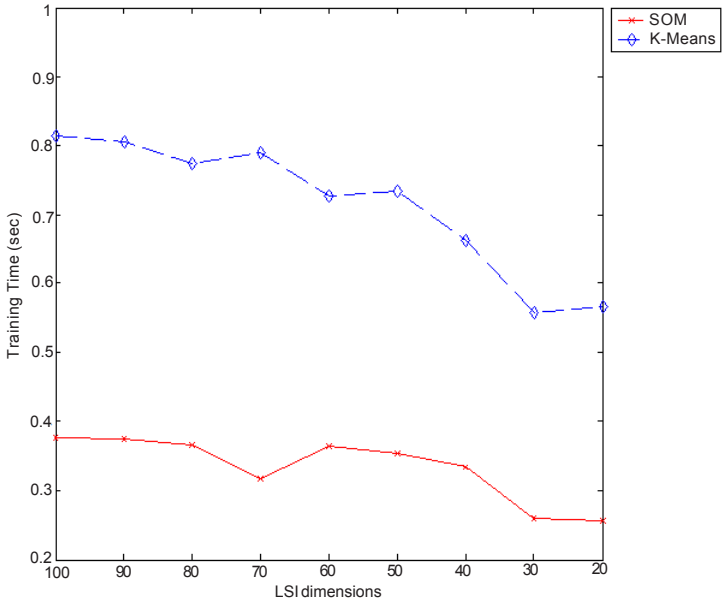
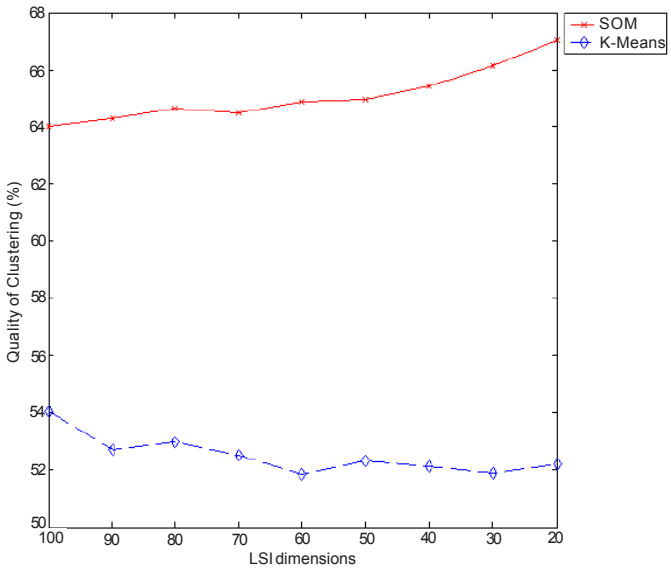


Figure 6. Quality of clustering vs. LSI dimensions for the 1152×108 part-machine incidence matrix problem, polluted with 5% random 1s



the k-means algorithm with k = 18, trained with the same data vectors. Each point in this figure has been obtained by performing 100 training trials and reporting the average *QoC* for both clustering methods at each LSI dimension shown. The results show that the *SOM* algorithm performs better than k-means. The reduction in the dimensionality of the LSI-salient features results in an increase to the attained *QoC* which, on average, tends to the optimal solution where *QoC* is maximum.

A further interesting point is related to computational complexity. It is well known that the computational cost of k-means for clustering n, d-dimensional data points is of order $\mathbf{O}(Tknd)$ where T is the number of iterations (Duda & Hart, 1973). The computational cost of SOM is of order $O(TM^2)$, where M is the number of nodes of the map (Kohonen, 1989). Therefore, it is expected that for this large-scale problem, SOM is computationally more efficient.

Figure 5 shows the average training time (in seconds) required for the clustering of each data point of Figure 4 illustrating the computational efficiency of SOM. In addition, as the number of LSI dimensions reduces, the average training time reduces (as expected). This fact, combined with the results shown in Figure 4, illustrates the overall reduction in complexity of the proposed method.

Figure 6 shows the results obtained when the original 1152×108 part-machine incidence matrix was corrupted with 5% random 1s (i.e., 5% of part operations performed outside a part's own cell). From this figure we can see that the QoC obtained by SOM is significantly better that the one achieved by k-means (all values reported are the average QoC for 100 training trials for each LSI dimension). Again we notice that the QoC of SOM tends to improve with decreasing LSI dimensions, which shows that the proposed methodology

retains its robustness with increasing problem size.

REAL-TIME DISTRIBUTION MANAGEMENT TO CLUSTERED CLIENTS

Background

The second application of supply chain management to be considered is related to products distribution, the fourth link of the chain of Figure 1. Distribution is a major downstream activity of supply chain execution operations and contributes significantly to total logistics costs (Ballou, 1999). Over the last four decades, considerable research has focused on route planning of vehicle fleets that deliver goods under a wide variety of practical conditions and constraints (see Toth & Vigo, 2002, for an extensive review of the vehicle routing problem). The solution of vehicle routing problems has been successfully addressed through the utilization of neural network architectures in Ghaziri (1996), Matsuyama (1991), Potvin and Robillard (1995), Vakhutinsky and Golden (1994), and Nygard, Jueli, and Kadaba (1990). However, during execution (of even a near-optimal distribution plan), delivery performance may deteriorate significantly due to traffic congestion, service delays, or vehicle breakdowns; thus, real-time response to these unforeseen disturbances is necessary. Recent advances in navigation and mobile telecommunications technologies offer the required infrastructure for the development of real-time vehicle management systems that enhance the performance of delivery fleets by adapting vehicle routes to respond effectively to such unforeseen events. As an example, consider the architecture proposed by Giaglis, Minis, Tatarakis, and Zeimpekis (2004) which enables re-planning of an original routing plan based on the state of the distribution system.

This section addresses a simple, yet important, case of real-time vehicle management to be handled by an intelligent system, such as the one described above. Consider a vehicle that executes a pre-defined distribution plan, which specifies an ordered set of clients to be served, a route, and a time horizon within which all deliveries should be completed. If during plan execution the vehicle experiences significant delays that do not allow completion of the plan within the acceptable time horizon, then a decision needs to be made concerning which clients to serve and which clients to notify for delivery postponement in order to optimize a selected metric (e.g., customer service, sales income, etc.). This single-vehicle re-planning problem has strong similarities with the so-called *orienteering* problem (OP), a generalization of the extensively studied traveling salesman problem (TSP). The first formulation of OP was proposed by Tsiligirides (1984), who also proposed two effective heuristic methods to solve it. For other OP solution methods, see Golden, Levy, and Vohra (1987), and Chao, Golden, and Wasil (1996). Exact solution approaches have been proposed by Laporte and Martello (1990), Ramesh, Yoon, and Karwan (1992), and Fischetti, Gonzales, and Toth (1998).

Certain differences exist, however, between OP and the single vehicle re-planning problem. The latter includes a significant service time requirement for each client which is not present in the OP. Furthermore, there are significant efficiency requirements that should be satisfied in a real-time implementation. The problem should be formulated and solved within a short time period, in order to respond effectively to undesirable changes of the system state. A key point to achieve is to effectively deal with problem complexity.

In many urban and suburban distribution networks, clients are naturally grouped into neighborhoods (and/or suburbs), called client clusters. Intra-cluster distances are short (e.g., 1-2 km) while inter-cluster distances are longer (5-10 km). Such an environment provides an opportunity to decrease the complexity of the problem by decomposing it into multiple smaller sub-problems, each corresponding to a single client cluster. Each sub-problem has a similar structure to the monolithic problem and can be addressed following an OP-like solution procedure. Since, however, the size of each sub-problem is considerably smaller than the size of the original problem, complexity and computational times are improved.

The Single Vehicle Re-Planning Model

Consider a network N comprising a set of vertices $V = \{0,1,...,n\}$ and a set of arcs A that interconnect these vertices. We associate a cost (travel time) c_{ij} to all arcs $x_{ij} \in A$. We also associate a service cost t_i and an income p_i to each vertex. The vertices in V represent the vehicle depot (0) and n clients to be served by the single vehicle, and the cost represents the time required to travel from vertex i to vertex $j, \forall i, j \in V$. The service cost is the time required to serve each client $i \in V \setminus \{0\}$, while the income p_i represents a business metric associated with client i, such as the estimated sales volume to this client or the client importance. A route is a sequence of arcs which the vehicle will follow, originating from the depot, to serve all n clients and return to the depot.

Consider a situation that the vehicle has already served some of the clients on its route (say the clients in $V_c = \{1,2,...,m\} \subset V$) and is on its way to the next client (say $m + 1$), when it is apparent that, due to delays, it is not possible for the vehicle to complete its route as planned within the available time T (a hard constraint), and thus, re-planning is necessary. In order to formulate the re-planning problem, consider a new network with $V_u = V \setminus V_c \cup \{s\}$ (where the vertex s represents the current position of the vehicle), an appropriate set of arcs A_u, and the related cost matrix and service time vector.

Let $y_i \in \{0,1\}$ be a binary variable, such that $y_i = 1$ if client $i \in V_u$ is served and $y_i = 0$ otherwise. Also let $x_{ij} \in \{0,1\}$ be another binary variable, such that $x_{ij} = 1$ if arc $x_{ij} \in A_u$ belongs to the new route and $x_{ij} = 0$ otherwise. Finally, let the objective be to optimize the total income resulting from serving the clients selected in the new plan. The mathematical program that models the single vehicle re-planning problem is given in Minis, Ampazis, and Mamassis (2005).

Solution Approach

A heuristic based on clustering using SOMs has been developed to solve this problem and consists of the following steps:

1. **Spatial Decomposition:** Group clients into clusters with respect to their geographical location.
2. **Time Decomposition:** Determine the appropriate time to serve each cluster.
3. **Solution of Sub-Problems:** For each cluster, solve an OP problem to determine the order of visiting the clients within the cluster; the objective is to maximize intra-cluster sales volume within the time allocated to the cluster by step 2.
4. **Improvement:** Allocate any remaining time to some of the remaining clients.

Each of these steps is described as follows.

Spatial Decomposition: Client Clustering

The SOM method described in the second section of this chapter has been used to cluster

clients into groups or neighborhoods based on the client's geographical position (i.e., its coordinates). It is noted that classical clustering algorithms, such as the k-means mentioned above, are also capable of handling similar problems. However, using these classical techniques, classes of similar objects are basically found by doing pair-wise comparisons among all the data elements. These clustering algorithms are serial in nature, in that pair-wise comparisons are made one at a time and the classification structure is created in a serial order.

The SOM neural network approach, on the other hand, addresses clustering and classification problems by means of a connectionist approach. Algorithms based upon neural networks are parallel in that multiple connections among the nodes allow for independent, parallel comparisons. As mentioned before, the SOM is an unsupervised technique for summarizing high-dimensional data so that similar inputs are, in general, mapped close to each other; advantages include improved quality of clustering, as well as improved efficiency, features that are both important in the present real-time application.

Time Decomposition

In this step, the available time horizon is divided into time intervals allocated to each cluster. This is performed in two stages. The first stage estimates the time required to travel from cluster to cluster, starting from the current position of the vehicle and finishing at the depot (inter-cluster travel time). This is a typical TSP problem formulated by considering the cluster centroids, as well as the starting and ending (depot) locations, and has been solved by the nearest neighbor heuristic followed by the 2-opt improvement procedure (see Lawler, Lenstra, Rinnooy, Kan, and Shmoys, 1985). The output of this step comprises all travel times τ_{kl} of the inter-cluster route.

The second stage allocates the available time horizon to the clusters formed by the decomposition procedure. To this end, an income function J_k (t) is developed for each cluster k = 1,2,..., c. Then an optimization problem is formed to maximize the total income with respect to the time τ_k that the vehicle spends within each cluster. Finally the entire time horizon available is distributed among the clusters.

Construction of the Income Function

Let the increasing function J_k (t) represent a measure of the income obtained by the vehicle while serving the clients of cluster k, as a function of time t. To obtain a "good" initial sequence of serving the clients within each cluster, a greedy ordering of clients in cluster k is performed, inspired by the OP method of Tsiligirides (1984). Starting from a random client within cluster k, construct a path by inserting clients iteratively based on a desirability measure. For every client i not yet included in the path, this measure is given by:

$$A_i = \left(\frac{p_i}{c_{i,last} + t_i} \right)^4 \qquad (6)$$

where p_i is the income obtained from this client, $c_{last,i}$ is the travel time from the last client of the path to client i, and t_i is the service time of client i. This is repeated until all clients of cluster k have been included in the path. For client i of the path: (a) the income ordinate of J_k (t) is determined by adding the income of all clients from the start up to client i, and (b) the time ordinate is determined by adding the travel and service times corresponding to these clients.

Determination of Intra-Cluster Time Intervals

A polynomial $I_k(t)$ of appropriate degree (≤ 3) is fitted to $J_k(t)$ in order to provide a smooth approximation of the income function per cluster. The coefficients of the fitting polynomial are evaluated using regression. The intra-cluster time τ_κ for each cluster is then obtained by solving the following non-linear constrained optimization problem.

$$max \sum_{k=1}^{c} I_k(\tau_\kappa) \quad (7)$$

$$\text{s.t.} \sum_{k=1}^{C} \tau_{ik} = \tau \quad (8)$$

$$\tau_k \geq 0 \ \ \forall k \quad (9)$$

where τ is the difference between the entire time horizon available and the total intra-cluster time estimated in the first stage above. This problem is solved iteratively by the Sequential Quadratic Programming method (SQP) (Gill, Murray, & Wright, 1999).

Distribution of Entire Time Horizon Among the Clusters

In order to form the OP sub-problems for each cluster, the time θ_κ available per cluster k is determined by:

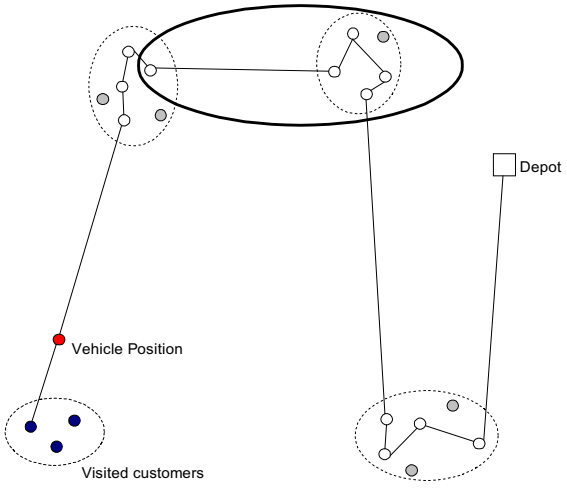
$$\theta_\kappa = \tau_\kappa + t_{\kappa-1,\kappa} \quad (10)$$

where τ_κ is the value determined from solving the optimization problem, and $t_{k-1,k}$ is the travel time between cluster $k-1$ and cluster k computed in the first stage above. Note that: (a) for the first cluster, point 0 is considered to be the initial location of the vehicle; and (b) for the last cluster in the route, the value of equation 10 is enhanced by adding the travel time between the centroid of this cluster to the depot (in order to complete the route).

Solution of Sub-Problems

For each of the client clusters, an OP-like problem is formulated and solved. For cluster k this problem is formulated by considering (see Figure 7): (a) as origin of the route of the last client served in cluster $k-1$ (for the first cluster, the origin is the initial location of the vehicle); (b) all clients of cluster k, the matrix of travel times, and the client income and service times; (c) the remaining time θ_κ; (d) to form the cost matrix for this problem, each travel time c_{ij} is augmented by the service time of client j, that is, each element of the matrix becomes $c_{ij} + \tau_j$; and (e) for the last cluster only, the destination is defined as the depot. The simple S-algorithm of Tsiligirides (1984) is used to solve the problem. The main concept of the S-algorithm is to construct a path starting from the origin and inserting clients iteratively until the available time is exhausted. For every client i not yet included in the path, the desirability of equation 6 is used. Considering the clients with

Figure 7. Structure of the OP sub-problem



the four highest values of A_i, the associated probability measures are computed from:

$$P_i = \frac{A_i}{\sum_{l=1}^{4} A_l}, \quad i = 1,...,4 \tag{11}$$

and the next point to be inserted in the path is selected randomly among the above four points with selection probability P_i. A large number of candidate paths are generated this way, and the solution with the highest income is selected as the preferred solution. The latter is improved by a 2-opt procedure, reducing the time spent to travel the route, and additional clients are inserted into the route taking advantage of the time savings.

Improvement

The time interval θ_κ allocated to each cluster k may not be exhausted, due to the fact that the time required to serve any client beyond the last one served in cluster k may be larger than the remaining time in this cluster. The improvement step examines whether the sum of the unused times per cluster can be exploited to serve one or more additional clients in any cluster(s). The iterative improvement procedure attempts to insert a client in the route of a cluster based on the desirability measure A_i of equation 6. For each candidate client, this measure is computed using the distance travel time from the last served client of the corresponding cluster. The client with the largest value of A_i is inserted in the route of its cluster, until the sum of unused times is exhausted.

Experimental Investigation

The algorithm presented in the section above has been implemented in MATLAB. It has been verified using Tsiligirides' standard problem instances and a single client cluster. To test the effectiveness of the proposed method, we have applied it to both numerical and practical problems.

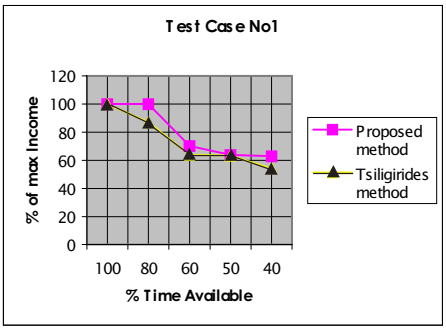
Numerical Test Cases

Four problem instances (cases) have been generated using a procedure that allocates 99 clients within an area of 20x20 km^2. The customer allocation is such that clusters of varying compactness are formed, ranging from distinct to significantly overlapping client clusters—that is, in case 1, each cluster covers a 1x1 Km^2 area, in case 2 it covers a 2.5x2.5 Km^2 area, in case 3 a 5x5 Km^2 area, and in case 4 a 10x10 Km^2 area. To quantify the quality of clustering, the Davies-Bouldin index, Q, has been used. The income and service times per client have been generated, again at random based on uniform distributions. For comparison purposes, all cases have also been solved using the original Tsiligirides approach (without clustering).

Each problem instance has been solved using decreasing values of the time available, from 100% of the time required to serve all clients to 40% of this value. Both the Tsiligirides algorithm and the OP part of the proposed algorithm ran for 100 iterations. In order to compare the two methods, the entire time horizon has been considered available to serve clients without requiring the vehicle to return to the depot, respecting the original formulation of the OP.

Figures 8 and 9 present the results obtained. From these figures the following can be concluded regarding the objective function value (income): for "tight" client clusters (Q = 0.08 – 0.67), the proposed algorithm outperforms the Tsiligirides' algorithm in the entire range of fractions of time available. This trend holds for all time fractions above 50% for moderate to inferior cluster quality. Only for case 4, in

Figure 8. Comparison of the proposed method vs. the Tsiligirides method (which does not use clustering) for cases 1 (Q = 0.08) and 2 (Q =0.40)



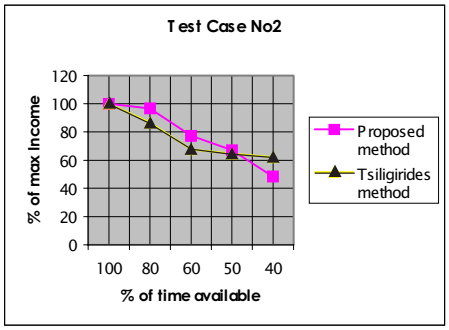
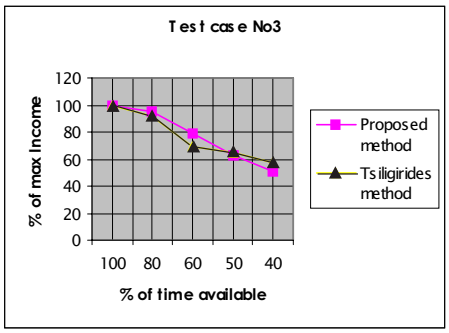
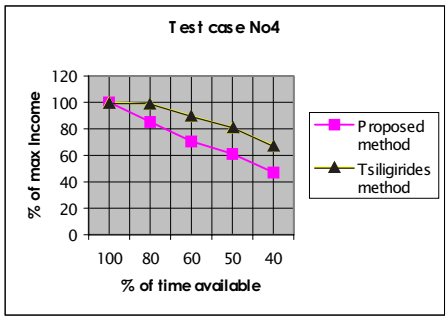


Figure 9. Comparison of the proposed method vs. the Tsiligirides method (which does not use clustering) for cases 3 (Q = 0.67) and 4 (Q = 0.98)





which there are no apparent clusters ($Q \approx 1$ and $\alpha \times \alpha = 10 \times 10$ km^2 in an total area of 20 x 20 km^2), the income results of the proposed method are inferior to those of the Tsiligirides method.

As far as computational time is concerned, for all cases the proposed method is significantly more efficient than the Tsiligirides method, due to the lower complexity involved. Specifically, in all test cases the proposed method was found to be two to three times faster. In fact, these time savings will grow when an increased number of iterations is used in the OP problem solution, as recommended by Tsiligirides. Overall, the method is suitable for real-time applications; the average computational time for the four cases in a 2.6 GHz PC is in the order of 30 seconds.

Practical Distribution Cases

Two practical cases from a Greek food manufacturer have been used as additional tests of the method's effectiveness. The first case is a route of 18 clients grouped into two clusters distributed in the greater Athens area. The second case is a route of 46 clients grouped into two clusters within the same area. The results are presented in Table 1.

In terms of solution quality, conclusions similar to those obtained from the test cases

Table 1. Experimental results for practical cases (Case 1: n = 18 clients, Q = 0.77; Case 2: n = 46 clients, Q = 1.84)

	CASE 1		CASE 2	
% Time Available	% of Max. Income (Proposed Method)	% of Max. Income (Tsiligirides Method)	% of Max. Income (Proposed Method)	% of Max. Income (Tsiligirides Method)
100	100	100	100	100
80	96.55	96.55	87.18	84.62
60	58.62	55.17	73.08	60.26
50	58.62	39.66	67.95	7.69
40	58.62	6.90		

can be deduced. It is emphasized, that the computational time in these cases is in the order of one to two seconds in a PC with a x86 family processor at 2.6 GHz, verifying the suitability of the method for real-time applications.

The results obtained by applying the proposed method to other large test cases, as well as to problem instances of actual distribution networks, indicate that in terms of solution quality, the method in general: (a) outperforms the methods used to solve the monolithic problem, even in environments with inferior client clustering; and (b) is more efficient, especially in cases involving a large number of clients.

NEURAL NETWORKS AND CLUSTERING IN OTHER SUPPLY CHAIN APPLICATIONS

In addition to the applications presented above, neural networks have been used in a variety of supply chain problems, especially in operations management (e.g., Garetti et al., 1999). Characteristic examples include: (a) the notoriously complex scheduling problem, as in Padman (1993), or in Gupta and Tunc (1997) and Gupta, Sexton, and Tunc (1999), in which neural networks have been used for selecting the most appropriate scheduling heuristic; (b) manufacturing system design (Mollaghasemi, LeCroy, & Georgiopoulos, 1998), in which neural networks have been used in conjunction with simulation modeling; and (c) quality control, where neural networks have been integrated with traditional statistical techniques to enhance their performance (Glover, 1988).

A final (and quite different) example that illustrates the significant potential of self-organizing maps in supply chain applications is sketched below. It concerns the development of standard part descriptions in the item master records of ERP systems. This data management problem is of great practical importance, especially in manufacturing companies with long history and legacy IT systems. Upgrading from such legacy systems to a modern ERP requires, in addition to migrating thousands of existing item master records, the development of new part classification information, in order to be able to intelligently manage the part data for a variety of applications, including procurement, B2B, use of alternative items in product bills of materials (BOMs), and so forth. Instead of using special codes for group identification and classification, natural language descriptions are far more appropriate. The difficulty, though, is to automatically generate standard descriptions (i.e., descriptions that use a predefined set of standard, natural language words in a certain structured way) based on existing (legacy) part descriptions provided in unstruc-

tured text and other data stored in the item master record.

SOMs have the ability to arrange text descriptions with similar content in neighboring regions, which, by analogy, is comparable to the situation encountered in conventional libraries where books are organized in thematic topics. Such an arrangement, combined with traditional information retrieval search tools and facilities, can help system users search for a part with a certain description and other characteristics, and retrieve item master records of parts that fit these specifications. Having identified such part clusters, appropriate standard descriptions are related to each part in the cluster (obviously, one part may belong to more than one cluster). The LSISOM method introduced recently in Ampazis and Perantonis (2004) is a technique that effectively represents text-by-word category (concepts) histograms. These histograms are formed by the SOM clustering of the Latent Semantic Indexing representation of distinct terms in a passage of text. Thus, item master records that contain text with similar concepts can be effectively grouped in neighboring regions of a second SOM trained with these new text descriptions. When this method is applied to a collection of text descriptions of parts, in effect, parts that share similar descriptions are automatically clustered. This technique, combined with a traditional numerical data clustering algorithm, operating on the alphanumerical attributes of parts, may create an integrated, fully automatic clustering system of parts based on both text description and other part information contained in the item master record. Research in this area is already underway.

CONCLUSION

We have presented applications of artificial neural networks in important supply chain applications. More specifically, we examined the cases of manufacturing shop design and real-time distribution management. Both applications share the requirement for drastic complexity reduction, which is addressed effectively by clustering using SOMs. In the first problem we have used SOMs to perform direct clustering of machines into cells, without first resorting to grouping parts into families. We also employed LSI to reduce the complexity of the problem resulting in more effective training of the network.

In the second problem, we deal with real-time management of a delayed delivery vehicle. In this case, the problem is decomposed into smaller sub-problems, taking advantage of the geographical distribution of clients into clusters (neighborhoods and/or suburbs). The available vehicle time is distributed to each cluster (identified by a SOM) by solving an appropriate non-linear optimization problem. Within each cluster an established OP heuristic is used to determine the clients to be served and the vehicle route.

The efficiency of all proposed algorithms was verified by experimental results. The results obtained by applying the proposed methods to large test cases, as well as to problem instances of actual distribution networks (for the real-time distribution management problem), indicate that in terms of solution quality, the proposed methods in general outperform previously proposed methods. Furthermore, the methods are more efficient, especially in cases involving a large number of data points. Finally, computational times are short, indicating the suitability of the methods that utilize SOMs.

Neural networks have and will continue to play a significant role in effectively solving complex problems in supply chain applications, some of which have been highlighted in this chapter.

REFERENCES

Ampazis, N., & Minis, I. (2004). Design of cellular manufacturing systems using latent semantic indexing and self-organizing maps. *Computational Management Science, 1*(3-4), 275-292.

Ampazis, N., & Perantonis, S. J. (2004). LSISOM—A Latent Semantic Indexing approach to self-organizing maps of document collections. *Neural Processing Letters, 19*(2), 157-217.

Ballou, R. H. (1999). *Business logistics management* (4th international ed.). Upper Saddle River, NJ: Prentice-Hall.

Berry, M. W. (1992). Large-scale singular value computations. *International Journal of Supercomputer Applications, 6*(1), 13-49.

Boctor, F. (1991). A linear formulation of the machine-part formation problem. *International Journal of Production Research, 29*(2), 343-356.

Chao, I. M., Golden, B. L., & Wasil, E. A. (1996). A fast and effective heuristic for the orienteering problem. *European Journal of Operation Research, 88,* 475-489.

Chen, D., Chang, J., Vohra, T., & Chen, H. (1990). A network approach to cell formation in group technology: Mace. *International Journal of Production Research, 28*(11), 2075-2084.

Chen, S. K. Mangiameli, P., & West, D. (1995). The comparative ability of self-organizing neural networks to define cluster structure. *Omega, 23,* 271-279.

Cheng, C. S., & Chen, S. J. (1995). A neural network-based cell formation algorithm in cellular manufacturing. *International Journal of Production Research, 33*(2), 293-318.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391-407.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis.* New York: John Wiley & Sons.

Fischetti, M., Gonzales, J. J. S., & Toth, P. (1998). Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing, 10*(2), 133-148.

Garetti, M., & Taisch, M. (1999). Neural networks in production planning and control. *Production Planning and Control, 10*(4), 324-339.

Ghaziri, H. (1996). Supervision in the self-organizing feature map: Application to the vehicle routing problem. In I. H. Osman & J. P. Kelly (Eds.), *Metaheuristics: Theory and applications* (pp. 651-660). Boston: Kluwer.

Giaglis, G. M., Minis, I., Tatarakis, A., & Zeimpekis, V. (2004). Minimizing logistics risk through real-time vehicle routing and mobile technologies: Research to-date and future trends. *International Journal of Physical Distribution and Logistics Management, 34*(9), 749-764.

Gill, P. E, Murray, W., & Wright, M. H. (1999). *Practical optimization.* San Diego: Academic Press.

Glover, D. E. (1988). Neural nets in automated inspection. *The Digest of Neural Computing, 2*(1), 17.

Golden, B. L., Levy, L., & Vohra, R. (1987). The orienteering problem. *Naval Research Logistics, 34,* 307-318.

Gunasingh, R., & Kashkari, K. (1990). Simultaneous grouping of parts and machines in cellular manufacturing systems an integer program-

ming approach. *Computers and Industrial Engineering, 20*(1), 111-117.

Gupta, J. N. D., Sexton, R. S., & Tunc, E. A. (1999). Selecting scheduling heuristic through neural networks. *INFORMS Journal of Computing, 12*(2), 150-162.

Gupta, J. N. D., & Tunc, E. A. (1997). Neural network approach to select scheduling heuristics for two-stage hybrid workshop. *International Journal of Management and Systems, 13*(283), 98.

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering*. Englewood Cliffs, NJ: Prentice-Hall.

Kamgar-Parsi, B., Gualtieri, J. A., & Devaney, J. E. (1990). Clustering with neural networks. *Biological Cybernetics, 63,* 201-208.

King, J. R. (1990). Machine-component grouping in production flow analysis: An approach using a rank order clustering algorithm. *International Journal of Production Research, 18*(2), 213-232.

King, R. A., Joines, J. A., & Culbreth, C. T. (1996). *A comprehensive review of production-oriented manufacturing cell formation techniques*. Technical Report, Furniture Manufacturing and Management Center, Department of Industrial Engineering, North Carolina State University, USA.

Kohonen, T. (1989). *Self-organization and associative memory* (3rd ed.). New York: Springer-Verlag.

Kumar, K. R., & Chandrasekharan, M. P. (1990). Grouping efficacy: A quantitative criterion for goodness if block diagonal forms if binary matrices in group technology. *International Journal of Production Research, 28*(2), 233-243.

Kusiak, A. (1987). The generalized group technology concept. *International Journal of Production Research, 25*(4), 561-569.

Kusiak, A., & Chung, Y. (1991). Gt/art: Using neural networks to form machine cells. *Manufacturing Review, 4,* 293-301.

Laporte, G., & Martello, S. (1990). The selective traveling salesman problem. *Discrete Applied Mathematics, 26,* 193-207.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (1985). *The traveling salesman problem*. New York: John Wiley & Sons.

Malave, C. O., Ramachandran, S., & Lee, H. (1992). A self-organizing neural network approach for the design of cellular manufacturing systems. *Journal of Intelligent Manufacturing, 3,* 325-333.

Matsuyama, Y. (1991). Self-organization via competition, cooperation and categorization applied to extended vehicle routing problems. *Proceedings of the International Joint Conference on Neural Networks,* Seattle (Vol. 1, pp. 385-390).

Milner, D. A., & Chan, H. N. (1982). Direct clustering algorithm for group formation in cellular manufacture. *Journal of Manufacturing Systems, 1*(1), 65-74.

Minis, I., Ampazis, N., & Mamassis, K. (2005). Efficient real-time management of goods distribution to clustered clients. *IJISM.*

Mollaghasemi, M., LeCroy, K., & Georgiopoulos, M. (1998). Application of neural networks and simulation modeling in manufacturing systems design. *Interfaces, 28*(100), 14.

Narendran, T. T., & Venugopal, V. (1992). Cell formation in manufacturing systems through

simulated annealing: An experimental evaluation. *European Journal of Operations Research, 63*(3), 409-422.

Nygard, K. E., Jueli, P., & Kadaba, N. (1990). Neural networks for selecting vehicle routing heuristics. *ORSA Journal of Computing, 2,* 353-364.

Padman, R. (1993). Choosing solvers in decision support systems. Neural network application in resource-constrained project scheduling. In *Recent developments in decision support systems* (pp. 559-574). Berlin: Springer-Verlag.

Potvin, J. Y., & Robillard, C. (1995). Clustering for vehicle routing with a competitive neural network. *Neurocomputing, 8,* 125-139.

Ramesh, R., Yoon, Y. S., & Karwan, M. H. (1992). An optimal algorithm for the orienteering tour problem. *ORSA Journal on Computing, 4*(2), 155-165.

Schweitzer, P. J., McCormick, W. T., & White, T. W. (1972). Problem decomposition and data reorganization by a cluster technique. *Operations Research, 20*(5), 993-1009.

Seifoddini, H., & Gupta, T. (1990). Production data based similarity coefficient for machine-component grouping decisions in the design of a cellular manufacturing system. *Computers and Industrial Engineering, 19*(1), 432-435.

Smith, K. A. (1999). Neural networks for combinatorial optimization: A review of more than a decade of research. *INFORMS Journal on Computing, 11*(1), 15-34.

Smith, K. A., & Gupta, J. N. D. (2000). Neural networks in business: Techniques and applications for the operations researcher. *Computers & Operations Research, 27,* 1023-1044.

Toth, P., & Vigo, D. (2002). *The vehicle routing problem.* Philadelphia: Siam.

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *Journal of Operational Research Society, 35*(9), 797-809.

Vakhutinsky, A. I., & Golden, B. L. (1994). Solving vehicle routing problems using elastic nets. *Proceedings of the IEEE International Conference on Neural Networks* (Vol. 7, pp. 4535-4540).

Wemmerlov, U., & Higher, N. L. (1989). Cellular manufacturing in the U.S. industry: A survey of users. *International Journal of Production Research, 27*(9), 1511-1530.

Chapter XL
An Object-Oriented Framework for Rapid Genetic Algorithm Development

Andrés L. Medaglia
Universidad de los Andes, Colombia

Eliécer Gutiérrez
Universidad de los Andes, Colombia

ABSTRACT

JGA, the acronym for Java Genetic Algorithm, is a computational object-oriented framework for rapid development of evolutionary algorithms for solving complex optimization problems. This chapter describes the JGA framework and illustrates its use on the dynamic inventory lot-sizing problem. Using this problem as benchmark, JGA is compared against three other tools, namely, GAlib, an open C++ implementation; GADS, a commercial Matlab® toolbox; and PROC GA, a commercial (yet experimental) SAS® procedure. JGA has proved to be a flexible and extensible object-oriented framework for the fast development of single (and multi-objective) genetic algorithms by providing a collection of ready-to-use modules (Java classes) that comprise the nucleus of any genetic algorithm. Furthermore, JGA has also been designed to be embedded in larger applications that solve complex business problems.

INTRODUCTION

Since the conception of genetic algorithms (GAs), researchers and practitioners alike faced the problem of building tools which could make the implementation of their own applications easier. At the beginning, the most widely used guide was the "Simple GA code" (SGA) implementation from Goldberg (1989) built in the Pascal programming language. Today, there is a broad array of offerings of genetic algorithm libraries available in different languages and computing platforms. For a thorough survey on the subject, the reader is referred to Pain and Reeves (2002).

Some of the earliest tools were coded in the C language. C evolved into C++, adopting the object-oriented programming (OOP) paradigm.

The advantage of using the C or C++ language is mainly its computer efficiency and outstanding performance in terms of speed. In this class, we found tools such as ECGA (Lobo & Harik, 1999), GALOPPS (Goodman, 2002), and GALib (Wall, 2005).

ECGA (Extended Compact GA) is a tool implemented in C++, based on the GA code from Goldberg, in which the user can replace or modify the core classes. This tool does not include templates or heritage concepts, but the class structure provides independency to every basic component of a genetic algorithm.

GALOPPS is a flexible generic GA implementation in C. To make it easier for users to learn and extend, it was also based upon Goldberg's SGA architecture. GALOPPS includes most of the chromosomal structures and operators described by Goldberg (1989). The latest version includes a parallel architecture mode in which each process can handle one or several interacting subpopulations.

GAlib contains a set of C++ classes for building genetic algorithms. GAlib provides a set of built-in representations and operators that could be extended and customized. The built-in chromosomes include binary, integer, and real arrays of variable length. It also includes more complex data structures such as lists and trees. Chromosome initialization, mutation, crossover, and comparison methods can be customized by deriving classes for the specific problem on hand. For simple applications, the user only needs to override the fitness function class. Some details need to be taken into account depending on the development platform.

Other genetic algorithm tools have been developed for commercial vendors such as Matlab and SAS. Special-purpose libraries in Matlab are known as toolboxes. These toolboxes define in Matlab's m-files the genetic components such as fitness functions, selection operators, and crossover and mutation operators. In the m-files, functions are defined taking advantage of Matlab's powerful language and basic types such as vectors and matrices. The main advantage of Matlab-based tools is the integration with the development environment, and the numerical and graphical power provided by the Matlab core engine and enhancing toolboxes. The commercial tools have a more complete set of built-in options, including multiple populations, migration and reinsertion operators, and multi-objective ranking of objective values. Some of the commercial Matlab toolboxes available are GEATbx (Pohlheim, 2005) and GADS (Mathworks, 2005). On the other hand, there are also publicly available toolboxes such as GPLAB (Silva, 2005), GAOT (Houck, Joines, & Kay, 1995), and GATbx (Chipperfield & Fleming, 1995). Another commercial vendor, the SAS Institute, released PROC GA, an experimental procedure integrated to the SAS system (SAS Institute, 2003).

Genetic algorithm tools are also available for Microsoft Excel. Some examples are the commercial tools GeneHunter (Ward Systems Group, 2003) and Evolver (Palisade Corp., 2005). The model and the objective function are specified by referencing specific cells in the worksheet. Both tools have options to redefine some components by means of Visual Basic programming or dynamic linked libraries (DLLs). These tools can also be accessed from other programs through their companion DLLs.

The Java Programming Language has had a fast penetration in the software and hardware market over the last decade. Some Java-based tools for genetic algorithms have been implemented. These packages follow many of the principles of the previously mentioned tools for C++. Two of these implementations are GGAT (Derderian, 2002) and JGAP (Rotstan & Meffert, 2005).

GGAT is a tool developed in Java with the philosophy of providing an open specification that allows users to create new components

such as chromosomes and operators. The current version solely provides structures based on binary digits, with built-in functions for decoding them into integer and real numbers. GGAT also includes several options for establishing the stopping criterion of the genetic algorithm. A distinctive feature of GGAT is its emphasis on the graphical interface for visualizing the evolutionary process.

JGAP is a Java package that provides basic genetic mechanisms that can be easily used to apply evolutionary principles to specific problems. The package structure is highly modular so that power users can easily plug in custom genetic operators and other components. JGAP provides a small core of built-in genetic operators, but it offers a great flexibility to construct new operators.

The purpose of this chapter is to describe a flexible and extensible computational object-oriented tool for rapid prototyping and implementation of evolutionary algorithms. We call this tool JGA, an acronym for Java Genetic Algorithm framework. According to our own experience, JGA has proved to be a flexible and extensible object-oriented framework for the fast development of single (and multi-objective) genetic algorithms. JGA has also been designed to be embedded in larger applications that solve complex business problems.

JGA ARCHITECTURE

JGA is a flexible and extensible computational object-oriented framework for rapid prototyping and implementation of evolutionary algorithms for solving complex optimization problems. We say that JGA shortens the development phase because the user of JGA does not have to implement the underlying logic of a genetic algorithm, but just has to focus on the unique aspects of the genetic algorithm dependent on its specific problem and field of application.

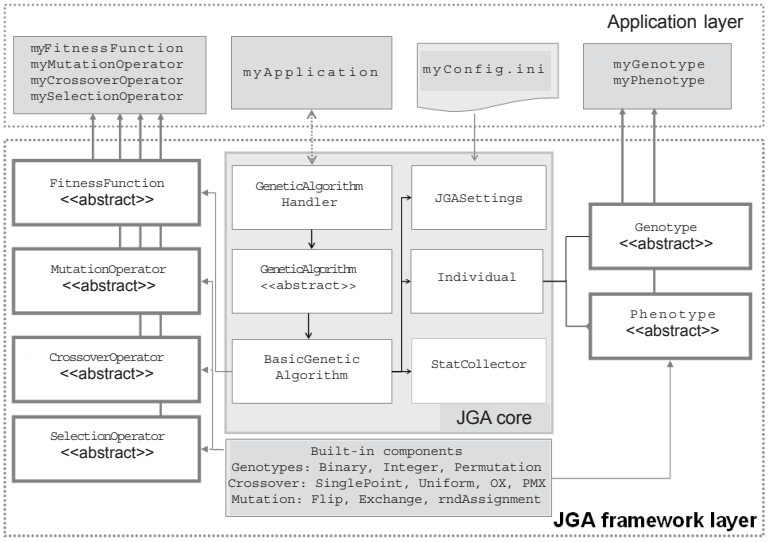
JGA provides the user with a collection of ready-to-use modules (classes) that comprises the nucleus of any genetic algorithm. JGA also provides a set of utility classes that allows for the rapid development of an evolutionary-based solution. Because JGA has been built using the Java language, it conforms to the object-oriented paradigm and offers, among other key advantages, flexibility, extensibility, and portability across different computing platforms. From an architectural point of view, JGA allows the user to extend existing elements of a genetic algorithm, such as genotypes, phenotypes, fitness functions, selection mechanisms, mutation, and crossover operators, among others. To foster flexibility and efficiency, the library or user-defined objects are loaded dynamically as needed by the Java Virtual Machine (JVM).

Figure 1 shows a diagram that illustrates the architecture of JGA. In the bottom layer, we show in boxes with bold lines the elementary components of every genetic algorithm (i.e., genotype, phenotype, fitness function, selection operator, mutation, and crossover operators). Inside the gray box, we show a set of classes and handlers that constitute the core. The JGA core is responsible for the fundamental (yet extensible) logic of the genetic algorithm. The built-in components are utility classes that implement some standard genetic algorithm operators and representation schemes found in the literature. The top layer, called the application layer, is the one that needs to be programmed by the user of a given application. If needed, the user could extend any of the existing elements and implement an application-specific component, such as the PMX crossover operator for routing problems (Michalewicz, 1996).

How JGA Works

The key component of JGA is the GeneticAlgorithmHandler. This class is respon-

Figure 1. JGA architecture



sible for loading the *main configuration file* and loading the different components of the genetic algorithm specified therein. An object of this class must be created in the main program, and the user must invoke its run method to trigger the execution of the genetic algorithm as illustrated in Figure 2.

The Genetic Algorithm Logic

The GeneticAlgorithmHandler loads the class with the logic of the genetic algorithm specified in the main configuration file (property field GENETICALGORITHM). Among the genetic algorithm community, there is no general agreement of a unique implementation of the logic of the algorithm. Some authors prefer to do mutation first, followed by the crossover operator; others allow only the children of a current population to mutate, and so forth. Therefore, JGA allows the user to code different logics by extending the abstract class *GeneticAlgorithm*. A particular implementation of a genetic algorithm is BasicGeneticAlgorithm, whose logic is presented in Figure 3, where t is the generation counter; T is the maximum number of generations; $P(t)$ is the population at generation t; $C_m(t)$ and $C_c(t)$ are the children populations obtained by the mutation and crossover operators, respectively; $C(t)$ is the children popula-

Figure 2. Example of the main program in Java of a sample JGA application

```
1: import edu.uniandes.copa.jga.*;
2: public class myApplication {
3: public static voic main (String args []){
4:    String configFileName = args [0];
5:    ArrayList genotypeParams = new ArrayList ();
6:    // genotypeParams construction...
7:    GeneticAlgorithmHandler ga = new Genetic AlgorithmHandler (configFileName, genotypeParams);
8:    ga.run();
9:  {
10:{
```

Figure 3. The logic behind BasicGeneticAlgorithm

1: $t \leftarrow 1$
2: initialize $\mathcal{P}(t)$
3: evaluate $\mathcal{P}(t)$
4: **while** $t \leq T$ **do**
5: mutate $\mathcal{P}(t)$ and generate $\mathcal{C}_m(t)$
6: cross $\mathcal{P}(t)$ and generate $\mathcal{C}_c(t)$
7: $\mathcal{C}(t) \leftarrow \mathcal{C}_m(t) \cup \mathcal{C}_c(t)$
8: evaluate $\mathcal{C}(t)$
9: $\mathcal{E}(t) \leftarrow \mathcal{P}(t) \cup \mathcal{C}(t)$
10: select $\mathcal{P}(t+1)$ from $\mathcal{E}(t)$
11: $t \leftarrow t+1$
12: **end while**

tion; and *E(t)* is the expanded population formed by the current population and their children.

JGA has also been designed to support multi-objective evolutionary algorithms. Such an extension of *GeneticAlgorithm* is presented in Chapter 42 of this book and shows how a multi-objective evolutionary algorithm, known as NSGA II (Deb, Pratap, Agarwal, & Meyarivan, 2002) is implemented in JGA to solve the strategic facility location problem arising in the context of logistics.

The Configuration Files

Every JGA implementation uses two configuration files: the *main* and the *problem-specific* configuration files. The *main configuration file* contains several property fields that are responsible for parameterizing the genetic algorithm and setting up the computational experiment. Even though these properties can be set in a single file, we can organize them into two categories: *structural* and *parametric* properties. The structural properties are those that point to the stored logic of a genetic algorithm component. These components are: the main logic of the algorithm, the individual's genotype and phenotype, the mutation and crossover operators, the fitness function, the selection mechanism, and the problem-specific configuration file. Table 1 shows the structural property fields along with their description and sample values. On the other hand, the parametric property fields are those parameters common to most genetic algorithms (i.e., population size, number of generations, mutation and crossover probabilities) and some others controlling output verbosity. Table 2 shows the parametric property fields along with their description and sample values. As the name suggests, the *problem-specific* configuration file is tied to a given application and provides necessary information that defines the problem instance. For example, in an inventory problem, the demand values along the planning horizon are an input that must be provided via the problem-specific configuration file.

Table 1. Structural property fields in the main configuration file

Property Field	Description	Sample Value
GENETICALGORITHM	Sets the class name with the main logic of the genetic algorithm.	edu.uniandes.copa.jga.BasicGeneticAlgorithm
GENOTYPE	Sets the class name that defines how a solution is coded into a genotype.	edu.uniandes.copa.jga.BinaryGenotype
PHENOTYPE	Sets the class name that defines and stores the individual's phenotype.	edu.uniandes.copa.jga.SingleFitnessPhenotype
FITNESSFCTN	Sets the class name with the implementation of the fitness function evaluation. Usually this is a problem-specific function that knows how to evaluate a specific genotype.	CVRPEvalFunction
MUTATION	Sets the class name with the implementation of the mutation operator.	edu.uniandes.copa.jga.FlipBinaryMutation
CROSSOVER	Sets the class name with the implementation of the crossover operator.	edu.uniandes.copa.jga.SinglePointRealCrossover
SELECTION	Sets the class name with the implementation of the selection mechanism.	edu.uniandes.copa.jga.RouletteWheelSelection
PROBLEMDATASETTINGS	Sets the file name with the problem-specific configuration file.	CVRPSettings.ini

Table 2. Parametric property fields and sample values in the main configuration file

Property Field	Description	Sample Value
POPSIZE	Sets the population size. Its value is an integer greater than or equal to 1.	100
MAXGEN	Sets the maximum number of generations. Its value is an integer greater than or equal to 1.	1000
MUTRATE	Sets the probability of mutation. Its use depends on the implementation of the mutation operator or the logic of the algorithm (extension of *GeneticAlgorithm*). It could be the probability that a given gene mutates or the probability that a whole individual of the population mutates. Its value is a real number between 0 and 1.	0.1
CROSSRATE	Sets the probability of crossover. Its use depends on the logic of the algorithm (extension of *GeneticAlgorithm*). For instance, in BasicGeneticAlgorithm, it is the probability that a given individual of the population is chosen as a parent, forming the so-called crossover pool. Its value is a real number between 0 and 1.	0.5
SEED	Sets the seed for the random number generator. It characterizes the execution of an independent run of the genetic algorithm. Its value is a non-negative integer.	1
CRITERIA	Sets the optimization criteria for the objective. Its value may be either MIN or MAX, for minimization or maximization, respectively.	MIN
STATISTICSCOLLECTOR	Sets the type of statistics to be collected during the genetic algorithm execution. The acceptable values for this property field will be explained later.	0
OUTPUTLEVEL	Sets the verbosity level for output messages. Depending on the type of messages to be displayed, its value is the result of the sum of the following values: 2^0=1, for warning messages; 2^1=2, for trace messages; 2^2=4, for debug messages; and 0, for silent output. For example, a property value of 5 (=1+4) turns on warning and debug messages.	0

Encoding and Decoding Solutions

In genetic algorithms, a solution must be encoded into an individual or chromosome. Correspondingly, in JGA lingo, an Individual is composed of its data structure, the *Genotype,* and its value, the *Phenotype*. In a given JGA application, the *FitnessFunction* is the component in charge of decoding the *Genotype* and computing the *Phenotype*, that is, the visible properties of the individual.

Any class extending *Genotype* must know how to create a new instance, how to initialize it randomly, and how to create a copy. To meet these requirements, the JGA user is forced to implement three methods called instance, initRandom, and clone, respectively. Through the method print (or toString), it is possible for the JGA user to implement a formatted output to visualize the genotype. For example, BinaryGenotype extends *Genotype,* its instance method allocates the necessary amount of memory to create an array of a given length, and the method initRandom generates the random sequence of 0s and 1s.

Similarly, any class extending *Phenotype* must know how to set and get the fitness value(s) (methods setFitnessValue and getFitnessValue), and how to compare to other *Phenotype* (method compare). Lastly, but most importantly, any *FitnessFunction* must be able to evaluate a specific *Genotype*, compute, and return its fitness value (method evaluate). In the inventory application, presented later with greater detail, the ULSFitness (extends *FitnessFunction*) computes the total cost (sum of holding and ordering costs). Due to the fact that the evaluation of the inventory policy is a single value, this total cost is stored in a SingleFitnessPhenotype (extends *Phenotype*).

Genetic Operators

Most genetic algorithms implement two genetic operators: mutation and crossover. The JGA user must declare which operators to use in the main configuration file. It is important to note that these operators are closely related to the genotypes because they operate with their underlying data structures. Therefore, the selection or design of a genetic operator cannot be done in isolation, that is, the user must know how the given *Genotype* works.

Every class extending *MutationOperator* must implement the method mutate, responsible for receiving a genotype, mutating it, and returning the status of the operation (true if the genotype is in fact mutated, or false otherwise). The rationale behind the logic of the mutation operator is to promote *exploration,* that is, to avoid getting trapped in local optima. An example of an existing mutation operator is FlipBinaryMutation, which flips each gene from 0 to 1 (or 1 to 0) with probability MUTRATE (property field in the main configuration file).

On the other hand, the *CrossoverOperator* is usually designed to promote *exploitation,* that is, to search thoroughly near a promising region of the solution space. Every concrete class extending *CrossoverOperator* must implement the method crossover. This method takes two parents (*Genotype*s) and produces one or more children. The way the parents are selected from the population depends on the specific logic of the *GeneticAlgorithm*. For instance, in the BasicGeneticAlgorithm a crossover pool of parents is formed by adding each individual from the population with a probability of CROSSRATE (property field in the main configuration file). An example of a classical crossover operator is SinglePointCrossover. This crossover operator is explained later in the context of an inventory application.

Selective Pressure

The *SelectionOperator* is the driving force of genetic algorithms. Its careful design provides balance between exploration and exploitation. If the selective pressure is too high, early convergence might result in local optima. On the other hand, a low selective pressure might result in slower convergence, but a wider search of the solution space. In a nutshell, the operator takes a population of individuals and selects a subset based on a given logic. This logic is implemented in the method select from classes extending *SelectionOperator*. The so-called sampling space is provided in the logic of the *GeneticAlgorithm*. For instance, the BasicGeneticAlgorithm selects POPSIZE individuals from an enlarged sample space formed by the union of parents and children (see Figure 3).

Some of the most widely used *SelectionOperator*s are the RouletteWheelSelection, BestIndividualSelection, and TournamentSelection. In RouletteWheelSelection, the probability of selecting a given individual in the population is proportional to its fitness value. BestIndividualSelection is an elitist mechanism that chooses the best individuals in the population in a deterministic fashion. Last, in TournamentSelection, subsets of individuals are randomly selected from the population and the best of each subset is selected.

Collecting Statistics

The StatCollector is the JGA component in charge of collecting statistics during the evolutionary process. The type of statistics that are collected can be defined in the main configuration file using the STATISTICSCOLLECTOR property field. The value for this field must be the result of the addition of the values shown in Table 3. For example, to collect both basic and

Table 3. Collected statistics by JGA

Add to STATISTICSCOLLECTOR	Option Description
2^0=1	Collects basic statistics such as the number of function evaluations, mutation, and crossover operations.
2^1=2	Tracks the best individual during the execution of the genetic algorithm. Also collects the generation number and the number of fitness function evaluations when the best individual was found.
2^2=4	Records the best and worst individuals for each generation.

Figure 4. Example of using the StatCollector in JGA

```
1:    import edu.uniandes.copa.jga.*;
2:    public class myApplication {
3:    public static void main (String args[]) {
4:          String configFileName = args[0] ;
5:          ArrayList genotypeParams = new ArrayList () ;
6:          // genotypeParams construction ...
7:          GeneticAlgorithmHandler ga = new GeneticAlgorithmHandler (configFileName , genotypeParams) ;
8:          ga.run() ;
9:          StatCollector statCollector = ga.getStatCollector () ;
10:         Individual bestInd = statCollector.getBestIndividual () ;
11:         long evaluations = statCollector.getEvaluationsForBestIndividual () '
12:         long iterations = statCollector.getIterationsForBestIndividual () ;
13:         long executionTime = statCollector.getExecutionTime () ;
14:   }
15: }
```

best-individual statistics, we set STATISTICSCOLLECTOR=3 (=1+2). Setting STATISTICSCOLLECTOR=0 results in no statistics collection.

The StatCollector class provides methods for retrieving the statistics after the genetic algorithm execution. To get a handle on the StatCollector, the user must use the getStatCollector method from the GeneticAlgorithmHandler object created in the main program (see Figure 4).

THE BENCHMARK PROBLEM: INVENTORY LOT SIZING

To illustrate the use of JGA and to compare it with other tools, we use as a test case the dynamic inventory lot-sizing problem arising in the context of operations management. First we define the problem, then we outline how this problem is solved by means of a genetic algorithm. The implementation of the genetic algorithm in various tools, including JGA, is delayed until the next section.

The Dynamic Inventory Lot-Sizing Problem

In the classical dynamic lot-sizing problem, the decision maker is interested in defining order quantities over a discrete-time planning horizon, while meeting a set of dynamic demand requirements (driven by customers) at a minimum total cost. The total cost includes variable inventory holding costs and fixed ordering costs. It is worth emphasizing that the customer's demand is both *deterministic* and *dynamic*. By *deterministic* we mean that future demand for an inventory item is known with certainty, while by *dynamic* we mean that demand changes over the planning horizon.

A formal definition of the dynamic inventory lot-sizing problem follows. For a single product with demand $D_1, \ldots, D_t, \ldots, D_T$ over a discrete horizon T, the problem is to find the number of

units X_t to be ordered at period *t,* so that the demand is satisfied at a minimal total cost. The total cost is given by the expression:

$$TC = \sum_{t=1}^{T} \left(A_t Y_t + h_t I_t \right)$$

where I_t is the inventory level at the end of period t defined by $I_t = I_{t-1} + X_t - D_t$ (I_0 is the initial inventory level); Y_t is a binary variable that takes the value of 1 if an order is placed at period t (it takes the value of 0, otherwise); h_t is the cost of carrying one unit in inventory at the end of period t; and A_t is the fixed cost of placing an order at period t.

The most widely used heuristic approaches for the lot-sizing problem with dynamic demand are Silver-Meal (Silver & Meal, 1973), Least Unit Cost (Silver, Pyke, & Peterson, 1998), and Part Period Balancing (Silver et al., 1998). For sufficiently small problems the most widely used exact methods are Wagner-Whitin (Wagner & Whitin, 1958) and mixed integer programming (Silver et al., 1998).

The Wagner-Whitin (WW) algorithm is an optimization procedure based on dynamic programming. WW uses the following recursive equation to calculate the total cost for period t, namely $G(t)$:

$$G(t) = \min\left(A_t + G(t-1), \min_{1 \le k < t}\left(A_k + \sum_{r=k}^{t-1} \sum_{s=r+1}^{t} h_r D_s + G(k-1) \right) \right) \quad (1)$$

The WW algorithm uses a property shown in Wagner and Whitin (1958) which states that the demand D_t for any period t in the optimal solution is produced completely (and not partially) in some period k such that $1 \le k \le t$. This property is the key result used in the representation scheme for the genetic algorithm that we outline in this chapter. The WW algorithm evaluates all possible ways of ordering so that demands ahead of the specific order period are covered. WW determines the optimal number of periods to be included in the order and skips to the next order period. The process is repeated until the planning horizon is covered completely. In a single-item problem with T periods, the total number of possible order policies is 2^{T-1}. Fortunately, the WW method uses a clever evaluation scheme that reduces the number of policies to be examined to the $O(T^2)$, in the worst case. The WW method is not widely used in industry because it is difficult to understand. Its major advantage is that it is useful to measure the effectiveness of other lot-sizing algorithms.

A Solution Approach Based on a Genetic Algorithm

For this problem, we use the binary representation proposed by Hernández and Gürsel (1999). Each chromosome consists of T genes, corresponding to the T periods. The gene t of a chromosome indicates if an order has been placed in period t or not (values of 1 or 0, respectively). The order quantity includes the demands of period t up to the next ordering period. Figure 5 shows an example of the order policy defined by a given chromosome for a problem with six periods. In this example, orders are placed at periods 1, 3, and 4.

Order quantities: $X_1 = D_1 + D_2$, $X_2 = 0$, $X_3 = D_3$, $X_4 = D_4 + D_5 + D_6$, $X_5 = 0$, and $X_6 = 0$

THE JGA IMPLEMENTATION

The genetic algorithm for this problem was implemented using many of the built-in components provided by JGA. As explained in the

Figure 5. Example of a chromosome for the dynamic lot-sizing problem

Chromosome:	1	0	1	1	0	0
Period	1	2	3	4	5	6
Demand	D_1	D_2	D_3	D_4	D_5	D_6

previous section, the key decision was to encode the solutions into a binary genotype. Even though a binary genotype already exists in JGA (class BinaryGenotype), we extended it to rewrite the toString method so that we can show the solution in a tabular user-friendly format. The new encoding is implemented in class ULSBinaryGenotype. Furthermore, the fitness function class ULSFitness is in charge of decoding the chromosome and evaluating its inventory and setup costs. The result of this evaluation is stored in the individual's phenotype (class SingleFitnessPhenotype).

For the binary genotype behind this lot-sizing problem, we chose a widely used set of genetic operators: single-point crossover (class SinglePointBinaryCrossover) and flip mutation (class FlipBinaryMutation). In single-point crossover, two parent chromosomes produce two offspring. First, a cut point for the parent chromosomes is selected randomly. Then, the first (second) offspring is generated by merging the genes to the left (right) of the cut point for the first parent with the genes to the right (left) of the cut point for the second parent. The mutation operator flips the value of a gene (0 to 1, or 1 to 0) with a given probability of mutation MUTRATE.

We used an elitist selection mechanism (class BestIndividualSelection) that selects the best individuals from an enlarged population. The enlarged population is formed by children produced from crossover and mutation, as well as with individuals from the current population. Table 4 summarizes the built-in components

Table 4. Summary of JGA classes used for the dynamic inventory lot-sizing problem

Component	Class	Extended from Class[1]
Genotype	ULSBinaryGenotype •	BinaryGenotype ♦
Phenotype	SingleFitnessPhenotype ♦	*Phenotype* ♦
Fitness Function	ULSFitness •	*FitnessFunction* ♦
Mutation Operator	FlipBinaryMutation ♦	*MutationOperator* ♦
Crossover Operator	SinglePointBinaryCrossover ♦	*CrossoverOperator* ♦
Selection Operator	BestIndividualSelection ♦	*SelectionOperator* ♦

[1] *Italics used to indicate abstract classes*

♦ *JGA built-in class*

• *User-defined class*

used from the JGA package to solve the dynamic inventory lot-sizing problem. Figure 6 shows the class diagram for the genetic algorithm implemented using JGA to solve the dynamic inventory lot-sizing problem.

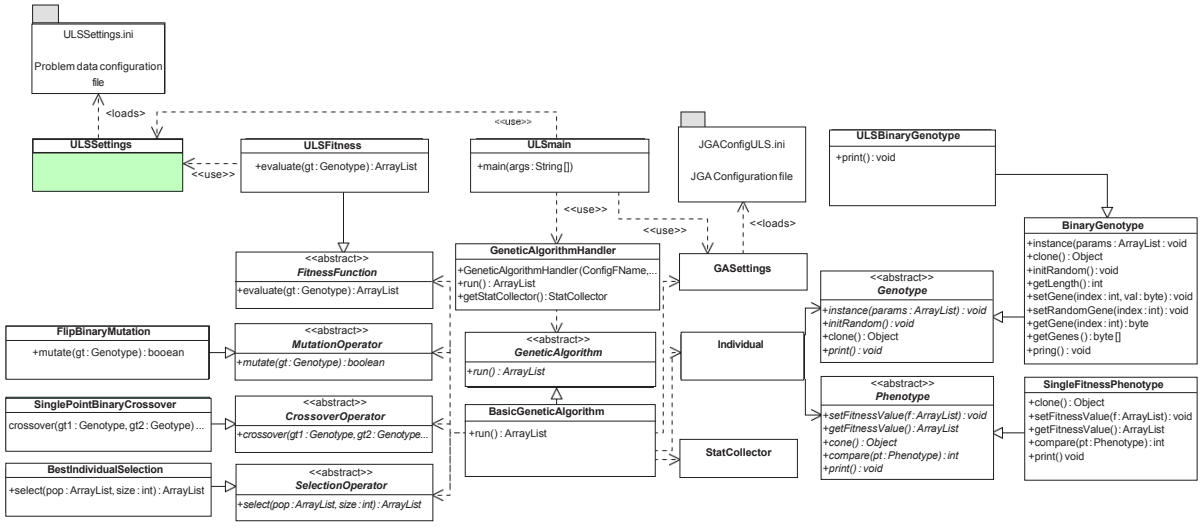
COMPARISON OF JGA WITH OTHER COMPUTATIONAL TOOLS

The dynamic inventory lot-sizing problem was used to compare JGA with three other tools: GAlib (Wall, 2005), PROC GA from SAS (SAS Institute, 2005), and the GADS toolbox from Matlab (Mathworks, 2005).

The GAlib Implementation

A GAlib user must write a main function in order to create instances of the required objects, including the genome and a GAsimpleGA object. For the lot-sizing application, it was necessary to code the fitness function and to set the genetic algorithm parameters via the main function. The genome selected for solving the problem was the GA1DBinaryStringGenome. The genetic operators and selection mechanism were set by means of a header file named *gaconfig.h* (see Figure 7). Figure 8 shows part

Figure 6. Class diagram for the JGA implementation of the dynamic lot-sizing problem



of the code used for solving the dynamic lot-sizing problem with GAlib.

The SAS (PROC GA) Implementation

SAS provides an experimental procedure called PROC GA, useful for solving optimization problems with binary, integer, and real variables. SAS also provides a combinatorial variable, useful for problems in which solutions are based on permutations. The variables can be encoded in fixed-length vectors allowing lower and upper bounds. The encoding may include several segments, each one of them with different data types. The parameters and operators are defined by means of calls to functions in the body of the procedure. The user must write the fitness function using the SAS programming language. PROC GA provides an elitist mechanism that lets the user preserve the best individuals for the next generation. Figure 9 shows the PROC GA code to solve the dynamic inventory lot-sizing problem.

In PROC GA, the user can define a subroutine for execution at the end of every itera-

Figure 7. GAlib configuration header file

```
// Configuration & preproprocesor directives

.....
.....

// These are the compiled-in defaults for various genomes and GA objects
#define DEFAULT_SCALING                    GALinearScaling
#define DEFAULT_SELECTOR                   GARouletteWheelSelector
#define DEFAULT_TERMINATOR                 TerminateUponGeneration

#define DEFAULT_1DBINSTR_INITIALIZER       UniformInitializer
#define DEFAULT_1DBINSTR_MUTATOR           FlipMutator
#define DEFAULT_1DBINSTR_CROSSOVER         OnePointCrossover
```

Figure 8. Code for implementing the dynamic lot-sizing problem in GAlib

```
#include <GASimpleGA.h>                 // use the simple GA
#include <GA1DBinStrGenome.h>           // use the 1D binary string genome
// main function
main (int argc, char **argv) {
        // genome construction.
        GA1DBinaryStringGenome genome (len, Objective) ;

// Set the GA parameters
GASunokeGA ga(genome) ;
ga.minimize () ;
ga.populationSize(popsize) ;
ga.nGenerations(ngen) ;
ga.pMutation (pmut) ;
ga.pCrossover (pcross) ;

// run the GA
ga.evolve() ;

// print out the best genome that the GA found.
cout <<"The GA found: \n" <<ga.statistics().bestIndividual() << "\n";
}
// Objective function definition.
float Objective(GAGenome & g) {
        GA1DBinaryStringGenome & genome = (GA1DBinaryStringGenome &) g ;
        float cost=0.0 ;
        // Objective function evaluation.....
        return cost ;
}
```

Figure 9. Code for implementing the GA application using the GA procedure from SAS

```
proc ga seed = 1 maxiter = 20;
/* Set the problem encoding */
call SetEncoding('I12') ;
/* Set upper and lower bounds on the solution components */
array LowerBound [12] /nosym (0 0 0 0 0 0 0 0 0 0 0 0) ;
array UpperBound [12] /nosym (1 1 1 1 1 1 1 1 1 1 1 1) ;
call SetBounds (LowerBound, UpperBound) ;
/* Set the objective function to be optimized */
call SetObjFunc ('ulsCost', 0) ;
/* Set the crossover paramters */
call SetCrossProb (crossoverProbability) ;
call SetCrossRoutine ('singlePointCrossover')
/* Set the mutation routine to flip */
call SetMutProb(1.0) ;
call SetMutRoutine('flipMutation') ;
/* Set the selection criteria
/* call SetElite(n) ;
/* = number of best individuals from last population to be retained */
call setElite(elitistSize) ;
/* Initialize the first generation, with 120 random solutions */
call Initialize ('DEFAULT', popSize) ;
run ;
/* Objective function to be optimized */
function ulsCost (selected [*]) ;
        /* Local array for reading chromosome information */
        array x[12] /nosym ;
        call ReadMember (selected,1,x) ;
        /* Calculating the total cost */
        return(totalCost) ;
endsub ;
```

tion or the evolutionary process. This feature is useful for reporting intermediate results, adjusting parameters dynamically, or performing local re-optimizations.

PROC GA also provides useful functions to deal with multi-objective and constrained problems. The first function returns the Pareto frontier from a population, while the second function evaluates a set of linear constraints for a given solution. The current version offers only a small set of built-in genetic operators and selection mechanisms.

The Matlab (GADS Toolbox) Implementation

The GADS toolbox from Matlab includes a set of elements and parameters very similar to SAS's PROC GA. The main difference is that all settings are passed via the parameters of Matlab's *ga* function. Figure 10 shows the code in the m-File used to run the *ga* function.

Matlab provides a graphical user interface (see Figure 11) by calling the *gatool* command. This interface generates a file with the line command code shown in Figure 10. GADS also includes a large set of options for fitness scaling, termination criteria, migration, and reporting (statistical graphs). By writing m-files on a given format, the user can implement new data types for individuals and genetic operators.

Computational Experiment: A Quantitative and Qualitative Comparison

An experiment was conducted to test the results of each implementation using a sample problem from Silver et al. (1998). Table 5 shows the problem data and the optimal solution (total cost of 501.2) for Silver's instance.

The implementation for each tool uses the same representation scheme and genetic operators outlined in the JGA implementation. The mutation and crossover probabilities were set to 0.01 and 0.80, respectively, while the population size was set to 40 individuals. To analyze the performance of these tools, we constructed several instances based on three

Figure 10. Code for implementing the GA application using the GADS toolbox from MATLAB

```
function [X,FVAL,REASON,OUTPUT,POPULATION,SCORES] =  ulsGA
%%   This is an auto generated M file to do optimization with the Genetic Algorithm and
%    Direct Search Toolbox. Use GAOPTIMSET for default GA options structure.
%%Fitness function
fitnessFunction = @ulsfunction;
%%Number of Variables
nvars = 12;
%Start with default options
options = gaoptimset;
%%Modify some parameters
options = gaoptimset(options,'PopulationType' ,'bitString');
options = gaoptimset(options,'PopulationSize' ,40);
options = gaoptimset(options,'EliteCount' ,4 );
options = gaoptimset(options,'Generations' ,20);
options = gaoptimset(options,'StallGenLimit' ,20);
options = gaoptimset(options,'StallTimeLimit' ,Inf);
options = gaoptimset(options,'FitnessScalingFcn' ,@fitscalingprop);
options = gaoptimset(options,'CrossoverFcn' ,@crossoversinglepoint);
options = gaoptimset(options,'MutationFcn' ,{ @mutationuniform 0.01 });
options = gaoptimset(options,'Display' ,'off');
%%Run GA
tic;[X,FVAL,REASON,OUTPUT,POPULATION,SCORES] = ga(fitnessFunction,nvars,options);toc;


% Fitness function for the ULS problem
% Computes the total cost ( inventory holding costs + ordering cost )
function score = ulsfunction(solution)

score = totalCost;
```

different levels for the number of generations (20, 100, and 1,000) and two problem sizes based on the number of periods *T* (12 and 48). For each instance, 30 runs were conducted on an Intel Pentium IV processor running at 2.8GHz with 256 MB of RAM. Table 6 and Table 7 show the results of these experiments for the average cost (fitness function) and CPU time, respectively.

GAlib shows the best performance in terms of execution time, always requiring less than a third of the time needed by JGA (see Table 7). GAlib is also a flexible tool that includes the major elements of traditional GAs. However, the main disadvantage is that it is targeted to advanced users with C++ programming skills. It took us a significant amount of time to compile and set up GAlib. Also, a number of compiler options must be set and verified depending on the selected platform.

SAS's PROC GA main advantage is its user-friendly programming language and the integration with the SAS system, which includes its powerful data processing capabilities. Very useful features from PROC GA are its built-in multi-objective and constrained opti-

Figure 11. Graphical user interface of GADS from MATLAB

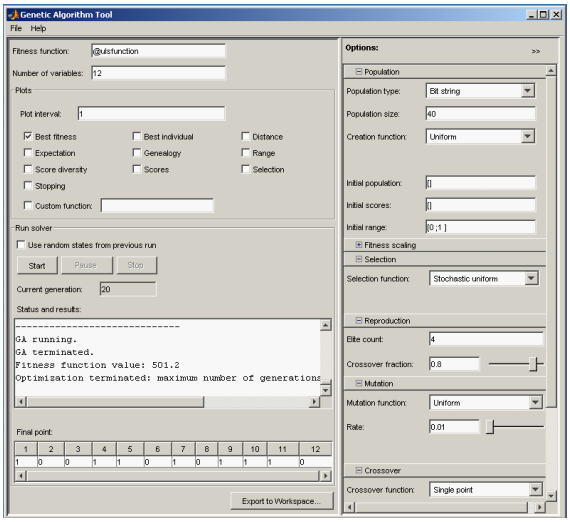


Table 5. Data and optimal solution for Silver's instance of the dynamic inventory lot-sizing problem

Period	1	2	3	4	5	6	7	8	9	10	11	12
Demand	10	62	12	130	154	129	88	52	124	160	238	41
Setup Cost	54	54	54	54	54	54	54	54	54	54	54	54
Inventory Cost	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
Order Quantity	84	0	0	130	283	0	140	0	124	160	279	0
Final Inventory	74	12	0	0	129	0	52	0	0	0	41	0

Table 6. Average cost results for the dynamic inventory lot-sizing problem

Problem Instance (T-P-NG[1])	**JGA**		**GAlib**		**PROC GA (SAS)**		**GADS (Matlab)**	
	Average Cost	**Average Gap**	**Average Cost**	**Average Gap**	**Average Cost**	**Average Gap**	**Average Cost**	**Average Gap**
12-40-0020	501.20	0.00%	508.13	1.38%	501.20	0.00%	502.90	0.34%
12-40-0100	501.20	0.00%	501.20	0.00%	501.20	0.00%	501.20	0.00%
12-40-1000	501.20	0.00%	501.20	0.00%	501.20	0.00%	501.20	0.00%
48-40-0020	1967.00	1.39%	2202.00	13.51%	1995.00	2.84%	2126.00	9.59%
48-40-0100	1940.00	0.00%	2015.00	3.87%	1953.00	0.67%	1985.00	2.32%
48-40-1000	1940.00	0.00%	1942.00	0.10%	1940.00	0.00%	1942.00	0.10%

[1] *T=Number or periods, P= Population size, NG=Number of generations*

Table 7. Average CPU time results for the dynamic inventory lot-sizing problem

Problem Instance (T-P-NG[1])	**JGA**	**GAlib**		**PROC GA (SAS)**		**GADS (Matlab)**	
	Time (ms)	**Time (ms)**	**Ratio[2]**	**Time (ms)**	**Ratio[2]**	**Time (ms)**	**Ratio[2]**
12-40-0020	12	3	0.25	135	11.25	73	6.08
12-40-0100	48	11	0.23	174	3.63	318	6.63
12-40-1000	474	103	0.22	848	1.79	3068	6.47
48-40-0020	22	6	0.27	166	7.55	82	3.73
48-40-0100	96	30	0.31	353	3.68	355	3.70
48-40-1000	890	262	0.29	2518	2.83	3123	3.51

[1] *T=Number or periods, P= Population size, NG=Number of generations*

[2] *Uses JGA's CPU time as the ratio denominator*

mization capabilities. However, the main drawbacks are its slow execution time and its limited and non-extensible data structures for solution encoding.

Matlab's GADS shares similar advantages to that of SAS, namely, its powerful, matrix-oriented, user-friendly language and its integration to Matlab's suite of toolboxes. The main disadvantage of GADS is its slow execution time. For instance, the time required to run GADS was about three to six times compared to that of JGA.

For every instance of the dynamic lot-sizing problem, JGA was consistently the best in terms of solution quality (it consistently achieved the optimum). In terms of computing performance, JGA was second to GAlib and clearly outperformed SAS and Matlab.

CONCLUSION

We have presented the Java Genetic Algorithm (JGA) framework, a flexible object-oriented computational tool for rapid prototyping and implementation of evolutionary algorithms. From a practical side, we have shown how it is possible to use genetic algorithms and JGA for modeling and solving complex business problems. GA-based solutions are often quite competitive, yet easier to implement than the state-of-the-art method of choice.

In this chapter, we compared JGA against other implementations of GAs. We compared the tools using as benchmark a dynamic inventory lot-sizing problem. Compared to the C++ implementation GAlib, JGA is slower. However, in terms of solution quality, JGA is a better performer. It is worth mentioning that GAlib produces native code, while JGA is platform independent and runs on top of the Java Virtual Machine. Some speed up can be achieved by a native compilation of JGA, but this affects its cross-platform independency. With respect to the commercial offerings from Matlab and SAS (GADS and PROC GA, respectively), JGA outperforms both in terms of time and quality. However, a nice feature of GADS and PROC GA, is that they are leveraged by a user-friendly and powerful language, being part of a wider system (i.e., suite of Matlab toolboxes and the SAS system, respectively).

Just as genetic algorithms provide a flexible framework to solve a broad range of problems, JGA and its constantly expanding set of class libraries, provides a flexible set of building blocks that makes it possible to implement a broad range of genetic algorithm-based solutions. We hope the readers can benefit directly by using JGA class libraries available at http://copa.uniandes.edu.co/soft-evol-jga.html to better manage scarce resources and optimize operations using nature-inspired computing.

REFERENCES

Chipperfield, A. J., & Fleming, P. J. (1995). The Matlab genetic algorithm toolbox. *IEE colloquium on applied control techniques using Matlab.* Digest No. 1995/014. Retrieved June 26, 2005, from http://www.shef.ac.uk/acse/research/ecrg/gat.html/

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182-197.

Derderian, K. (2002). *GGAT: General genetic algorithm tool.* Retrieved June 26, 2005, from http://www.karnig.co.uk/ga/ggat.html/

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning.* Reading, MA: Addison-Wesley.

Goodman, E. (2002). *GALOPPS 3.2.4: Genetic algorithm optimized for portability and parallelism system.* Genetic Algorithms Research and Applications Group, Michigan State University, USA. Retrieved June 22, 2005, from http://garage.cse.msu.edu/software/galopps

Hernández, W., & Gürsel A. S. (1999). Genetic algorithms in lot sizing decisions. *Proceedings of the Congress on Evolutionary Computation* (CEC99), Washington, DC (Vol. 3, pp. 2286-2289). IEEE Press.

Houck, C., Joines, J., & Kay, M. (1995). *A genetic algorithm for function optimization: A Matlab implementation.* Technical Report 95-09, Industrial Engineering Department, North Carolina State University, USA. Retrieved June 26, 2005, from http://www.ie.ncsu.edu/mirage/GAToolBox/gaot/

Lobo, F. G., & Harik, G. R. (1999). *ECGA: Extended compact GA in C++*. Report 99016, Illinois Genetic Algorithm Laboratory, University of Illinois, USA. Retrieved June 22, 2005, from http://www-illigal.ge.uiuc.edu/sourcecd.html/

MathWorks. (2005). GADS: *Genetic algorithm and direct search toolbox.* Retrieved June 26, 2005, from http://www.mathworks.com/products/gads

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs.* New York: Springer-Verlag.

Pain, A. W., & Reeves, C. R. (2002). Genetic algorithm optimization software class libraries. In S. Vo² & D. L. Woodruff (Eds.), *Optimization software class libraries* (pp. 295-329). Secaucus, NJ: Kluwer Academic.

Palisade Corp. (2005). *Evolver.* Retrieved June 26, 2005, from http://www.palisade.com/evolver/

Pohlheim, H. (2005). *GEATbx introduction: Evolutionary algorithms: Overview, methods and operators.* Retrieved June 26, 2005, from http://www.geatbx.com/

Rotstan, N., & Meffert, K. (2005). *JGAP: Java genetic algorithm package.* Retrieved June 26, 2005, from http://jgap.sourceforge.net/

SAS Institute. (2003). *SAS/OR 9.1 user's guide: Local search optimization.* Cary, NC: SAS Publishing.

Silva, S. (2005). *GPLAB: A genetic programming toolbox for MATLAB.* Evolutionary and Complex Systems Group, University of Coimbra, Portugal. Retrieved June 26, 2005, from http://gplab.sourceforge.net/

Silver, E. A., & Meal, H. C. (1973). A heuristic for selecting lot size quantities for the case of a deterministic time-varying demand and discrete opportunities for replenishment. *Production and Inventory Management Journal, 14*(2), 64-74.

Silver, E. A., Pyke, D. F., & Peterson, R. (1998). *Inventory management and production planning and scheduling.* New York: John Wiley & Sons.

Wagner, H., & Whitin, T. M. (1958). Dynamic version of the economic lot size model. *Management Science, 5*(1), 89-96.

Wall, M. (2005). *GAlib: A C++ library of genetic algorithm components*. Retrieved June 22, 2005, from http://lancet.mit.edu/ga/

Ward Systems Group. (2003). *GeneHunter.* Retrieved June 26, 2005, from http://www.wardsystems.com/

KEY TERMS

Abstract Class: In object-oriented technology, a type of class that defines a master structure from which concrete classes are derived. An abstract class provides a contract that users must subscribe to when implementing their own classes extending from it.

Chromosome: Data structure (genetic code) that encodes a solution (individual) to the problem that the genetic algorithm is trying to solve.

Class: Blueprint of an object, but not the object itself. It defines attributes and methods common to all objects of a certain kind.

Crossover: Genetic operator that combines (mates) chromosomes (parents) to produce new ones (offspring). It is analogous to the biological reproduction.

Fitness Function: Objective function that quantifies the adaptability of an individual. Solutions with better fitness values are more likely to survive and join the next generation in a genetic algorithm. It is analogous to the fitness concept in the natural selection process.

Gene: Basic information unit in a chromosome.

Genetic Algorithm (GA): Stochastic (population-based) search technique, inspired in the natural selection process, used to find approximate solutions to complex optimization problems.

Genotype: Genetic makeup of an individual.

Method: Fundamental element in object-oriented programming that defines the object's behavior.

Mutation: Genetic operator that alters one or more genes in a chromosome. Mutation is used to maintain genetic diversity in the population of individuals. It is analogous to biological mutation.

Object: Building block of object-oriented programming. Every object has a state, behavior, and identity. The object's state is defined by the instance variables, while its behavior is defined by the object's methods. An object is capable of receiving messages, processing data, and sending messages to other objects.

Object-Oriented Programming (OOP): Computer programming paradigm in which a computer program is written by defining objects, its behavior, and interrelations. The fundamental terms in OOP are objects, classes, and methods.

Phenotype: Features or quantifiable measurements of an individual. In a genetic algorithm, the phenotype is the value associated with the fitness function evaluation.

Selection: Operation in charge of choosing which individuals from the current population will survive and become part of the next generation.

Chapter XLI
Applications of JGA to Operations Management and Vehicle Routing

Andrés L. Medaglia
Universidad de los Andes, Colombia

Eliécer Gutiérrez
Universidad de los Andes, Colombia

ABSTRACT

Two of the most complex activities in production and operations management (POM) are inventory planning and operations scheduling. This chapter presents two problems related to these activities, namely, the capacitated lot-sizing and scheduling problem and the capacitated vehicle routing problem. For each of these problems, the authors discuss several solution methods, present a competitive genetic algorithm, and describe its implementation in the Java Genetic Algorithm (JGA) framework. The purpose of this chapter is to illustrate how to use JGA to model and solve complex business problems arising in POM. The authors show that JGA-based solutions are quite competitive and easier to implement than widely used methods found in the literature.

INTRODUCTION

JGA, the acronym for Java Genetic Algorithm, is a flexible and extensible computational object-oriented framework for rapid development of evolutionary algorithms for solving complex optimization problems. JGA shortens the implementation phase because the user of JGA does not have to implement the underlying logic of a genetic algorithm, but just has to focus on the unique aspects of its specific problem. The previous chapter presents the technical aspects of JGA in detail; the purpose of this chapter is to illustrate the use of JGA with problems arising in the areas of operations management and vehicle routing.

Production and operations management (POM) decisions deal with the efficient use of the resources in the production of goods and services. Two important aspects in POM are inventory planning and operations scheduling. The former defines inventory policies and determines order quantities. The latter determines the detailed sequence of the production orders or jobs through the machines. These problems, also known in the literature as lot-sizing and scheduling problems, must support several (often conflicting) objectives, such as minimizing the total inventory and setup costs, while maximizing service level.

Our first application of POM was presented in the previous chapter. We illustrated JGA and other genetic algorithm tools in the classical dynamic lot-sizing problem, in which the decision maker is interested in defining order quantities over a discrete-time planning horizon, while meeting a set of dynamic demand requirements at a minimum total cost.

Traditional solution approaches of POM deal with lot-sizing and sequencing problems separately and do not consider the interrelationship between them. A common solution procedure is to solve these problems sequentially—that is, first find a solution for the lot-sizing problem (as in the previous chapter), and then solve the sequencing problem taking as input the lot-sizing solution without considering capacity constraints. This approach generates infeasible solutions which often exceed production capacity and must be adjusted in order to satisfy the capacity constraints. This adjustment often generates high-cost solutions. In this chapter we show how by integrating both problems by means of a genetic algorithm; it is possible to obtain a significant operational cost reduction.

The second application presented in this chapter deals with the distribution of products and services, namely, with the vehicle routing problem (VRP). The VRP is to find the best way of assigning a group of customers to a fleet of vehicles and determining the sequence of customer visits. The managerial objective is to provide a high service level to the customers, while simultaneously minimizing operational and investment costs.

The next two sections of this chapter discuss the capacitated lot-sizing and scheduling problem and the capacitated vehicle routing problem. For each of these problems, we discuss its solution methods, and present a genetic algorithm approach and its implementation in JGA along with a computational experiment.

THE CAPACITATED LOT-SIZING AND SCHEDULING PROBLEM

This problem deals with the integration of lot-sizing and scheduling decisions over a parallel machine environment. This problem, known as capacitated lot-sizing and scheduling (CLSS), formally can be stated as follows. There are N different products to be manufactured in a system with M parallel and identical machines over a finite horizon of time. The time horizon comprises T evenly spaced periods. Product i has a deterministic demand of D_{it} units for period t. There is an inventory holding cost h_i for each unit of product i held in inventory per time period. Each product is processed by one machine and requires p_i units of time per unit of product i. Associated with the production process, there is a cost C_p per unit of processing time. When a machine changes from processing product i to product j, it requires s_{ij} units of time for setup. Associated with this setup, there is a cost C_s per unit of time. Production is constrained to a maximum capacity of C units of time per period per machine. In conclusion, the CLSS problem is to find the number of units X_{it} of product i to be produced at period t, and

the sequence in which the production orders associated to X_{it} are to be produced in each period. The objective is to reduce the total relevant costs, that is, the sum of the inventory carrying costs plus the setup costs.

Scheduling methods are classified in the literature according to the type of process configuration and machine arrangement. These methods cover a single machine, parallel machines, flow shops, job shops, and open shops (Nahmias, 1998). Typical objectives include minimizing average flow time, tardiness, maximum completion time (also known as makespan or C_{max}), and the number of tardy jobs (Sipper & Bulfin, 1997).

In parallel machine scheduling, the simplest case assumes that any product can be processed on any machine, the product's processing time is the same on any machine (i.e., machines are identical), and products require a single operation. For this problem, a family of approximation algorithms is based on the list scheduling (LS) approach (Cheng & Sin, 1990). An LS is an ordered sequence of the products generated using some criterion, like a dispatching rule. The schedule is generated by assigning the next product on the list to the machine with the least amount of work assigned (i.e., first available machine). This approach has been extensively used for solving problems with flowtime and due-date performance measures.

Lee and Pinedo (1997) present a three-phase heuristic to schedule jobs on parallel machines with sequence-dependent setup times. They minimize the sum of weighted tardiness. The sequence is constructed using a modified version of the apparent tardiness cost with setup (ATCS) rule. A statistical analysis is proposed to determine the parameters for the dispatching rule. A post-processing procedure based on neighborhood search is performed to improve the solution.

Some work has been done that integrates the lot-sizing and scheduling problems. Sikora, Chhajed, and Shaw (1996) describe an integrated approach (IA) for a general problem of scheduling a flow line with sequence-dependent setups and capacity constraints. IA is an iterative approach that does the sequencing for each period before an additional lot is scheduled. To decide the next lot to be scheduled, the lot-sizing module calls the sequencing heuristic. The sequencing module finds the sequence that minimizes the makespan and returns it to the lot-sizing module. Then the lot-sizing module decides when the next lot is scheduled based on the utility function concept developed by Dixon and Silver (1979). This approach is able to give feasible solutions in terms of capacity with reasonable cost values.

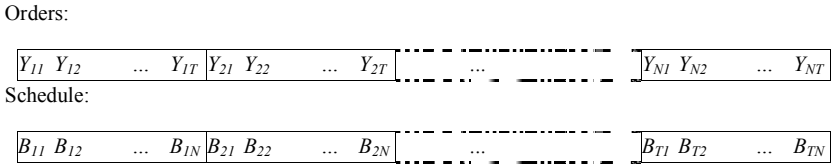
A Solution Approach Based on a Genetic Algorithm

The main challenge in the design of the genetic algorithm is the integration of the lot-sizing and scheduling problems. Therefore, chromosome representation and genetic operators must include and handle both problems. The inventory problem deals with the order plan for each product; the scheduling problem deals with finding the sequence of orders in the machines. The total cost evaluation takes information from both problems: the inventory holding costs are evaluated from the order plan, while the setup costs are obtained from the schedule.

Representation Scheme

The chromosome used to represent a solution for the problem is composed of two parts. The first part represents the order plan for all the products; the second part represents the sequence of orders in the schedule for all the

Figure 1. Chromosome structure for the CLSS problem



periods. Figure 1 shows the structure of the chromosome.

In Figure 1, for orders, each product has a block in which each binary gene Y_{it} indicates if an order for product i has been placed or not in period t (values of 1 or 0, respectively). This is an extension of the representation used for the dynamic lot-sizing problem (see previous chapter) to the multi-product case. For the schedule, each period has a block with a permutation that defines the sequence in which orders must be scheduled on the machines. Each integer gene B_{ti} specifies the i-th product in the sequence for period t. Under this representation, machines are not considered explicitly in the chromosome, but this allocation is performed later by means of the list scheduling algorithm. The representation based on permutations (also called *order representation*) is the most natural representation for sequences (Gen & Cheng, 2000).

The fitness function uses the list scheduling algorithm (LS) to assign and sequence orders into the parallel machine system. The list scheduling algorithm assigns each order to the machine with the least completion time. Figure 2 shows an example of LS algorithm for 10 orders and three machines using the permutation {9,1,3,8,2,4,5,7,6,10}. The black boxes between orders correspond to setup times.

Two other list-schedule-based algorithms were included in the design: smallest setup time (LS-SST) (Sipper & Bulfin, 1997) and Smallest Cost (LS-SC) (Gutiérrez, 2002). The first algorithm assigns the order to the machine with the smallest setup time; the second uses the total costs as the comparison criterion.

The inventory holding costs and setup costs are added into the fitness function used to evaluate the individuals in the evolutionary process. The capacity constraint is handled by penalizing the individual's fitness if the completion time for period t, $C_{max}(t)$, exceeds the maximum available capacity C. The general form of the fitness function follows:

Figure 2. Gantt chart representing the schedule using LS algorithm

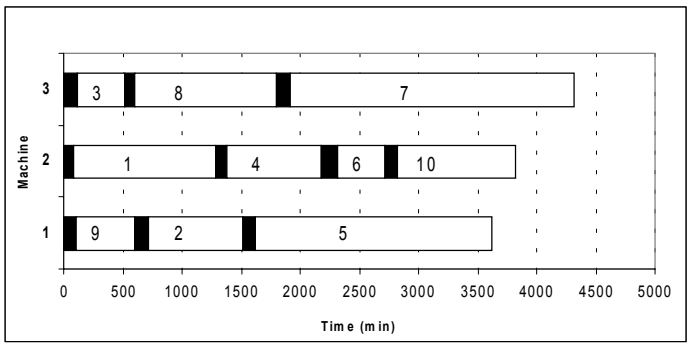
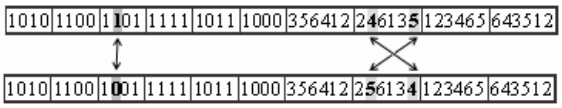


Figure 3. Example of the mutation operator for the CLSS chromosome



$$\text{Fitness} = \text{Inventory holding costs} + \text{setup costs} + \sum_{t=1}^{T}(1+\alpha)\max\{0, C_{\max}(t) - C\}C_p \quad (1)$$

where α is a penalty factor ($1 \leq \alpha \leq 2$).

Genetic Operators

For each individual, orders and schedule are mutated with a given mutation's probability. The mutation operator combines flip mutation and exchange mutation. Orders mutate by selecting randomly a position (product and period) and flipping its associated gene from 0 to 1 (or from 1 to 0). The schedule mutates by randomly selecting a period and randomly exchanging two positions in the sequence for the given period. Figure 3 shows an example with an individual for a problem consisting of six products and four periods. In the example, the second gene for product 3 was flipped from 1 to 0. Also, the second and the sixth genes for period 2 were exchanged.

The crossover operator is also different for each part of the chromosome. The first part, corresponding to the binary genes, is crossed as a whole using a single-point crossover as explained for the dynamic lot-sizing case. The second part, corresponding to the sequencing problem, is crossed for each period using the order crossover (OX) (Davis, 1991). The OX crossover operator works as follows. First, two cut points for the parent chromosomes are selected randomly. Then the contained sub-strings between the two cut points of both parents are exchanged. Finally, it completes the remaining positions for each chromosome starting from the right cut point for both parents by omitting the duplicated genes. When the last position of the sequence is reached, it continues from the first position until the chromosome is completed. Figure 4 shows the process of crossing two individuals. For the orders part, gray zones are the left and right sides of the cut point. For the scheduling part, gray zones correspond to the sub-strings between the two cut points for each period.

The JGA Implementation for the CLSS

For the CLSS, several new Java classes were implemented according to the genetic algorithm described in the previous section. The chromosome was implemented in class CLSSGenotype to mix the binary and permutation encoding. The toString method in this class formats the output to show the order quantities for each product and the schedule for each period. The previously described genetic operators were implemented in classes CLSSMutation and CLSSCrossover. These genetic operators work

Figure 4. Example of the crossover operator for the CLSS chromosome

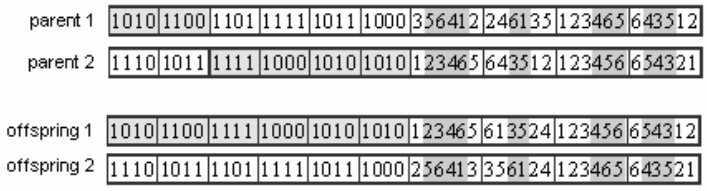


Table 1. Summary of classes for the capacitated lot-sizing and scheduling (CLSS) problem

Component	Class	Extended from Class[1]
Genotype	CLSSGenotype•	*Genotype*♦
Phenotype	SingleFitnessPhenotype♦	*Phenotype*♦
Fitness Function	CLSSFitnessLS• CLSSFitnessLS-SST• CLSSFitnessLS-SC•	*FitnessFunction*♦
Mutation Operator	CLSSMutation•	*MutationOperator*♦
Crossover Operator	CLSSCrossover•	*CrossoverOperator*♦
Selection Operator	BestIndividualSelection♦	*SelectionOperator*♦

[1] *Italics used to indicate abstract classes*

♦ *GA built-in class*

• *User-defined class*

Table 2. Results for JGA for the capacitated lot-sizing and scheduling problems

	LS Algorithm			LS-SST Algorithm			LS-SC Algorithm		
Problem (N,T,M)	Average Cost	Variation Coef.♦	Average Time (ms)	Average Cost	Variation Coef.♦	Average Time (ms)	Average Cost	Variation Coef.♦	Average Time (ms)
P1(10-12-4)	1,090,319	0.038	7,518	1,023,846	0.023	7,809	1,011,228	0.022	7,940
P2(20-12-7)	2,588,237	0.088	15,628	2,271,085	0.122	17,671	1,953,944	0.081	18,136

♦ *Variation Coefficient = Standard Deviation/Average*

on the CLSSGenotype. The fitness function class provides the service of decoding the chromosome and evaluating the total cost. A specific fitness function class was implemented for each of the three available algorithms to generate the schedule on the parallel machines (classes CLSSFitnessLS, CLSSFitnessLS-SST, and CLSSFitnessLS-SC). The result of the fitness function evaluation is stored in a single-value phenotype (class SingleFitnessPhenotype).

Table 1 summarizes the implemented classes and built-in components used from JGA to solve the capacitated lot-sizing and scheduling problem.

Computational Experiments

An experiment was conducted to test the results of the implemented genetic algorithm for the CLSS problem using two problems from Gutierrez (2002). Problem 1, labeled P1(10-12-4), consists of 10 products, 12 periods, and 4 machines. Problem 2, labeled P2(20-12-7), has 20 products, 12 periods, and 7 machines. Both problems were also solved using the Integrated Approach (Sikora et al., 1996). Mutation (MUTRATE) and crossover (CROSSRATE) probabilities were set to 0.4 and 0.9 respectively. The population size (POPSIZE) was set to 200 individuals, and the number of generations (MAXGEN) to 600.

Table 2 shows the results for the GA using the three scheduling algorithms described previously. Each method is applied to both problems, and 20 replications (runs) were executed. JGA's CLSS was run on a Pentium IV running at 2.8 GHz with 256MB of RAM. Results show better performance for the smallest cost algorithm compared to the small setup time and the

Table 3. Results for best solutions found by the genetic algorithm

Problem (N,T,M)	Integrated Approach	LS-Algorithm	Improvement	LS-SST Algorithm	Improvement	LS-SC Algorithm	Improvement
P1(10-12-4)	1,416,631	1,027,744	27%	977,369	31%	971,744	31%
P2(20-12-7)	2,687,048	2,76,809	19%	1,835,480	32%	1,726,960	36%

Table 4. Lot-sizing for best solution found for problem 2

Product 1												
Genotype	1	0	1	0	1	0	1	0	1	1	0	1
Orders												
Period	1	2	3	4	5	6	7	8	9	10	11	12
Demand	56	55	57	57	59	58	56	57	56	58	58	58
Order Quantity	111	0	114	0	117	0	113	0	56	116	0	58
Inventory Level	55	0	57	0	58	0	57	0	0	58	0	0
Product 2												
Genotype	1	0	1	0	0	0	0	1	0	0	1	1
Orders												
Period	1	2	3	4	5	6	7	8	9	10	11	12
Demand	16	15	14	8	12	10	18	22	18	12	19	30
Order Quantity	31	0	62	0	0	0	0	52	0	0	19	30
Inventory Level	15	0	48	40	28	18	0	30	12	0	0	0

list scheduling algorithms. Statistical tests were made to verify that the differences in the results for the three algorithms were indeed statistically significant.

Table 3 shows the results for the best solution found using each algorithm. It also shows the improvement with respect to the integrated approach algorithm. For both problems, the best solution generated by the genetic algorithm has an average improvement of about 33.5% over the integrated approach algorithm.

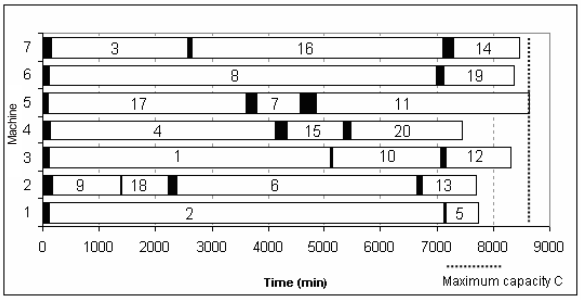
Table 4 shows the lot-sizing solution for product 1 and product 2 in the best solution found for problem 2. Figure 5 shows the schedule for period 1 using a Gantt chart.

THE VEHICLE ROUTING PROBLEM

The vehicle routing problem is the problem of determining a set of routes such that customer demand is satisfied. Each route is performed by a vehicle and consists of a sequence of visits to a set of customers, starting and finishing at a single depot. The objective of the problem is to minimize total cost, while simultaneously meeting operational constraints on the routes, the vehicles, and the customers. There are several variants of the VRP, namely, the distance constrained VRP (DCVRP), the VRP with time windows (VRPTW), the VRP with backhauls (VRPB), the VRP with pickup and delivery (VRPPD), and the most widely known, the capacitated VRP (CVRP). This chapter deals exclusively with the CVRP. There are several good survey papers and books on the topic (Bodin, 1990; Laporte, Gendreau, Potvin, & Semet, 2000; Toth & Vigo, 2002).

Formally, the CVRP is defined on a complete graph $G=(N,E)$, where $N=\{0,1,\ldots,n\}$ is the set of nodes. In the symmetric case, the edge set E is defined by $E=\{(i,j)|i,j\in N; i<j\}$. Also, the cost of traveling from node i to node j, namely c_{ij}, is the same cost of traveling from node j to node i. Node 0 is called the depot,

Figure 5. Gantt chart for period 1 for the best solution found for problem 2



while nodes *1* through *n* represent customer locations. Each customer has associated a non-negative demand d_i ($i \in N - \{0\}$), while the depot (e.g., distribution center) has a demand of zero units ($d_0 = 0$). The depot has available a set of up to *K* identical vehicles with capacity *C*. Without loss of generality, we assume that $d_i \leq C$ ($i \in N$). The CVRP consists of finding a set of at most *K* vehicle routes of total minimum cost, such that every route starts and ends at the depot, each customer is visited exactly once, and the sum of the demands in each vehicle route does not exceed the vehicle's capacity. Note that in our problem definition, we have chosen to treat the number of vehicles as a decision variable.

The CVRP has attracted the attention of many researchers, basically for two reasons: (1) being an NP-hard problem, it is a challenging problem to solve; and (2) it has real-world applications in the context of distribution and logistics. Despite its complexity, some researchers have worked on solving instances of the CVRP to optimality (Agarwal, Mathur, & Salkin, 1989; Araque, Kudva, Morin, & Pekny, 1994; Baldacci, Hadjiconstantinou, & Mingozzi, 2004; Fisher, 1994; Miller, 1995). These researchers have used mathematical programming techniques to solve CVRP instances found in the literature. Among these techniques, the most successful is *branch-and-cut* (Naddef & Rinaldi, 2002). However, the larger reported instance solved to date has only 135 nodes (Augerat et al., 1995), and it has also been reported that these methods can only consistently solve instances of up to 50 nodes.

Due to the CVRP's complexity and the size of instances arising in a real-world distribution setting, heuristics have been widely used for tackling large-sized problems. These heuristics can be broadly organized into two groups: classical heuristics and meta-heuristics. In general, classical heuristics are simple, fast, and produce good-quality solutions. On the other hand, meta-heuristics have several parameters to tune, are time consuming, but produce better solutions. For a review on classical heuristics for the CVRP, the reader is referred to Laporte et al. (2000).

In recent years, researchers have used meta-heuristics extensively to improve the best-known results for the CVRP. In general, meta-heuristics search a larger solution space than classical heuristics. This improved search mechanism often produces better solutions, but it comes at a price: larger computing times. For an introduction to meta-heuristics for combinatorial optimization, the reader is referred to Glover and Kochenberger (2003), Aarts and Lenstra (1997), and Reeves (1993). For a survey on meta-heuristics for the CVRP, the reader is referred to Gendreau, Laporte, and Potvin (2002).

Tabu search has produced the best results for the CVRP. This meta-heuristic produces a sequence of solutions based on a neighborhood definition and the concept of a tabu list. Through this list, solutions that have been recently visited are forbidden, to avoid search inefficiencies. The method can be enhanced by the use of multi-level memory management, aspiration levels, and diversification and intensification strategies. Among tabu search algorithms for the CVRP, it is worth mentioning those by Osman (1993); Taillard (1993); Taburoute by Gendreau, Hertz, and Laporte, (1994); the adaptive memory procedure by Rochat and Taillard (1995); the unified tabu search algorithm (UTSA) by Cordeau, Laporte, and Mercier (2001); and the granular tabu search (GTS) by Toth and Vigo (2003).

Other meta-heuristics such as simulated annealing and ant colony optimization have been proposed for the CVRP, but they have been outperformed by tabu search (Gendreau et al., 2002). Genetic algorithms have performed well on harder versions of the CVRP, such as the CVRPTW, in which clients expect vehicle visits to meet their time windows (Thangiah, 1995; Potvin & Bengio, 1996). Until very recently, analogous results have not been reported on instances of the plain CVRP. This lack of good results has been surprising, but also has motivated several researchers who have recently proposed efficient genetic algorithms for the CVRP (Baker & Ayechew, 2003; Prins, 2004).

A Solution Approach Based on a Genetic Algorithm

Vehicle routing heuristics are often measured by only two attributes: *accuracy* (distance from the best-known solution) and *speed*. However, Cordeau, Gendreau, Laporte, Potvin, and Semet (2002) call for a new generation of heuristics in which *simplicity* and *flexibility* must also be achieved.

The genetic vehicle representation (GVR) proposed by Pereira, Tavares, Machado, & Costa (2002) is a representation scheme for the CVRP. The GVR has proven to be simple and flexible, even though it may not have been thoroughly tested by its authors, nor has it been enhanced by sophisticated local search mechanisms to improve the search of the solution space. In terms of accuracy, the authors reported gaps with respect to best-known solutions of up to 3.5% for the tested instances, which are acceptable, taking into account GVR's simplicity.

The best way to describe the GVR representation is by means of an example. Let us consider instance S013-04e with distance matrix shown in Figure 6. We use the naming convention of Toth and Vigo (2002) for VRP instances, where S means that the instance has a symmetric non-Euclidean distance matrix (E, for Euclidean distance matrix); 013 means that

Figure 6. Distance between nodes for instance S013-04e

c_{ij}	J												
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0		9	14	23	32	50	21	49	30	27	35	28	18
1			21	22	36	52	24	51	36	37	41	30	20
2				25	38	5	31	7	36	43	29	7	6
3					42	12	35	17	44	31	31	11	6
4						22	37	16	46	37	29	13	14
5							41	23	10	39	9	17	16
6								26	21	19	10	25	12
7									30	28	16	27	12
8										25	22	10	20
9											20	16	8
10												10	10
11													10

i

Table 5. Client demands for instance S013-04e

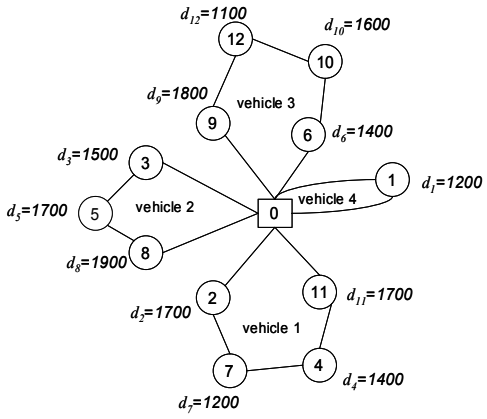
Client (i)	Demand (d_i)
1	1200
2	1700
3	1500
4	1400
5	1700
6	1400
7	1200
8	1900
9	1800
10	1600
11	1700
12	1100

there are 13 nodes, including the depot; 04 means that there are four vehicles available; and e means that the instance comes from the set of problems by Christofides and Eilon (1969). Furthermore, the capacity for each vehicle is 6,000, and the client demands are shown in Table 5.

The optimal solution for instance S013-04e is shown graphically in Figure 7. Figure 8 shows the GVR representation of this solution (with total cost of 247). The solution is formed by a list of vehicle routes (diamonds). Note that each vehicle route in the GVR solution (Figure 7) is represented by a list of nodes in the GVR representation (Figure 8). Because every node starts and ends at the depot, it is not necessary to explicitly add node 0 to the routes. For instance, the second vehicle route departs from the depot (node 0) and visits clients 3, 5, and 8, before returning to the depot. Note that the sequence in which clients are visited form a node sequence or permutation.

The crossover operator in GVR operates as follows. One child is produced by the combination of the genetic material of two parents. The child inherits all the traits from the first parent. Then the second parent, called the donor, provides a small fraction of genetic material (sub-route) which is inserted in the best possible

Figure 7. Graph representation of the optimal solution for instance S013-04e



location. Neither of the two parents are modified in this process. After the insertion, the resulting child may represent an infeasible solution for the CVRP. In a reconstruction phase, feasibility is looked after by eliminating duplicate visits to any client. This crossover operation is illustrated in Figure 9. In the JGA implementation, we show how we enforce the vehicles' capacity.

Four different mutation operations are proposed in the original paper (Pereira et al, 2002): *swap, inversion, insertion,* and *displacement.* For sake of clarity, we will only describe the one that we have implemented, the inversion operator. This operator works as follows: (1) it randomly selects a vehicle route; (2) it randomly picks a sub-route within the vehicle's

Figure 8. GVR representation of the optimal solution for instance S013-04e

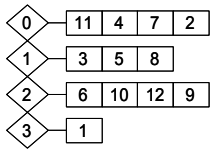


Figure 9. Example of crossover in GVR

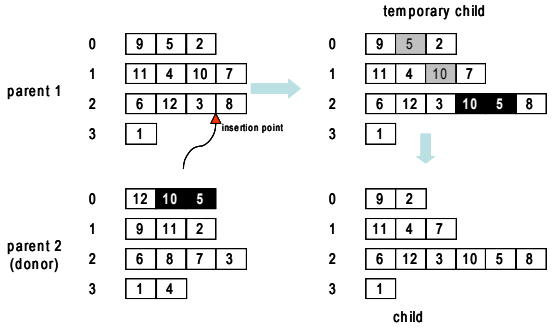
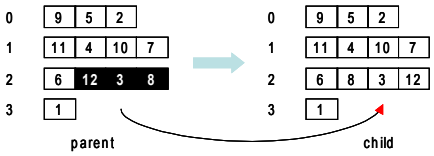


Figure 10. Example of inversion mutation in GVR



route; and (3) it inverts the order of the nodes in the sub-route. Figure 10 illustrates how inversion works in an individual. Note that in this case, only the sequence of visits to clients 12, 3, and 8 in the third vehicle changes to a new sequence, namely, 8, 3, and 12.

The JGA Implementation for the CVRP

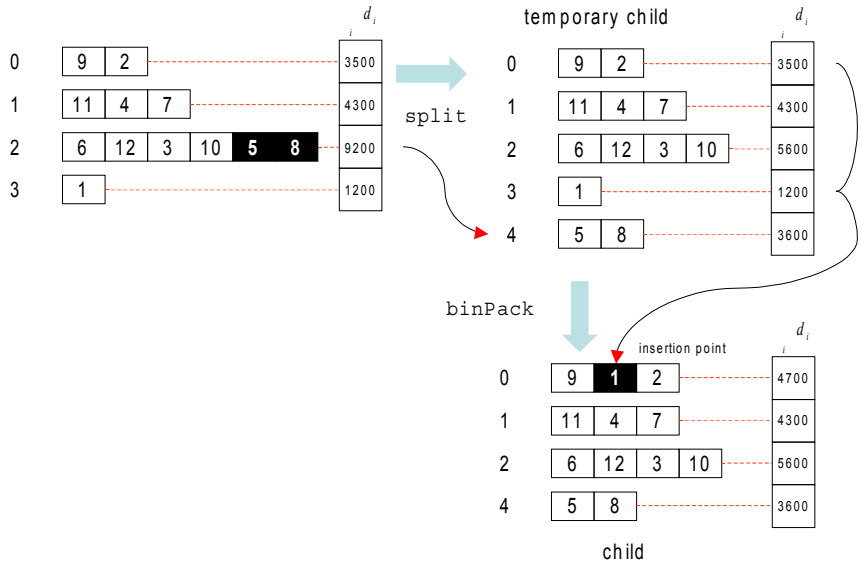
The GVR representation shown in Figure 8 was implemented in GVRGenotype. This encoding inherits from another class that might be part of the JGA library in the near future, namely, MultiPermutationGenotype. The latter contains a dynamic list that holds the vehicle routes. Each route is encoded by a dynamic list of integers representing a sequence of visits to customers. Note that this data structure might be well suited for other combinatorial problems different from the CVRP. The class GVRGenotype inherits from MultiPermutationGenotype and includes a dynamic list with the vehicles' capacities. GVRGenotype also has a set of utility methods that is useful for handling permutations. For instance, some of these methods are pickSubRoute, insertSubRoute, split, and binPack, among others.

Each genotype is evaluated by means of the fitness function class CVRPEvalFunction. This class is in charge of decoding the routes in GVRGenotype by retrieving the sequence of visits to the customers and evaluating the total cost by querying the distance matrix. The result of this evaluation is stored in the individual's phenotype (class SingleFitnessPhenotype).

The crossover for GVRGenotype is implemented in class GVRCrossover. Figure 9 describes how crossover is performed. Pereira et al. (2002) do not clearly describe how an infeasible solution obtained after the crossover operation is repaired. In Figure 11 we show how our JGA implementation handles infeasible children from the crossover operator. In the resulting child of Figure 9, the third vehicle exceeds its capacity (i.e., 9200 > 6000). Using method split, JGA iterates over the vehicles checking for violated capacities. The remaining sub-route, which exceeds the capacity, forms a new vehicle route (see the fifth route, after splitting). Due to the fact that by splitting, JGA may end up with an inefficient number of routes, JGA performs a first-fit packing heuristic (Parker, 1995) using the method binPack.

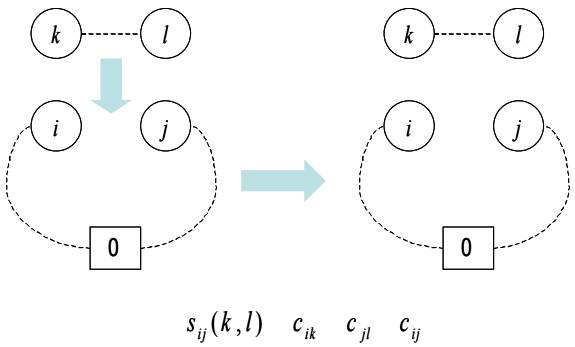
It is worth mentioning that the cost of inserting a sub-route with end nodes k and l into an arc of a route connecting nodes i and j is $s_{ij}(k,l) = c_{ik} + c_{jl} - c_{ij}$. The best insertion could be achieved by exploring all the arcs *(i,j)* in the route. This cheapest insertion mechanism is described in Figure 12.

Figure 11. Genotype reparation in JGA's GVR implementation



The mutation operator for GVRGenotype is implemented in class InversionMutation. Figure 10 shows how inversion mutation has been implemented. The route selection is performed randomly as well as the selection of the sub-route (cut points). Each genotype in the population has a probability of MUTRATE of being chosen for mutation.

Figure 12. Insertion cost of a sub-route into a route



$s_{ij}(k,l) \quad c_{ik} \quad c_{jl} \quad c_{ij}$

We used an elitist selection mechanism (class BestIndividualSelection) that selects the best individuals from an enlarged population. The enlarged population is formed by children produced from crossover and mutation, as well as by individuals from the current population. Table 6 summarizes the JGA components used to solve the CVRP using the GVR representation.

Computational Experiments

We conducted an experiment to compare JGA's GVR against the parallel version of Clarke and Wright (1964) on the set of instances from Christofides and Eilon (1969). Both algorithms were coded in Java and executed using the Java HotSpot™ Client Virtual Machine (version 1.4.2). Clarke and Wright was run on a Dell Optiplex with an Intel Pentium III processor running at 1GHz with 256MB of RAM, whereas JGA's GVR was run on a Dell Optiplex GX280 with an Intel Pentium IV processor running at 3GHz with 1GB of RAM.

Table 6. Summary of JGA classes for the GVR implementation for the CVRP

Component	Class	Extended from Class[1]
Genotype	GVRGenotype.java•	MultiPermutationGenotype♦
Phenotype	SingleFitnessPhenotype♦	*Phenotype*♦
Fitness Function	CVRPEvalFunction•	*FitnessFunction*♦
Mutation Operator	InversionMutation•	*MutationOperator*♦
Crossover Operator	GVRCrossover•	*CrossoverOperator*♦
Selection Operator	BestIndividualSelection♦	*SelectionOperator*♦

[1] *Italics used to indicate abstract classes*

♦ *JGA built-in class*

• *User-defined class*

Table 7 shows the results with the parallel version of Clarke and Wright (1964). We compare against this algorithm because it is a common benchmark and is widely used in commercial applications. As reported in the literature, the algorithm proves to be very fast, always taking less than one second to solve. However, when compared against the best solution found in the literature, its average gap is about 12.0% (setting negative gaps to zero). It is also quite variable, having a standard deviation of 18.7%. For instance, in problem S031-07e, it performs very badly, achieving a solution that is 72.8% off from the best-known solution. On the other hand, it performs very well in a slightly larger instance such as E033-04e, being only 0.8% off from the best-known solution. Note that the fifth column, labeled with K^*, shows the number of vehicles used by the algorithm. In instance E030-03e, the algorithm used one more vehicle than the best-known solution, but it achieved a lower total cost (i.e., traveled distance). That explains the negative gap with respect to the best-known solution in that instance.

Table 7. Results with parallel Clarke and Wright

Instance	Best Solution	Cost	Gap♦	K^*	Time (ms)
S013-04e	247	275	11.3%	4	20
E022-04e	375	388	3.5%	4	50
E023-03e	569	631	10.9%	3	40
E030-03e	534	530	-0.7%	4	70
S031-07e	379	655	72.8%	8	70
E033-04e	835	842	0.8%	4	90
E051-05e	521	580	11.3%	6	180
E076-07e	682	750	10.0%	7	380
E076-08e	735	785	6.8%	8	380
E076-10e	830	894	7.7%	11	341
E076-14e	1021	1071	4.9%	15	340
E101-08e	817	896	9.7%	8	691
E101-14e	1071	1138	6.3%	15	611

♦ *Negative gaps occur because more vehicles were used to reduce the total cost, which is the main objective of the algorithm.*

Table 8 shows the results with the JGA implementation of GVR. For each instance, we conducted 10 independent runs. We set the algorithm parameters as follows: POPSIZE to 100, MAXGEN to 1000, MUTRATE to 0.2, and CROSSRATE to 0.7.We report the average gap, cost, time, number of fitness function evaluations, and number of vehicles used (K^*). We also report the best solution in the set of 10 runs (under the columns labeled Minimum Cost and Minimum Gap). The average gap with respect to the best-known solution is about 4.5% (setting negative gaps to zero), with a standard deviation of 4.3%. When we calculate the average gap with respect to the best solution found in the set of 10 runs, it lowers to just 2.1% (with a standard deviation of 2.1%). Even though JGA's GVR is slower than Clarke and

Table 8. Results with JGA's GVR

Name	Best Solution	Average Cost	Average Gap♦	Minimum Cost	Minimum Gap♦	Average *K**	Average Time (ms)	Average Evaluations
S013-04e	247	248.0	0.4%	247	0.0%	4.0	4,973.4	54,805.4
E022-04e	375	375.7	0.2%	375	0.0%	4.0	6,223.3	54,817.1
E023-03e	569	569.0	0.0%	569	0.0%	3.0	5,788.8	54,781.0
E030-03e	534	504.4	-5.5%	503	-5.8%	4.0	7,701.5	54,768.6
S031-07e	379	438.0	15.6%	404	6.6%	7.0	9,623.5	54,810.5
E033-04e	835	852.9	2.1%	839	0.5%	4.0	8,173.6	54,842.3
E051-05e	521	545.1	4.6%	532	2.1%	5.1	11,806.5	54,813.1
E076-07e	682	723.6	6.1%	700	2.6%	7.0	16,534.4	54,781.8
E076-08e	735	775.2	5.5%	766	4.2%	8.0	16,631.3	54,781.8
E076-10e	830	876.3	5.6%	841	1.3%	10.4	17,323.3	54,781.8
E076-14e	1021	1078.5	5.6%	1050	2.8%	15.0	19,409.3	54,781.8
E101-08e	817	869.1	6.4%	848	3.8%	8.0	20,628.0	54,810.2
E101-14e	1071	1142.1	6.6%	1107	3.4%	14.0	22,439.0	54,810.2

♦ *Negative gaps occur because more vehicles were used to reduce the total cost, which is the main objective of the algorithm.*

Wright's, in the instance that takes the longest to run it averages a reasonable 22.4s.

In conclusion, JGA's GVR implementation shows a genetic algorithm that is flexible, easy to implement, robust, accurate enough (2.1% off from the best known), but slower than the savings algorithm of Clarke and Wright.

CONCLUSION

We have presented two applications of Java Genetic Algorithm (JGA). From a practical side, we have shown how it is possible to use genetic algorithms and JGA for modeling and solving complex business problems arising in operations management and vehicle routing. It is worth mentioning that the JGA-based solutions are often quite competitive, yet easier to implement than the state-of-the-art method of choice. Last, but not least, JGA class libraries, examples, and documentation are available at http://copa.uniandes.edu.co/soft-evol-jga.html

REFERENCES

Aarts, E., & Lenstra, J. K. (Eds.). (1997). *Local search in combinatorial optimization.* West Sussex, UK: John Wiley & Sons.

Agarwal, Y., Mathur, K., & Salkin, H. M. (1989). A set-partitioning-based exact algorithm for the vehicle routing problems. *Networks, 19,* 731-749.

Araque, J. R., Kudva, G., Morin, T. L., & Pekny, J. F. (1994). A branch-and-cut algorithm for vehicle routing problems. *Annals of Operations Research, 50,* 37-59.

Augerat, P., Belenguer, J., Benavent, E., Corberán, A., Naddef, D., & Rinaldi, G. (1995). *Computational results with a branch and cut code for the Capacitated Vehicle Routing Problem*. Technical Report RR949-M, Université Joseph Fourier, France.

Baker, B. M., & Ayechew, M. A. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research, 30,* 787-800.

Baldacci, R., Hadjiconstantinou, E., & Mingozzi, A. (2004). An exact algorithm for the Capacitated Vehicle Routing Problem based on a two-commodity network flow formulation. *Operations Research, 52*(5), 723-738.

Bodin, L. D. (1990). Twenty years of routing & scheduling. *Operations Research, 38*(4), 571-579.

Cheng, T. C. E., & Sin, C. C. S. (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research, 47,* 271-292.

Christofides, N., & Eilon, S. (1969). An algorithm for the vehicle dispatching problem. *Operational Research Quarterly, 20,* 309-318.

Clarke, G., & Wright, J. V. (1964). Scheduling vehicles from a central depot to a number of delivery points. *Operations Research, 12,* 568-581.

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J.-Y., & Semet, F. (2002). A guide to vehicle routing heuristics. *Journal of the Operational Research Society, 53,* 512-522.

Cordeau, J. F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society, 52,* 928-936.

Davis, L. (1991). *Handbook of genetic algorithms.* New York: Van Norstrand Reinhold.

Dixon, P. S., & Silver, E. A. (1979). A heuristic solution procedure for the multi-item, single level, limited capacity lot-sizing problem. *IIE Transactions, 2*(2), 112-123.

Fisher, M. L. (1994). Optimal solution of vehicle routing problems using minimum *K*-trees. *Operations Research, 42,* 626-642.

Gen, M., & Cheng, R. (2000). *Genetic algorithms and engineering optimization.* New York: John Wiley & Sons.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science, 40,* 1276-1290.

Gendreau, M., Laporte, G., & Potvin, J.-Y. (2002). Meta-heuristics for the Capacitated VRP. In P. Toth & D. Vigo (Eds.), *The vehicle routing problem.* Philadelphia: SIAM.

Glover, F., & Kochenberger, G. A. (Eds.). (2003). *Handbook of meta-heuristics.* Kluwer Academic.

Gutiérrez, E. (2002). *An integrated approach for lot-sizing and scheduling problems using meta-heuristics: Genetic algorithms (GAs) and simulated annealing (SA).* ME Dissertation, University of Puerto Rico.

Laporte, G., Gendreau, M., Potvin, J.-Y., & Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research, 7,* 285-300.

Lee, J., & Pinedo, M. (1997). Scheduling jobs on parallel machines with sequence-dependent setup times. *European Journal of Operational Research, 100,* 464-474.

Miller, D. L. (1995). A matching based exact algorithm for capacitated vehicle routing problems. *ORSA Journal on Computing, 7,* 1-9.

Naddef, D., & Rinaldi, G. (2002). Branch-and-cut algorithms for the Capacitated VRP. In P. Toth & D. Vigo (Eds.), *The vehicle routing problem.* Philadelphia: SIAM.

Nahmias, S. (1998). *Production and operations analysis.* Chicago: Irwin.

Osman, I. H. (1993). Meta-strategy simulated annealing and tabu search algorithms for the vehicle routing problems. *Annals of Operations Research, 41,* 421- 451.

Parker, G. (1995). *Deterministic scheduling theory.* London: Chapman & Hall.

Pereira, F. B ., Tavares, J., Machado, P., & Costa, E. (2002). GVR: A new genetic representation for the vehicle routing problem. In M. O'Neil et al. (Eds.), *Proceedings of AICS 2002, the 13th Irish Conference on Artificial Intelligence and Cognitive Science* (pp. 95-102), Limerick, Ireland.

Potvin, J.-Y., & Bengio, S. (1996). The vehicle routing problem with time windows. Part II: Genetic search. *INFORMS Journal on Computing, 8,* 165-172.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research, 31,* 1985-2002.

Reeves, C. (Ed.). (1993). *Modern heuristic techniques for combinatorial problems.* Oxford, UK: Blackwell.

Rochat, Y., & Taillard, E. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics, 1,* 147-167.

Sikora, R., Chhajed, D., & Shaw, M. J. (1996). Integrating the lot-sizing and sequencing decisions for scheduling a capacitated flow line. *Computers and Engineering, 30*(4), 699-679.

Sipper, D., & Bulfin, R. (1997). *Production: Planning, control and integration.* New York: McGraw-Hill.

Taillard, E. D. (1993). Parallel iterative search methods for vehicle routing problems. *Networks, 23,* 661-673.

Thangiah, S. R. (1995). Vehicle routing with time windows using genetic algorithms. *Proceedings of the 6th International Conference on Genetic Algorithms* (pp. 536-543). Los Altos, CA: Morgan Kaufmann.

Toth, P., & Vigo, D. (Eds.). (2002). *The vehicle routing problem.* Philadelphia: SIAM.

Toth, P., & Vigo, D. (2003). The granular tabu search and its application to the vehicle-routing problem. *INFORMS Journal on Computing, 15*(4), 333-346.

KEY TERMS

Chromosome: Data structure (genetic code) which encodes a solution (individual) to the problem that the genetic algorithm is trying to solve.

Crossover: Genetic operator that combines (mates) chromosomes (parents) to produce new ones (offspring). It is analogous to the biological reproduction.

CVRP: Acronym for capacitated vehicle routing problem. The CVRP consists of finding a set of at most *K* vehicle routes of total minimum cost, such that every route starts and ends at the depot, each customer is visited exactly once, and the sum of the demands in each vehicle route does not exceed the vehicle's capacity.

CLSSP: Acronym for the capacitated lot-sizing and scheduling problem. The CLSS problem is to find the number of units of a given product to be produced at a given period (production orders), and the sequence in which the production orders are to be produced in each period.

Fitness Function: Objective function that quantifies the adaptability of an individual. So-

lutions with better fitness values are more likely to survive and join the next generation in a genetic algorithm. It is analogous to the fitness concept in the natural selection process.

Genetic Algorithm (GA): Stochastic (population-based) search technique, inspired in the natural selection process and used to find approximate solutions to complex optimization problems.

Genotype: Genetic makeup of an individual.

JGA: Acronym for Java Genetic Algorithm, a flexible and extensible computational object-oriented framework for rapid development of evolutionary algorithms for solving complex optimization problems.

Mutation: Genetic operator that alters one or more genes in a chromosome. Mutation is used to maintain genetic diversity in the population of individuals. It is analogous to biological mutation.

Phenotype: Features or quantifiable measurements of an individual. In a genetic algorithm, the phenotype is the value associated with the fitness function evaluation.

Selection: Operation in charge of choosing which individuals from the current population will survive and become part of the next generation.

Chapter XLII
Solving Facility Location Problems with a Tool for Rapid Development of Multi-Objective Evolutionary Algorithms (MOEAs)

Andrés L. Medaglia
Universidad de los Andes, Colombia

Eliécer Gutiérrez
Universidad de los Andes, Colombia

Juan Guillermo Villegas
Universidad de Antioquia, Colombia

ABSTRACT

The low price of coffee in the international markets has forced the Federación Nacional de Cafeteros de Colombia (FNCC) to look for cost-cutting opportunities. An alternative that has been considered is the reduction of the operating infrastructure by closing some of the FNCC-owned depots. This new proposal of the coffee supplier network is supported by (uncapacitated and capacitated) facility location models that minimize operating costs while maximizing service level (coverage). These bi-objective optimization models are solved by means of NSGA II, a multi-objective evolutionary algorithm (MOEA). From a computational perspective, this chapter presents the multi-objective Java Genetic Algorithm (MO-JGA) framework, a new tool for the rapid development of MOEAs built on top of the Java Genetic Algorithm (JGA). We illustrate MO-JGA by implementing NSGA II-based solutions for the bi-objective location models.

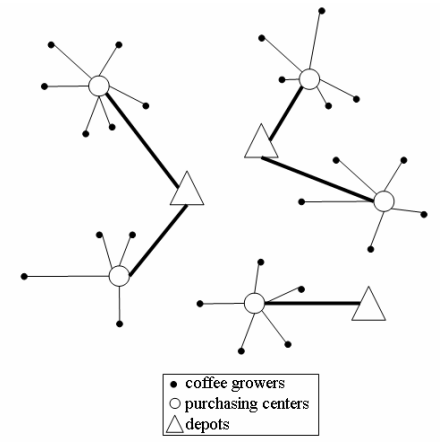
THE CASE OF THE COLOMBIAN COFFEE SUPPLIER NETWORK

Colombia is the second largest coffee producer in the world. The Colombian National Coffee Growers Federation (in Spanish, Federación Nacional de Cafeteros de Colombia—FNCC), is a nonprofit private organization whose main activities include: supporting worldwide marketing activities for the Café de Colombia and Juan Valdez® brands; conducting research on coffee-related topics; providing technical and financial assistance to the coffee growers; guaranteeing the quality of the Colombian coffee exports; and buying, storing, processing, and exporting Colombian top-quality coffee. During the last six years, the FNCC exported about 30% of the Colombian coffee production (FNCC, 2005).

The coffee supplier network operates as follows. First, the coffee growers sell their crops to purchasing centers located in towns nearby their farms. Once these centers—which are cooperatives owned by coffee growers—have collected enough coffee, they send them to a larger FNCC-owned depot. Coffee is stored in these depots until the time comes for processing followed by exportation. This supplier network is composed of over 400 purchasing centers and over 30 FNCC owned depots. Figure 1 shows the structure of the coffee supplier network.

Due to the low prices of coffee in international markets, the FNCC has been looking for opportunities to reduce operating costs on its supplier network. One of the possible alternatives that has been considered is to reduce the operating infrastructure by shutting down some of the FNCC-owned depots (CAIC, 2002). However, by doing so, it may not always be possible to ship coffee from a purchasing center to a nearby FNCC-owned depot. This guarantee of being covered by a nearby depot is deeply appreciated among cooperatives (purchasing centers), and ultimately, among coffee growers who own the FNCC.

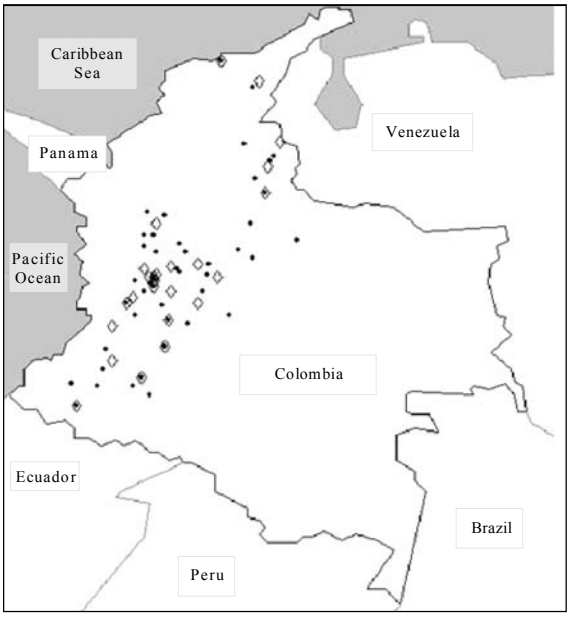
Figure 1. Coffee supplier network



Following the same approach of Bramel and Simchi-Levi (1997), the 450 purchasing centers were aggregated into 47 clustered purchasing centers by considering both distance and purchasing volume. Each purchasing center is represented by its main purchasing agencies (from one to three agencies per coop), and the total amount of coffee purchased from the coffee growers was consolidated into those agencies. The supply for each purchasing center comes from the operation of year 2001 (FNCC, 2001). The location of the purchasing agencies and their distance to the FNCC-owned depots are known, and the covering distance was set to 150 kilometers. After consolidating the storage capacity of the depots located in the same town, the supplier network ends up with a total of 25 candidate depots. Figure 2 shows the geographical distribution of the purchasing centers and depots after consolidation.

Clearly, reducing costs and operating depots located near the purchasing centers are two conflicting objectives. Making the supplier

Figure 2. Aggregated purchasing centers and depots in the Colombian coffee supplier network



network more efficient may imply a reduction in the number of depots, thus causing purchasing centers to be attended by further depots. On the other hand, guaranteeing coverage by nearby depots implies having more depots and larger operating costs. This situation calls for the development of a bi-objective facility location model.

This chapter is organized as follows. The next section presents an overview of facility location models and presents the bi-objective (uncapacitated and capacitated) facility location problem arising in the context of the coffee supplier network. We then present a powerful bio-inspired methodology for solving multi-objective optimization problems, namely, multi-objective evolutionary algorithms (MOEAs). The next section presents MO-JGA, the acronym for the Multi-Objective Java Genetic Algorithm, a tool that extends the JGA framework presented in the previous two chapters of this book. We illustrate MO-JGA through the implementation of NSGA-II, a powerful MOEA. We apply NSGA-II on the bi-objective (capacitated and uncapacitated) location problem arising in the context of the Colombian coffee supplier network.

BI-OBJECTIVE (UNCAPACITATED AND CAPACITATED) FACILITY LOCATION PROBLEMS

Location problems are concerned with finding a set of facilities in a given space that is able to satisfy customers' demand for a given good or service. This type of location decision arises both in public and private organizations. In the public setting, some examples are the location of health care facilities (Verter & Lapierre, 2002), schools (Pizzolato, Barros, Barcelos, & Canen, 2004), and solid waste landfills (Antunes, 1999). In the private setting, some examples are the design of distribution systems (Aikens, 1995) and computer networks (Filho & Galvão, 1998). For an in-depth presentation of location models and solving techniques, the reader is referred to ReVelle, Marks, and Liebman (1970); Aikens (1985); Brandeau and Chiu (1989); Owen and Daskin (1998); Hale and Moberg (2003); Klose and Drexl (2005); ReVelle and Eiselt (2005); Daskin (1995); and Drezner and Hamacher (2001).

According to Daskin (1995), facility location problems can be classified based on the characteristics of their underlying components. For instance, we could form a taxonomy based on the space of location decisions (i.e., continuous, discrete, or network); the objective function (e.g., coverage, cost, or equity); number of objectives (i.e., single or multi-objective); number of products (i.e., single or multi-commodity); type of customer demand (i.e., static or

dynamic); and facility capacity (i.e., uncapacitated or capacitated), among other characteristics. Even though in this chapter we are interested in solving a bi-objective location problem, we use several single-objective discrete static location problems as building blocks, namely, the uncapacitated facility location problem (UFLP), the single source capacitated facility location problem (SSCFLP), and the maximal covering location problem (MCLP). It is worth mentioning that the UFLP is also known as the simple plant location problem (SPLP) or the uncapacitated warehouse location problem (UWLP).

The facility location problems in this chapter share a set of common elements. Let $I = \{1, ..., m\}$ be the candidate sites for the facilities and f_i be the fixed cost of operating facility i. Let $J = \{1, ..., n\}$ be the set of customers and d_j be the demand for customer j. Let c_{ij} be the cost of attending the whole demand for customer j from facility i. Let y_i be the decision variable that indicates if facility i is chosen ($y_i=1$) or not ($y_i=0$); and let x_{ij} be the decision variable that indicates if the (whole) demand for customer j is attended by facility i ($x_{ij}=1$) or not ($x_{ij}=0$). Using this notation, the UFLP can be formulated as follows:

$$\min \sum_{i\in I} f_i y_i + \sum_{i\in I}\sum_{j\in J} c_{ij} x_{ij} \tag{1}$$

subject to,

$$\sum_{i\in I} x_{ij} = 1, \quad j \in J \tag{2}$$

$$x_{ij} \le y_i; \quad i \in I, j \in J \tag{3}$$

$$y_i \in \{0,1\}, \quad i \in I \tag{4}$$

$$x_{ij} \in \{0,1\}, \quad i \in I, j \in J \tag{5}$$

The objective function 1 represents the total operation cost, where the first term is the fixed cost of operating the facilities, and the second term represents the cost of attending customers from opened facilities. Constraint 2 guarantees that every customer is attended by one facility, constraint 3 assigns customers to opened facilities, and constraints 4 and 5 establish the binary nature of the decisions.

The SSCFLP formulation is obtained by substituting constraint 3 with the following constraint:

$$\sum_{j\in J} d_j x_{ij} \le y_i s_i, \quad i \in I \tag{6}$$

Constraint 6 enforces the assigned demand to facility i not to exceed its capacity s_i. Constraints 2 and 5 force each customer to be satisfied by just one facility (i.e., single source).

In contrast to UFLP and SSCFLP, where the underlying objective is cost minimization, the MCLP finds a set of facilities with cardinality $p \le |I|$, such that the largest number of customers (weighted by their demand d_j) are satisfied from facilities within a distance h_{max}. Let Q_j be the set of facilities able to attend customer j within the covering distance h_{max}; namely, $Q_j = \{i \in I : h_{ij} \le h_{\max}\}$, where h_{ij} is the distance between facility i and customer j. Let w_j be a variable that indicates if customer j is covered ($w_j=1$) or not ($w_j=0$). The MCLP formulation follows:

$$\max \sum_{j\in J} d_j w_j \tag{7}$$

subject to,

$$\sum_{i\in Q_j} y_i \ge w_j, \quad j \in J \tag{8}$$

$$\sum_{i\in I} y_i = p \tag{9}$$

$$y_i \in \{0,1\}, \quad i \in I \tag{10}$$

$$w_j \in \{0,1\}, \quad j \in J \tag{11}$$

The objective function 7 maximizes covered demand; constraints 8 and 11 determine if a given customer is covered (or not) by a facility within the covering distance h_{max}; constraint 9 guarantees that exactly p facilities are opened; and constraint 10 establishes the binary nature of opening/closing decisions.

The strategic nature of location decisions and their long-term impact imply that in many situations it might be desirable to consider multiple (possibly conflicting) criteria. For example, in a distribution system design problem, it is of interest to consider both cost and timely response as objectives (Hapke, Jaszkiewicz, & Zak, 2002). Even though many of the real-world location decisions are inherently multi-objective (ReVelle & Eiselt, 2005), there is still a lot of room for improvement (Klose & Drexl, 2005). Current, Min, and Schilling (1990) have made a thorough survey of model formulations and objectives considered in multi-objective location problems. Multi-objective location problems have been treated in Brimberg and ReVelle (1998); Jayaraman (1999); Nozick (2001); Nozick and Turnquist (2001); Hapke et al. (2002); and Villegas, Palacios, and Medaglia (2006).

For the case study presented in this chapter, we use a bi-objective model similar to the one used by ReVelle and Laporte (1996). By combining elements from the UFLP and MCLP, the bi-objective location model in this chapter minimizes the total operating cost and maximizes the total demand attended within covering distance. The bi-objective uncapacitated facility location problem (BOUFLP) follows:

$$\min \; z_1 = \sum_{i \in I} f_i y_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \tag{12}$$

$$\max \; z_2 = \sum_{j \in J} d_j \sum_{i \in Q_j} x_{ij} \tag{13}$$

$$\text{subject to} \; \sum_{i \in I} x_{ij} = 1, \quad j \in J \tag{14}$$

$$x_{ij} \le y_i; \quad i \in I, j \in J \tag{15}$$

$$y_i \in \{0,1\}, \quad i \in I \tag{16}$$

$$x_{ij} \in \{0,1\}; \quad i \in I, j \in J \tag{17}$$

From the BOUFLP, it is worth mentioning that objective function (13) measures coverage by adding up demand attended by facilities within the covering distance. That is, a given customer j is considered covered if and only if it is attended by a facility from the set Q_j.

A capacitated version of the bi-objective capacitated facility location problem (BOCFLP) is obtained by replacing constraint 15 by capacity constraint 6. The model formulation for the BOCFLP follows:

$$\min \; z_1 = \sum_{i \in I} f_i y_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \tag{18}$$

$$\max \; z_2 = \sum_{j \in J} d_j \sum_{i \in Q_j} x_{ij} \tag{19}$$

subject to,

$$\sum_{i \in I} x_{ij} = 1, \quad j \in J \tag{20}$$

$$\sum_{j \in J} d_j x_{ij} \le y_i s_i, \quad i \in I \tag{21}$$

$$y_i \in \{0,1\}, \quad i \in I \tag{22}$$

$$x_{ij} \in \{0,1\}; \quad i \in I, j \in J \tag{23}$$

Meta-heuristics have been used for several single-objective facility location problems, for example: simulated annealing (Golden & Skiscim, 1986), heuristic concentration (Rosing & ReVelle, 1997), GRASP (Resende, 1998), evolutionary algorithms (Jaramillo, Badhury, & Batta, 2002), and tabu search (Michel & Van Hentenryck, 2004), among others.

Evolutionary algorithms have shown to be effective dealing with location problems. Examples of applications of evolutionary algorithms to location problems are: UFLP (Kratica, Tosic, Filipovic, & Ljubic, 2001), *p*-median (Alp, Erkut, & Drezner, 2003), MCLP (Jaramillo et al., 2002), and SSCFLP (Cortinhal & Captivo, 2001).

Even though evolutionary algorithms have shown to be competitive in single-objective location applications, they have even better potential in the multi-objective setting shown in this chapter. Multi-objective evolutionary algorithms are simple, yet flexible and robust. We will show in this chapter how easy it is to decouple location decisions (variables y_i) from assignment decisions (variables x_{ij}) in an evolutionary algorithmic framework. Finally, we will show how easy is to deal with both uncapacitated and capacitated versions of a difficult combinatorial problem within the same algorithmic framework, something that is quite challenging in a more traditional mathematical programming approach.

INTRODUCTION TO MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS

Traditionally, many decision-making problems have been modeled using single-objective optimization, that is, the main concern is to optimize a single objective such as cost, revenue, traveled distance, or investment return, among others. However, in many of these problems it is desirable to consider multiple optimization criteria, thus forcing the decision maker to consider multi-objective optimization models. When considering multiple objectives, it is often the case that there is no single optimal solution that achieves all the objectives simultaneously, therefore it is necessary to search for a set of efficient solutions which present the best tradeoff among the multiple criteria. These efficient solutions are also known as non-dominated or Pareto solutions. By definition, a Pareto solution is one in which it is not possible to improve upon an objective function without deteriorating another. For an up-to-date treatment on multi-objective optimization, the reader is referred to Ehrgott and Gandibleux (2002), and Collette and Siarry (2003).

According to Ehrgott and Gandibleux (2004), the practical significance of multi-objective optimization problems and their computational complexity have generated an increasing interest in the development of approximate solution methods. Several variations of the most popular meta-heuristics have been proposed to tackle multi-objective optimization problems, namely, tabu search (Gandibleux, Mezdaoui, & Fréville, 1997), simulated annealing (Ulungu, Teghem, Fortemps, & Tuyttens, 1999), ant colony optimization (Doerner, Gutjahr, Hartl, Strauss, & Stummer, 2004), and evolutionary algorithms (Coello, 2001). For updated surveys on meta-heuristics for multi-objective optimization, the reader is referred to Jones, Mirrazavi, and Tamiz (2002); Ehrgott and Gandibleux (2004); and Gandibleux, Sevaux, Sörensen, and T'kindt (2004).

According to Jones et al. (2002), the most broadly used multi-objective meta-heuristics are multi-objective evolutionary algorithms. The pioneering work on MOEA was the vector evaluated genetic algorithm (VEGA) developed by Schaffer (1985). Even though some

authors have developed MOEAs based on *a priori* articulation of preferences (Murata, Ishibuchi, & Tanaka, 1996), the most widely used MOEAs are based on *a-posteriori* articulation of preferences. One of the reasons for this trend is that being MOEA population-search algorithms, they are particularly powerful discovering (in parallel) and maintaining the efficient frontier in their population in a single run (Coello, 2001; Jones et al., 2002; Zitzler & Thiele, 1999).

Generally speaking, it is desirable to have MOEAs that incorporate efficiency or dominance criteria in their evaluation and selection mechanisms, so that the population evolves toward the efficient frontier. It is equally important to favor mechanisms that maintain population diversity and ultimately produce an evenly spread frontier. Finally, there is a general agreement in the MOEA community about the convenience of using elitism and information from an archive of non-dominated solutions for improving the performance of MOEAs.

The different ways of handling evaluation, selection, population diversification, elitism, archived non-dominated solutions, and hybridization with other algorithms have driven the MOEA community to propose several algorithms. The most popular MOEAs are:

- the multiple objective genetic algorithm (MOGA) by Fonseca and Fleming (1993);
- the niched Pareto genetic algorithm (NPGA) by Horn, Nafpliotis, and Goldberg (1994);
- the nondominated sorting genetic algorithm (NSGA) by Srinivas and Deb (1994), and the NSGA II (Deb, Pratap, Agarwal, & Meyarivan, 2002), which includes elitism and improves upon the computational complexity of the non-dominated sorting mechanism;
- the strength Pareto evolutionary algorithm (SPEA) by Zitzler and Thiele (1999), and the improved SPEA2 (Zitzler, Laumanns, & Thiele, 2002);
- the Pareto archived evolutionary strategy (PAES) by Knowles and Corne (2000), and the memetic algorithm variant known as M-PAES (Knowles & Corne, 2000);
- the Pareto-envelope-based selection algorithm (PESA) by Corne, Knowles, and Oates (2000);
- the multiple objective genetic local search (MOGLS) by Jaszkiewicz (2002); and
- the micro genetic algorithm for multi-objective optimization (micro-GA) by Coello and Toscano (2001).

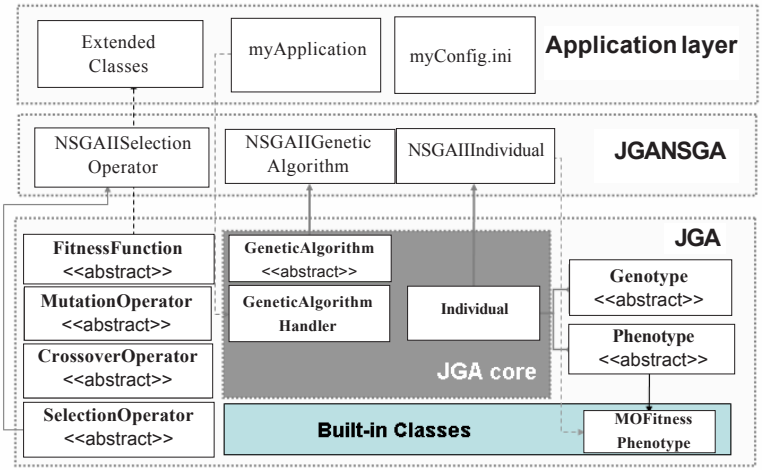
For more detail on these algorithms, the reader is referred to the evolutionary multi-objective optimization (EMOO) Web page maintained by Coello (2005) and the books by Coello, Van Veldhuizen, and Lamont (2002), and Deb (2001).

MO-JGA: EXTENDING JGA TO HANDLE MOEAS

MO-JGA (multi-objective Java Genetic Algorithm) is a computational object-oriented framework for solving complex multi-objective optimization problems using evolutionary algorithms. The tool extends the JGA framework presented in the two previous chapters of this book, and similarly to the single-objective case, it provides the user with a set of built-in components for multi-objective evolutionary algorithms. In this chapter, MO-JGA is illustrated with the implementation of an efficient MOEA known as NSGA-II (Deb et al., 2002).

Similar to JGA, MO-JGA allows the user to extend the fundamental components of a genetic algorithm, such as the genotype, phenotype, fitness function, and the genetic (mutation and crossover) operators. As will be illustrated

Figure 3. MO-JGA architecture for NSGA-II



with NSGA-II in this chapter, this framework can be used to implement other MOEAs or to develop applications based on the current MOEA implementations. Figure 3 shows a diagram that illustrates the MO-JGA architecture and the implementation of NSGA-II.

The bottom layer in Figure 3 corresponds to the JGA components that are in charge of the application's control and provide the basic built-in classes. JGA also contains the application program interfaces (APIs) for the extensible components (e.g., Genotype) by means of abstract classes.

The middle layer in Figure 3 corresponds to the NSGA-II implementation, including the main logic for the NSGA-II (class NSGAIIGeneticAlgorithm) and the NSGA-II selection mechanism (class NSGAIISelectionOperator) based on Pareto's dominance. These classes are extended from the JGA abstract classes *GeneticAlgorithm* and *SelectionOperator*, respectively.

The NSGA-II implementation defines a specific data structure for the individual, namely NSGAIIIndividual. This class extends from JGA's Individual, which includes additional information pertaining to NSGA-II (e.g., non-dominated sorting counters). The NSGA-II implementation employs a multi-objective valued phenotype defined in JGA called MOFitnessPhenotype. This phenotype applies the Pareto's dominance concept to compare pairs of phenotypes.

Finally, the top layer in Figure 3 corresponds to the specific application implemented by the user. Examples of such an application are the implementations for the bi-objective facility location problems included in this chapter. In the application layer, the user could make use of the built-in components provided by JGA or could extend new components for solving a given problem.

According to the JGA framework, the user creates in the main program an object of type GeneticAlgorithmHandler and invokes its run method to start the execution of the genetic algorithm as illustrated in Figure 4. To use a particular implementation of an MOEA, some structural properties must be set in the *main configuration file*. Table 1 shows the property fields and their values defined in the NSGA-II implementation. All the other properties can be

Figure 4. Example of the main program (in Java) for a NSGA-II-based application

```
1:    import edu.uniandes.copa.jga*;
2:    import edu.uniandes.copa.mojga.nsga2.*;
3:    public class myApplication {
4:        public static void main (String args[]) {
5:            String configFileName = args [0]
6:            ArrayList genotypeParams = new ArrayList();
7:            // genotypeParams construction . . .
8:            GeneticAlgorithmHandler ga = new GeneticAlgorithmHandler (configFileName,genotypeParams)
9:            ga.run()
10:       }
11:   }
```

Table 1. Structural property fields in the main configuration file for NSGA-II

Property Field	Value
GENETICALGORITHM	edu.uniandes.copa.mojga.nsga2.NSGAIIGeneticAlgorithm
PHENOTYPE	edu.uniandes.copa.jga.MOFitnessPhenotype
SELECTION	edu.uniandes.copa.mojga.nsga2.NSGAIISelection

set by the user according to the specification defined in the JGA framework (see the previous two chapters in this book).

A MO-JGA IMPLEMENTATION OF NSGA-II FOR BI-OBJECTIVE FACILITY LOCATION PROBLEMS

The most important issues in the implementation of an evolutionary algorithm are the solution encoding and the genetic operators. This section provides the key design aspects shared by the evolutionary algorithms for the bi-objective (uncapacitated and capacitated) facility location problems.

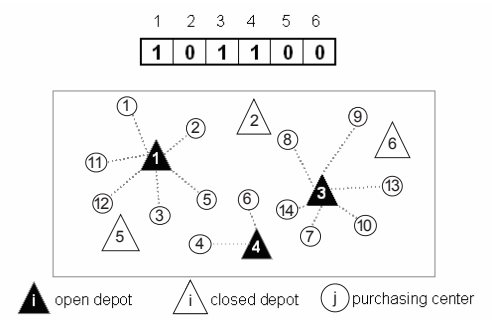
Solution Representation

In the literature, there are mainly two types of encoding schemes used for facility location problems, namely, integer and binary representations. The integer representation is the most natural encoding, consisting of a string with n integer digits mapping to each one of the n customers (i.e., the j-th position indicates the depot assigned to client j). This representation scheme was used by Chu and Beasley (1997) for the Generalized Assignment Problem (GAP), and Cortinhal and Captivo (2001) and Zhou, Min, and Gen (2003) for location problems.

In the binary representation (Kratica et al., 2001; Jaramillo et al., 2002; Villegas et al., 2006), the chromosome consists of m binary digits, corresponding to the m depots. Gene i of a chromosome indicates if depot i is open (value of 1) or closed (value of 0). This representation defines a set of opened depots and requires a procedure to make the detailed assignment of clients to depots. The advantage of the binary representation over the integer representation is the reduction of the search space.

Our implementation uses the binary representation along with a greedy heuristic for making the assignment of purchasing centers to depots. The assignment heuristic tries to minimize the total cost without deteriorating coverage. For the capacitated case, while making the assignments the heuristic also verifies the available capacity to satisfy the capacity constraints. Figure 5 illustrates the binary encoding for an example with 6 depots and 14 purchasing cen-

Figure 5. Binary encoding for location problems



ters. The example includes the assignment performed by the heuristic.

Fitness Evaluation and Dominance

The bi-objective facility location problem presented in this chapter deals with the optimization of two conflicting objectives: minimizing total costs and maximizing coverage. Both objective functions z_1 and z_2 are evaluated after decoding the chromosomes according to equations 12 and 13 for the BOUFLP (and 18 and 19 for the BOCFLP). In the NSGA-II implementation, the concept of Pareto´s dominance is applied on the z_1 and z_2 values.

For the capacitated case, the capacity constraint is handled by penalizing the total cost according to equations 24 and 25, where *P(x)* is the penalty function, and α is a penalty factor ($\leq \alpha \leq 2$). Penalization occurs when assigned customer demand exceeds the depot capacity.

$$z_1 = z_1 P(x) \tag{24}$$

$$P(x) = \begin{cases} \alpha + \sum_{i \in I} \max\left(0, \sum_{j \in J} d_j x_{ij} - s_i\right) \Big/ \sum_{i \in I} s_i & \text{if } \sum_{i \in I} \max\left(0, \sum_{j \in J} d_j x_{ij} - s_i\right) > 0 \\ 1 & \text{if } \sum_{i \in I} \max\left(0, \sum_{j \in J} d_j x_{ij} - s_i\right) = 0 \end{cases} \tag{25}$$

Genetic Operators

The evolutionary process uses mutation to diversify the search and to avoid getting trapped on local optima. Our implementation employs flip binary mutation, in which every gene of the chromosome is flipped from 1 to 0 (or from 0 to 1) with a given mutation probability MUTRATE.

The NSGA-II's crossover implementation provided by MO-JGA builds a pool in which each individual has a chance of being recombined equal to the crossover probability CROSSRATE. We select the uniform crossover operator described by Michalewicz (1996). In the uniform crossover, every child's gene is provided randomly by either one of the two parents.

MO-JGA Implementation of NSGA-II

The bi-objective evolutionary algorithm was developed using the NSGA-II implementation provided by MO-JGA. Classes for genotype, phenotype, and genetic operators were selected directly from the built-in components provided by JGA. Table 2 summarizes the implemented classes and the built-in components used from MO-JGA and JGA to solve the bi-objective facility location problems described in this chapter.

Fitness function classes were implemented for both the uncapacitated and capacitated version of the problem (classes BOUFLPFitness and BOCFLPFitness, respectively). Both problems use the binary encoding implemented in class BinaryGenotype from JGA. The result of the fitness function evaluation is stored in a multi-objective phenotype (class MOFitnessPhenotype). The genetic algorithm logic and the selection operator were defined in classes NSGAIIGeneticAlgorithm and NSGAIISelection, respectively. For further detail on the evolutionary algorithm and a math-

Table 2. Summary of classes used for the bi-objective (uncapacitated and capacitated) facility location problems

Component	Class	Extended from Class[1]
Genotype	BinaryGenotype♦	*Genotype*♦
Phenotype	MOFitnessPhenotype♦	*Phenotype*♦
Fitness Function	BOUFLPFitness• BOCFLPFitness•	*FitnessFunction*♦
Mutation Operator	FlipBinaryMutation♦	*MutationOperator*♦
Crossover Operator	UniformBinaryCrossover♦	*CrossoverOperator*♦
Genetic Algorithm	NSGAIIGeneticAlgorithm ■	*GeneticAlgorithm*♦
Selection Operator	NSGAIISelection ■	*SelectionOperator*♦

[1] *Italics are used to indicate abstract classes*
♦ *JGA built-in class*
■ *MO-JGA built-in class*
• *User-defined class*

ematical-programming approach for the BOUFLP, the reader is referred to Villegas et al. (2006).

COMPUTATIONAL EXPERIMENTS

The MO-JGA implementation of NSGA-II for the BOUFLP and BOCFLP, described in the previous section, was used to generate alternatives for the redesign of the Colombian coffee supplier network. For both the uncapacitated and capacitated scenarios, the algorithm parameters were set as follows: population size (POPSIZE) equal to 100, maximum number of generations (MAXGEN) equal to 100, probability of crossover (CROSSRATE) equal to 0.8, and probability of mutation (MUTRATE) equal to 0.01.

Due to the inherent stochastic nature of evolutionary algorithms, we conducted 20 independent runs for each scenario, namely, BOUFLP and BOCFLP. At the end of each run, the approximate Pareto frontier was stored. For each scenario, we report the (aggregated) approximate Pareto frontier after selecting the non-dominated solutions based on the stored frontiers coming from the 20 independent runs. The average and maximum CPU time for the BOUFLP was 2,608 ms and 2,843 ms, respectively; while the average and maximum CPU time for the BOCFLP was 2,854 ms and 3,032 ms for the BOCFLP, respectively. These runs were conducted on an Intel® Pentium® processor running at 3GHz with 1GB of RAM on Windows XP Professional.

For the BOUFLP, we found an approximate Pareto frontier with 39 different configurations of the supplier network (see Figure 6). As reference, the current configuration is represented by a square in the frontier. Table 3 shows the efficient solutions in the objective space (i.e., cost and coverage). The first and fifth columns show the solution ID; the second and sixth columns correspond to coverage, as a percent of the total demand of the purchasing centers attended by depots located within the covering distance; the third and seventh columns show the cost as a percentage above the minimal cost configuration, found by solving integer programming models of the UFLP and SSCFLP using Xpress-MP; and finally, the

Figure 6. Approximate Pareto frontier for the BOUFLP (uncapacitated)

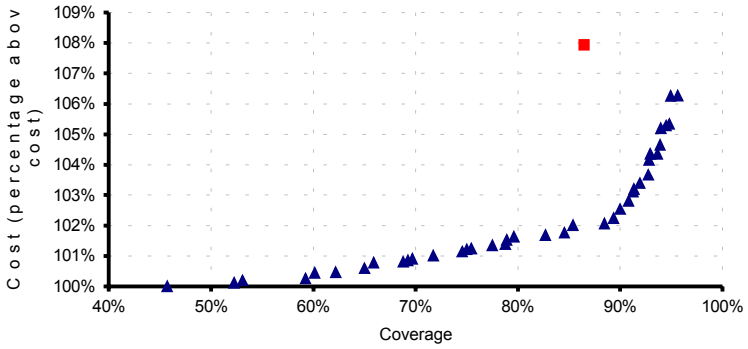


Table 3. Efficient solutions (in objective space) for the BOUFLP

Solution ID	Coverage	Cost	Exceeded capacity	Solution ID	Coverage	Cost	Exceeded capacity
1	45.73%	100.00%	39.45%	21	84.53%	101.76%	22.38%
2	52.23%	100.12%	38.17%	22	85.41%	102.00%	21.46%
3	53.11%	100.19%	36.56%	23	88.47%	102.06%	22.25%
4	59.23%	100.26%	37.25%	24	89.34%	102.24%	20.63%
5	60.10%	100.44%	35.64%	25	89.97%	102.53%	22.49%
6	62.17%	100.47%	36.66%	26	90.85%	102.80%	20.87%
7	65.00%	100.60%	34.31%	27	91.27%	103.10%	20.08%
8	65.87%	100.78%	32.70%	28	91.36%	103.20%	22.22%
9	68.77%	100.80%	36.39%	29	91.90%	103.40%	21.94%
10	69.21%	100.85%	23.64%	30	92.77%	103.67%	20.32%
11	69.65%	100.91%	34.77%	31	92.81%	104.16%	20.65%
12	71.72%	101.01%	35.80%	32	92.94%	104.35%	19.63%
13	74.55%	101.13%	33.45%	33	93.68%	104.35%	19.04%
14	74.99%	101.22%	23.24%	34	93.92%	104.65%	19.78%
15	75.42%	101.25%	31.83%	35	93.96%	105.19%	20.12%
16	77.49%	101.34%	32.86%	36	94.51%	105.28%	18.89%
17	78.76%	101.39%	22.77%	37	94.83%	105.33%	18.50%
18	78.92%	101.52%	23.11%	38	94.92%	106.26%	18.87%
19	79.64%	101.63%	21.86%	39	95.66%	106.26%	18.36%
20	82.70%	101.69%	22.64%				

fourth and eighth columns show the percentage in which the capacity is exceeded (if these constraints were imposed).

If we could increase depot capacities, it is worth mentioning that coverage could increase up to 9.1%. Note that in Table 3, at least 17 configurations in the approximate Pareto frontier offer better coverage than the current configuration (i.e., 86.49 %). By increasing depot capacities, the maximal coverage configuration could operate with just 12 depots, opposed to the current configuration that operates with 25. Furthermore, this could be achieved with fewer depots without sacrificing service level to the purchasing centers (i.e., coverage). This slimmer non-

Figure 7. Maximal covering solution for the BOUFLP (uncapacitated)

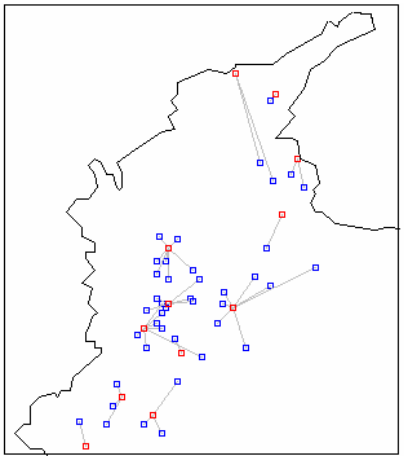


Table 4. Efficient solutions (in objective space) for the BOCFLP

Solution ID	Coverage	Cost	Solution ID	Coverage	Cost
1	56.73%	100.00%	15	77.68%	103.55%
2	57.88%	100.43%	16	78.42%	103.87%
3	62.50%	100.64%	17	82.05%	103.99%
4	62.75%	100.73%	18	82.88%	104.79%
5	63.24%	100.80%	19	83.55%	104.81%
6	63.62%	100.98%	20	84.29%	105.12%
7	68.91%	101.04%	21	84.70%	105.47%
8	73.52%	101.24%	22	85.12%	105.97%
9	74.35%	102.05%	23	85.53%	106.28%
10	75.03%	102.06%	24	86.20%	106.30%
11	75.77%	102.38%	25	86.95%	106.61%
12	76.18%	102.73%	26	87.03%	107.10%
13	76.60%	103.22%	27	87.77%	107.46%
14	77.26%	103.25%	28	87.94%	108.40%

redundant maximal covering solution is depicted in Figure 7.

The second part of this analysis is concerned with the BOCFLP, where capacity constraints are enforced to satisfy current depot capacities. From the 20 independent runs, we found an approximate Pareto frontier with 28 different configurations of the constrained supplier network (see Figure 8). Again, as reference, the current configuration is represented by a square in the frontier. Table 4 shows the efficient solutions in the objective space (i.e., cost and coverage) following the same convention used in Table 3. Due to the fact that capacity constraints on the depots are enforced in the BOCFLP, there is no column showing exceeded capacity.

From Figure 8, it is clear that the left side of the frontier is quite flat (up to solution 8), meaning that coverage can be improved significantly with just a little investment above the minimal cost configuration.Table 4 shows that by investing only 1.26% above the minimum cost configuration, it is possible to increase coverage by 16.79% (see solution 8). Furthermore, with an investment of 3.99% above the

Figure 8. Approximate Pareto frontier for the BOCFLP (capacitated)

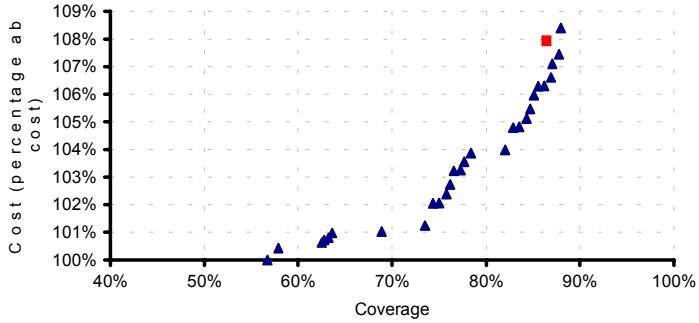


Figure 9. Efficient frontier aproximations for BOUFLP and BOCFLP

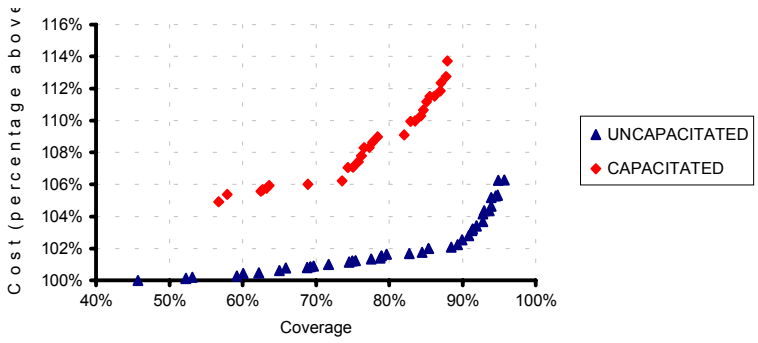
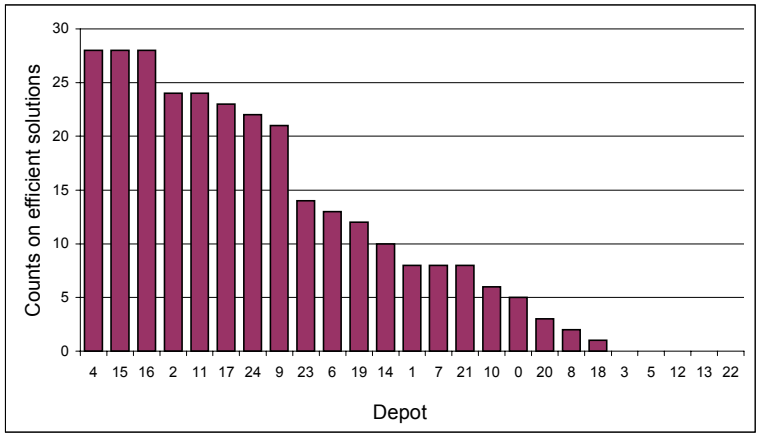


Figure 10. Relative importance based on frequency counts in the approximate efficient frontier for the BOCFLP

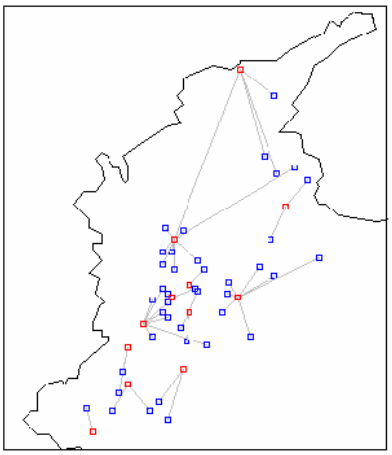


minimum cost, it is possible to increase coverage by 25.32%. On the other hand, in the right side of the frontier, it is quite costly to improve coverage. For instance, to increase coverage just 0.16%, the FNCC must incur a 0.94% cost increment (see solutions 27 and 28). The reason why it is so expensive to increase coverage in the right side of the frontier is that new depots must be open to cover small and distant purchasing centers.

To illustrate how capacity constraints impact the supplier network design, Figure 9 shows an overlay of the efficient frontiers for the BOUFLP and BOCFLP. The vertical axis is measured against the minimum cost configuration for the BOUFLP. As expected, the BOCFLP's frontier is dominated by the BOUFLP's—that is, enforcing capacity constraints implies higher costs and lower coverage.

To evaluate the relative importance of the depots in the Colombian coffee supplier network, we count how many times each depot appears in the approximate Pareto frontier of

Figure 11. Solution 25 for the BOCFLP



the BOCFLP. Figure 10 shows the result of this count. Several depots consistently appear in the efficient solutions, while some do not. Based on this analysis, depots 3, 5, 12, 13, and 22 are strong candidates for closing.

Finally, it is worth mentioning that the current supplier configuration is not part of the efficient frontier (see Figures 6 and 8). Moreover, as shown in Figure 11, solution 25 offers a cheaper solution with slightly better coverage.

CONCLUSION

We have presented the MO-JGA framework, a flexible object-oriented computational tool for rapid prototyping and implementation of evolutionary algorithms. We illustrated how multiobjective evolutionary algorithms, such as the NSGA-II, can be extended and implemented from MO-JGA.

From a practical side, we have shown how the use of an evolutionary algorithm is able to provide valuable insight into a complex biobjective location problem arising in the context of the Colombian coffee supplier network. We also illustrate how both the uncapacitated and capacitated versions of the location problem can be implemented from a common evolutionary core with minor modifications. This latter point is important because it highlights the robustness of the evolutionary solution strategy.

Being JGA the building block of MO-JGA, it is worth mentioning that as the JGA framework grows, MO-JGA will be able to solve more complex problems arising in the business setting. Last, but not least, MO-JGA class libraries, examples, and documentation are available at http://copa.uniandes.edu.co/software.

ACKNOWLEDGMENT

We thank Dash Optimization for providing us with licenses of Xpress-MP via the Academic Partner Program subscribed with Universidad de los Andes.

REFERENCES

Aikens, C. H. (1985). Facility location models for distribution planning. *European Journal of Operational Research, 22,* 263-279.

Alp. O., Erkut, E., & Drezner, Z. (2003). An efficient genetic algorithm for the p-median problem. *Annals of Operations Research, 122,* 21-42.

Antunes, A. P. (1999). Location analysis helps manage solid waste in central Portugal. *Interfaces, 29*(4), 32-43.

Bramel, J., & Simchi-Levi, D. (1997). *The logic of logistics*. New York: Springer.

Brandeau, M. L., & Chiu, S. S. (1989). An overview of representative problems in loca-

tion research. *Management Science, 35,* 645-674.

Brimberg, J., & ReVelle, C. (1998). A bi-objective plant location problem: Cost vs. demand served. *Location Science, 6*(1-4), 121-135.

CAIC (Comisión de Ajuste a la Institucionalidad Cafetera). (2002). *El Café capital social y estratégico.* Bogota: CAIC.

Chu, P. C., & Beasley, J. E. (1997). A genetic algorithm for the generalized assignment problem. *Computers & Operations Research, 24*(1), 17-23.

Coello, C. A. (2000). An updated survey of GA-based multiobjective optimization techniques. *ACM Computing Surveys, 32*(2), 109-143.

Coello, C. A. (2001). A short tutorial on evolutionary multiobjective optimization. *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization* (pp. 21-40). London: Springer-Verlag.

Coello, C. A., & Toscano, G. (2001). A micro-genetic algorithm for multiobjective optimization. *Proceedings of the 1st international Conference on Evolutionary Multi-Criterion Optimization* (pp. 126-140). London: Springer-Verlag.

Coello, C. A., Van Veldhuizen, D. A., & Lamont, G. A (2002). *Evolutionary algorithms for solving multi-objective problems.* New York: Kluwer Academic.

Coello, C. A. (2005). *EMOO Web page.* Retrieved August 13, 2005, from http://delta.cs.cinvestav.mx/~ccoello/EMOO/

Collette, Y., & Siarry, P. (2003). *Multiobjective optimization: Principles and case studies.* Berlin: Springer-Verlag.

Corne, D. W., Knowles, J. D., & Oates, M. J. (2000). The Pareto-envelope based selection algorithm for multiobjective optimization. *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature (PPSN VI)* (pp. 839-848). Berlin: Springer-Verlag.

Cortinhal, M. J., & Captivo, M. E. (2001). Genetic algorithms for the single source capacitated location problem: A computational study. *Proceedings of MIC'2001, the 4th Metaheuristics International Conference,* Porto, Portugal (pp. 355-359).

Current, J., Min, H., & Schilling, D. (1990). Multiobjective analysis of facility location decisions. *European Journal of Operational Research, 49,* 295-307.

Daskin, M. S. (1995). *Network and discrete location. Models, algorithms and applications.* New York: John Wiley & Sons.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms.* New York: John Wiley & Sons.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 181-197.

Doerner, K., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2004). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research, 131,* 79-99.

Drezner, Z., & Hamacher, H. W. (Eds.). (2001). *Facility location: Applications and theory.* Berlin: Springer-Verlag.

Ehrgott, M., & Gandibleux, X. (Eds.). (2002). *Multiple criteria optimization: State-of-the-*

art annotated bibliographic survey series. Boston: Kluwer Academic.

Ehrgott, M., & Gandibleux, X. (2004). Approximative solution methods for multiobjective combinatorial optimization. *TOP, 12*(1), 1-89.

Filho, V. J. M. F., & Galvão, R. D. (1998). A tabu search heuristic for the concentrator location problem. *Location Science, 6,* 189-209.

FNCC (Federación Nacional de Cafeteros de Colombia). (2001). *Participación de las cooperativas en el mercado cafetero durante el año civil 2001.* Bogotá: FNCC.

FNCC. (2005). *Información economía cafetera.* Retrieved August 17, 2005, from http://www.cafedecolombia.com/economiacafetera/cuadro8.html

Fonseca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 416-423), University of Illinois at Urbana, Champaign, USA. San Mateo, CA: Morgan Kauffman.

Gandibleux, X., Sevaux, M., Sörensen, K., & T'kindt, V. (Eds.). (2004). *Metaheuristics for multiobjective optimization.* Berlin: Springer-Verlag.

Gandibleux, X., Mezdaoui, N., & Fréville A. (1997). A tabu search procedure to solve multiobjective combinatorial optimization problems. *Proceedings of the 2nd International Conference on Multi-Objective Programming and Goal Programming* (pp. 291-300). New York: Springer-Verlag.

Golden, B. L., & Skiscim, C. C. (1986). Using simulated annealing to solve routing and location problems. *Naval Research Logistics Quarterly, 33,* 261-279.

Hale, T. S., & Moberg, C. R. (2003). Location science research: A review. *Annals of Operations Research, 123*(1-4), 21-35.

Hapke, M., Jaszkiewicz, A., & Zak, J. (2002). The design of the physical distribution system with the application of the multiple objective mathematical programming. Case study. In T. Traskalik, & J. Michnik (Eds.), *Multi-objective programming and goal programming. Recent developments* (pp. 297-309). Heidelberg: Physica-Verlag.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A Niched Pareto Genetic Algorithm for multiobjective optimization. *Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence* (Vol.1, pp. 82-87). Piscataway, NJ: IEEE Press.

Jaramillo, J. H., Badhury, J., & Batta, R. (2002). On the use of genetic algorithms to solve location problems. *Computers & Operations Research, 29,* 761-779.

Jaszkiewicz, A. (2002). Genetic local search for multiple objective combinatorial optimization. *European Journal of Operational Research, 137*(1), 50-71.

Jayaraman, V. (1999). A multi-objective logistics model for a capacitated service facility problem. *International Journal of Physical Distribution & Logistics, 29*(1), 65-81.

Jones, D. F., Mirrazavi, S. K., & Tamiz, M. (2002). Multi-objective meta-heuristics: An overview of the current state of the art. *European Journal of Operational Research, 137*(1), 1-9.

Klose, A., & Drexl A. (2005). Facility location models for distribution system design. *European Journal of Operational Research, 162*(1), 4-29.

Knowles, J. D., & Corne, D. W. (2000). M-PAES: A memetic algorithm for multiobjective optimization. *Proceedings of the Congress on Evolutionary Computation (CEC00)* (pp. 325-332). Piscataway, NJ: IEEE Press.

Knowles, J. D., & Corne, D. W. (2000). Approximating the non-dominated front using the Pareto archived evolution strategy. *Evolutionary Computation, 8*(2), 149-172.

Kratica, J., Tosic, D., Filipovic, V., & Ljubic I. (2001). Solving the simple plant location problem by genetic algorithm. *RAIRO Operations Research, 35,* 127-142.

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs.* New York: Springer-Verlag.

Michel, L., & Van Hentenryck, P. (2004). A simple tabu search for warehouse location. *European Journal of Operational Research, 157*(3), 576-591.

Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Multi-objective genetic algorithm and its applications to flowshop scheduling. *Computers & Industrial Engineering, 30*(4), 957-968.

Nozick, L. K., & Turnquist, M. A. (2001). Inventory, transportation, service quality and the location of distribution centers. *European Journal of Operational Research, 129*(2), 362-371.

Nozick, L. K. (2001). The fixed charge facility location problem with coverage restrictions. *Transportation Research Part E, 37,* 281-296.

Owen, S. H., & Daskin, M. S. (1998). Strategic facility location: A review. *European Journal of Operational Research, 111,* 423-447.

Pizzolato, N. D., Barros, A. G., Barcelos, F. B., & Canen, A. G. (2004). Localização de escolas públicas: Síntese de algumas linhas de experiências no Brasil. *Pesquisa Operacional, 24*(1), 111-131.

Resende, M. G. C. (1998). Computing approximate solutions of the maximum covering problem with GRASP. *Journal of Heuristics, 4,* 161-171.

ReVelle, C. S., & Laporte, G. (1996). The plant location problem: New models and research prospects. *Operations Research, 44,* 864-874.

ReVelle, C. S., & Eiselt, H. A. (2005). Location analysis: A synthesis and survey. *European Journal of Operational Research, 165*(1), 1-19.

ReVelle, C. S., Marks, D., & Liebman, J. C. (1970). An analysis of private and public sector location models. *Management Science, 16,* 692-707.

Rosing, K. E., & ReVelle, C. S. (1997). Heuristic concentration: Two-stage solution construction. European. *Journal of Operational Research, 97,* 75-86.

Schaffer, J. D. (1984). *Multiple objective optimization with vector evaluated genetic algorithms.* PhD thesis, Vanderbilt University, USA.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using non-dominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221-248.

Ulungu, E. L., Teghem, J., Fortemps, P. H., & Tuyttens, D. (1999). MOSA method: A tool for solving multi-objective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis, 8,* 221-236.

Verter, V., & Lapierre, S. D. (2002). Location of preventive health care facilities. *Annals of Operations Research, 110,* 123-132.

Villegas, J. G., Palacios, F., & Medaglia, A. L. (2006). Solution methods for the bi-objective (cost-coverage) unconstrained facility location problem with an illustrative example. *Annals of Operations Research*.

Zhou, G., Min, H., & Gen, M. (2003). A genetic algorithm to the bi-criteria allocation of customers to warehouses. *International Journal of Production Economics, 86,* 35-45.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257-271.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In (K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, & T. Fogarty (Eds.), *Evolutionary methods for design, optimization, and control* (pp. 95-100). Barcelona: CIMNE.

KEY TERMS

Efficient Frontier: Let X_E be the set of efficient solutions. The Efficient Frontier (or Pareto Optimal Front) is the set of images (in the criteria space) of the solutions in X_E. Let PF_E be the efficient frontier defined by $PF_E := \left\{ \left(f_1(x),\ldots,f_k(x),\ldots,f_p(x)\right) \in R^p \middle| x \in X_E \right\}$.

Evolutionary (Genetic) Algorithm: An evolutionary algorithm (EA) is an stochastic search procedure inspired by the evolution process in nature. In this process, a population of individuals evolve, and the fitter ones have a better chance of reproduction and survival. The reproduction mechanisms favor the characteristics of the stronger parents and hopefully produce better children, guaranteeing the presence of those characteristics in future generations.

Efficient Solution (Non-Dominated Solution): A feasible solution $x \in X$ is called efficient if there does not exist $x' \in X$ such that $f_k(x') \le f_k(x)$ for all $k = 1,\ldots,p$ and $f_k(x') < f_k(x)$ for some k.

Facility Location Problem (or Facility Siting Problem): Facility location problems (FLPs) deal with choosing from among some candidate facilities a subset from which an organization will serve its customers. The FLP is also concerned with determining how these customers will be served by the open facilities.

JGA: Acronym for Java Genetic Algorithm; a flexible and extensible computational object-oriented framework for rapid development of evolutionary algorithms for solving complex optimization problems.

Multi-Objective Evolutionary Algorithm (MOEA): The extension of evolutionary algorithms to solve the multi-objective optimization problem.

Multi-Objective Optimization Problem (MOP): The MOP is defined as

$$\min_{x \in X} \left(f_1(x),\ldots,f_k(x),\ldots,f_p(x)\right)$$

where $X \subset R^n$ is the feasible set and $f : R^n \to R^p$ is a vector valued objective function.

Chapter XLIII
Worker Performance Modeling in Manufacturing Systems Simulation

Peer-Olaf Siebers
University of Nottingham, UK

ABSTRACT

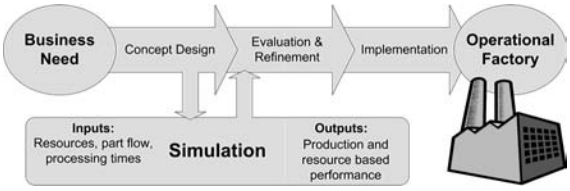
Discrete event simulation is generally recognized as a valuable aid to the strategic and tactical decision making that is required in the evaluation stage of the manufacturing systems design and redesign processes. It is common practice to represent workers within these simulation models as simple resources, often using deterministic performance values derived from time studies. This form of representing the factory worker ignores the potentially large effect that human performance variation can have on system performance, and it particularly affects the predictive capability of simulation models with a high proportion of manual tasks. The intentions of the chapter are twofold: firstly, to raise awareness of the importance of considering human performance variation in such simulation models; and secondly, to present some conceptual ideas for developing a worker agent for representing worker performance in manufacturing systems simulation models.

INTRODUCTION

Manufacturing systems are most often highly complex constructs and their behavior is of a dynamic and stochastic nature. They consist of extensive interactions between people, information, materials, and machines. Systems like assembly lines may look quite simple because their tasks are mainly done in a sequential order. In reality, these systems are quite complex constructs due to breakdowns of various types and natural variation in processing times, which makes them non-deterministic. The breakdowns can be machine failures, but in systems like manual assembly lines where humans play a key role, they can also be unusually long task completion times or the unavailability of workers.

When it comes to the design or redesign of manufacturing systems, it is common to use a methodological approach. Discrete event simulation (DES) is generally recognized as a valu-

Figure 1. Steps in manufacturing systems design



able aid to the strategic and tactical decision making that is required in the evaluation stage of the design process. Figure 1 depicts the way in which DES integrates into the manufacturing system design process. A major advantage of simulation models, compared to the analytical ones that are also in use, is their ability to model random events based on standard and non-standard distributions and to predict the complex interactions between these events. This allows the system designer to obtain a system-wide view of the effect of local changes to the performance of the overall system and enables him or her to predict system performance, to compare alternative system designs, and to determine the effect of alternative policies on system performance.

Among other things, DES models are used to determine the amount of machines, buffers, and operators that are needed to produce a certain target output. Companies that have groups which specialize in studying multi-million-dollar systems using DES include Honda, Ford, General Motors, Harley-Davidson, and Renault (Baudin, 2002). The simulation experts within these groups have a high degree of responsibility to ensure the accuracy of the results. Inaccuracy can prove very costly, as it may lead to poor system performance and failure to meet the production demand.

Due to the complexity of the real world, a system model can only be a restricted copy of a real system. Therefore, abstraction and simplification have to be used in order to cope with this complexity. Abstraction comprises or concentrates in itself the essential qualities or behaviors of a thing, but not necessarily in the same form or detail as in the original, while simplification entails stripping away unimportant details and assuming simpler relationships (Shannon, 1975).

It is commonly observed that a gap exists between the performance predictions of a manufacturing system simulation model and the performance of the real system. As a consequence of abstraction and simplification, system models tend to model the real world too optimistically compared to real systems. Another common observation is that performance predictions of systems involving a high proportion of manual tasks are notably less accurate than those of highly automated systems. This is attributed to the way in which the human element is represented within the system simulation model. It is common practice within DES models to represent workers as simple resources, often using deterministic performance values derived from time and motion studies. This is an extreme simplification as the work measurement literature indicates clearly that workers' task performance varies. This variation occurs between different workers carrying out the same task, and moreover for the same worker when repeating a task (e.g., Dudley, 1968). It has also been shown that workers' task performance varies as a consequence of its dependence on past events and the current state of the system. The current approach of representing workers within DES models ignores the potentially large effect that human performance variation (HPV) can have on the system performance of the labor-intensive manufacturing system.

This chapter has been written with two objectives in mind: firstly, to raise awareness of the importance of considering HPV in human-oriented DES models; and secondly, to offer some conceptual ideas for developing a more

sophisticated representation of direct workers (people dedicated to predominately manual routines) in DES models. The chapter is therefore split into two main parts. The first part reports on research that has been conducted to address the first objective. It describes a field experiment carried out to quantify direct worker performance variations in automotive manual assembly lines and test the sensitivity of simulation models towards these variations. It then discusses the results and the limitations of the approach that has been chosen to represent these variations.

The second part consists of a collection of ideas which is intended to contribute towards achieving the second objective in the near future. A literature review is provided to identify design opportunities that allow a more advanced representation of worker variability and behavior in manufacturing systems simulation models. These opportunities are derived from the systems engineering and the social science literature, where human performance and behavior modeling is already used for many different purposes in many different ways. An agent-based approach is identified as the most suitable way forward, and concepts for a worker agent and its integration into manufacturing system simulation models are developed. The chapter concludes by discussing the problems of implementing the proposed concepts and identifying possibilities for future work.

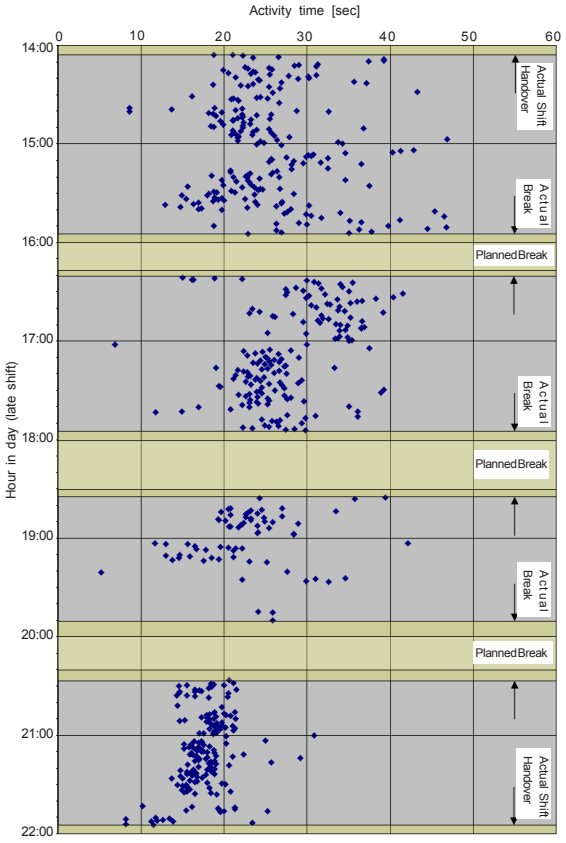
THE NEED FOR NON-DETERMINISTIC MODELS OF WORKER PERFORMANCE

A common means of representing the performance of direct workers within manufacturing system simulation models is to use so-called standard times. These are the times required by an average skilled worker, working at a normal pace, to perform a specific task using a prescribed method, allowing time for personal needs, fatigue, and delay. They are mean values derived either from direct work measurements through means of time studies or from indirect work measurements through means of synthetic timing. Using mean values and ignoring the natural variation in these task completion times represents a significant simplification that can have a high impact on the simulation model runtime behavior and consequently on the accuracy of the performance predictions of the simulation model. On the other hand it might be that due to the long runtime of the simulation models (usually several months are simulated), the variation equilibrates and therefore the simplification is legitimate. This section describes some research that has been carried out to investigate this issue by means of a sensitivity analysis.

Sensitivity Analysis Using Empirical Frequency Distributions

Firstly, a long-term data collection exercise was conducted to quantify the performance variation of direct workers in a typical automotive manual engine assembly flow line (Siebers, 2004). The line that has been observed is divided into 10 different work zones. In each work zone the work is performed by a team of six to twelve workers. The workers rotate by the hour in their subsequent zone, which means that each individual works on all the tasks in his or her section. Data about the activity times (basically the time that an individual is working on a particular engine) were collected at 10 different workstations over a period of three months using an automated data collection method. The collected data were also used to analyze the break-taking behavior of the workers. Figure 2 shows a time series plot of the collected data points for one particular work station over an eight-hour shift. Due to the rotation mentioned earlier, each hour slice rep-

Figure 2. Activity time variations for a manual workstation during an eight-hour shift



resents the activity times of a different worker. The diagram shows clearly that there are differences in activity times when a worker repeats a task and also between different workers. Furthermore it can be seen that the actual breaks are significantly longer than the planned breaks.

The collected data were then transformed into empirical frequency distributions representing activity time variations and break-taking behavior. This format supports the integration of the collected data into DES models. By looking at the resulting frequency distributions, it was found that the scattering of activity times is dependent on the nature of the task and that there is no generic pattern for manual tasks. For the frequency distributions representing break-taking behavior, it was shown that the scattering of the break start and break duration does not depend on the break length. The average disruption due to leaving the workstation early and coming back late was the same for all breaks.

In order to enable a sensitivity analysis, the designed empirical frequency distributions were inserted into simulation models of existing engine assembly lines (each with approximately 100 workstations) to represent the performance variation of the direct workers at the individual workstations. Through designed experiments, the effect that this form of HPV modeling has on the behavior of the simulation models was investigated.

The results from these experiments showed that the representation of HPV can have a significant impact on the behavior of the system simulation models. The impact depends basically on the type of variation to be represented as well as on the system to be modeled. While the impact of activity time variation is system dependent (which is the same in the real world) and depends among other things on the buffer sizes (bigger impact on systems with smaller buffer sizes), the impact of break-taking behavior is always present.

As a result of these investigations, system designers employing simulation as a decision support tool should now be aware of the consequences that ignoring variation in worker performance may have on the validity of their system analysis.

Limitations of the Empirical Frequency Distribution Approach

Despite the fact that using empirical frequency distributions is a step forward compared to current standards in the simulation community,

it has two major drawbacks. The first one is that the distributions are context specific, that is to say, form and magnitude of HPV is among others a function of task, workforce composition, environment, and location. No generic distribution shape could be identified to describe HPV. Therefore, using the distributions in locations other than their origin will reduce their validity. The second drawback, which is even more severe, is that the distributions on their own are not capable of expressing interdependencies between events and do not consider the state of the system as values are chosen at random.

Conversely, research has shown that the activity time of workers is dependent on past events and the current state of the system. Schultz, Juran, Boudreau, McClain, and Thomas (1998), for example, have presented evidence that worker behavior differs, depending on the size and status of the buffers surrounding them and with whom the worker is teamed. Doerr, Mitchell, Klastorin, and Brown (1996) found that workers speed up when they are the cause of idle time, even in the absence of management pressure. De Souza and Zhao (1997) argue that a complete and effective representation of dynamic behavior would require a composite representation of knowledge in various forms.

It has been proposed that the use of a combination of rules and distributions for representing the dynamic behavior of workers would be a step forward. The rules would allow a guided choice of stochastic values based on the system status, profile and state of the individual worker, and the work group he or she is working in. This would preserve the stochastic nature of the individual component while considering interdependencies with other components of the system.

Ideally, workers would be modeled as autonomous and proactive entities, constantly monitoring their environment, reasoning, and reacting to internal and external stimuli. This would account for the fact that in human-oriented systems, compared to non-human ones, an additional level of feedback is occurring. People notice what is going on around them and adjust their behavior accordingly, a phenomenon also known as second-order emergence (Gilbert & Troitzsch, 1999).

A LITERATURE REVIEW ON MODELS OF HUMAN PERFORMANCE

Over the past few decades, tools and techniques for modeling and predicting human performance in complex systems have evolved and matured. Pew and Mavor (1998) state that models and techniques are emerging within the systems engineering and social science domains which clearly indicate that some valid modeling of operator performance is possible. Table 1 provides a classification of human performance modeling approaches. Details about the elements can be found in Elkind, Card, Hochberg, and Huey (1990) for the systems engineering approaches, and in Gilbert and Troitzsch (1999) for the social science approaches.

Table 1. Human performance modeling in systems engineering and social science

Systems Engineering	References
Bio mechanical models	e.g. Kroemer et al. (1988)
Information sensing and processing models	e.g. Harris et al. (1986)
Knowledge based/cognitive approach	e.g. Sasou et al. (1996)
Optimal control theory models	e.g. McCoy & Levary (2000)
Anthropometric models	e.g. Kroemer et al. (1988)
Task network models	e.g. Laughery (1998)
Workload prediction models	e.g. Hendy & Farrell (1997)
Situational awareness models	e.g. Corker (1999)
Human reliability models	e.g. Sebok et al. (1997)
Micro models	e.g. Spencer (1987)
Integrated models	e.g. Bunting & Belyavin (1999)
Social Science	**References**
Artificial society modeling (some of the main forms)	e.g. Epstein & Axtell (1996)
- Microsimulation	e.g. Orcutt (1986)
- Cellular automata	e.g. Gaylord & D'Andria (1998)
- Production systems	e.g. Ye & Carley (1995)
- Multi-agent modeling	e.g. Weiss (1999)
- Learning and adaptive models	e.g. Axelrod (1987)
Descriptive human performance modeling	e.g. Schultz et al. (1999)
Emergency simulation	e.g. Olenick & Carpenter (2003)

Simulation is a new way of examining social and economic processes by studying the emergence of complex behavior from relative simple activities (Gilbert & Troitzsch, 1999). The study of complex systems is a new field of science related to complexity theory. It examines how parts of a system lead to the collective behaviors of the system, and how the system interacts with its environment. It cuts across all traditional disciplines of science, as well as engineering, management, and medicine, and is about understanding indirect effects that are not obviously related to their causes (Bar-Yam, 1997). Complex adaptive systems (CASs) are a specific category of complex systems that change their behavior in response to their environment (Bar-Yam, 1997). They are denoted by the following three characteristics: evolution, aggregate behavior, and anticipation (Holland, 1992; Trisoglio, 1995). Here, evolution refers to the adaptation of systems to changing environments, aggregate behavior refers to the emergence of overall system behavior from the behavior of its components, and anticipation refers to the expectations the intelligent agents involved have regarding future outcomes. Since CASs adapt to their environment, the effect of environmental change cannot be understood by considering its direct impact alone. Therefore, the indirect effects also have to be considered due to the adaptive response.

Organizations, which are basically groups of people that are working together in order to attain common goals, can be characterized as CASs composed of intelligent, task-oriented, boundedly rational, socially situated agents. These agents are faced with an environment that has the potential for change (Carley & Prietula, 1994). Computational organization theory (COT) is concerned with building new concepts, theories, and knowledge about organizing and organization in the abstract, to develop tools and procedures for the validation and analysis of computational organizational models, and to reflect these computational abstractions back to actual organizational practice through both tools and knowledge (Carley & Gasser, 1999). One of the most commonly used techniques to model CASs is multi-agent-based modeling (Skvoretz, 2003) where the organization is composed of a number of intelligent agents. Unlike traditional multi-agent models, COT models draw on and have implemented into them empirical knowledge from organization science about how the human organizations operate and about basic principles for organizing (Carley & Gasser, 1999).

Another way of describing a human-oriented manufacturing system is to see it as an organization in which groups of people work together to attain common goals. These groups include for example machine operators or assembly line workers. It seems that COT can also provide a promising paradigm for modeling the behavior of factory workers. The use of multi-agent-based models is also supported by Gazeendam (1993), who states that multi-agent-based models show promise as models of organizations, since they are based on the idea that most of the work in human organizations is done based on intelligence, communication, cooperation, and massive parallel processing.

PROPOSAL FOR AN AGENT-BASED APPROACH TO WORKER PERFORMANCE MODELLING

As already discussed above, workers within manufacturing system simulation models would ideally be modeled as autonomous and proactive entities, constantly monitoring their environment, reasoning, and reacting to internal and external stimuli. Multi-agent-based modeling has been identified as a promising way forward. In the following section we develop some

Figure 3. Tool development after step 1; generic theory-building approach

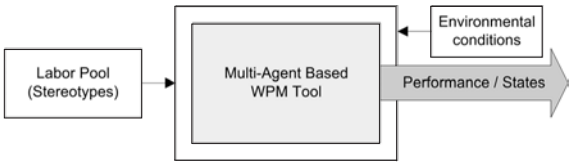
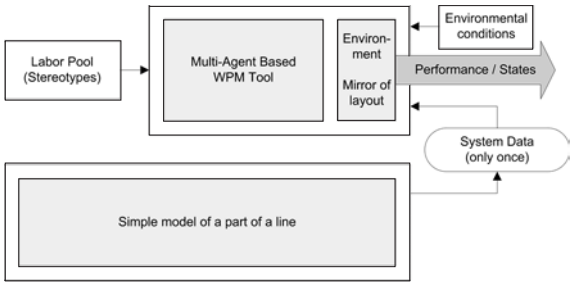


Figure 4. Tool development after step 2; task-based approach



ideas of what such a system could look like. The motivation is not to present a working multi-agent system (MAS), but to present some considerations and thoughts to support the development of such a system.

A top-down approach has been taken. It begins by describing some conceptual ideas for a multi-agent-based worker performance modeling (WPM) tool and its integration into the DES modeling environment of the manufacturing system. This is followed by a brief review of existing agent architectures. Different models and frameworks from occupational psychology and organizational behavior research are then examined that could help to decide about the factors and state descriptors to be considered inside the agent architecture. Finally, a worker agent framework is presented that in the author's view could be used to develop some worker agents.

Ideas for a Multi-Agent-Based Worker Performance Modeling Tool

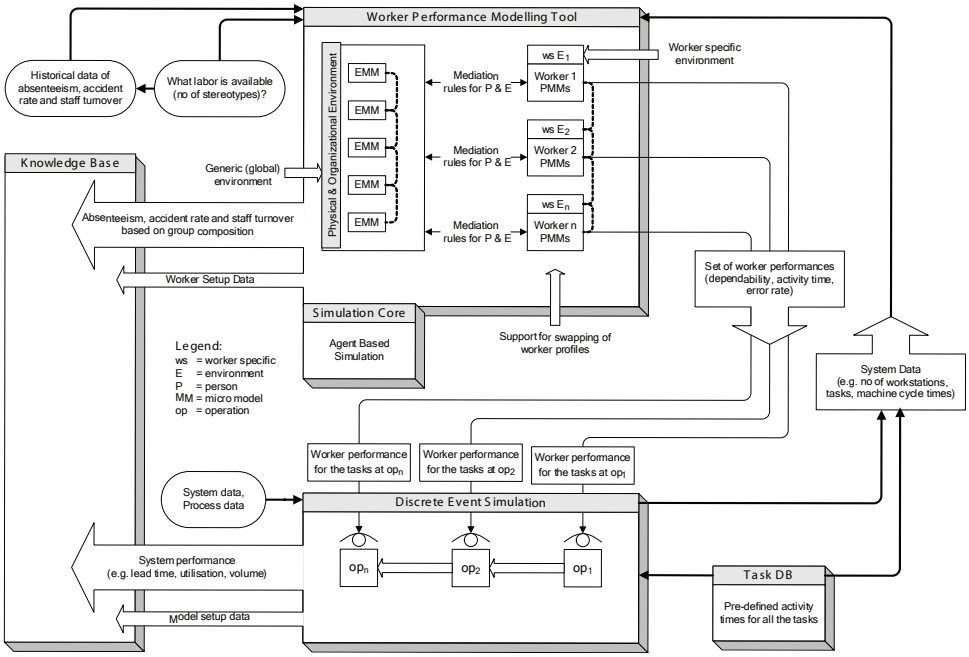
The task of developing a multi-agent-based modeling tool that is capable of representing workers within DES models is very challenging. Therefore a three-step development process is proposed that gradually increases in complexity and still produces useful tools at the end of each development step.

Figure 3 represents the deliverable at the end of the first step which is a generic theory-building tool. No manufacturing system process data is required. The tool can be seen as an artificial white room—that is, a simulation of a laboratory as it is used for data gathering under controlled conditions. The output of the tool would support the development of lookup tables, functions, and distributions that describe system-independent variations in state and performance of a workforce at the individual and group level. These outputs can be used to represent worker and workgroup performance in new or existing manufacturing simulation models.

Figure 4 represents the deliverable at the end of the second step which consists of a task-based approach. In comparison to the first tool, the output would be system related. The tool would require basic process and layout setup data, as well as task definitions for the individual agents. Therefore the output of the tool would support the development of lookup tables, functions, and distributions tailored for a specific manufacturing system.

Figure 5 represents the deliverable at the end of the third and final step which consists of a situation-based or integrated approach. Here the MAS takes over the representation of the workers within the manufacturing system simulation model. The state of the agents is updated whenever the state of the system changes and

Figure 5. Tool development after step 3; situation-based approach



feedback of agent state changes is given to the manufacturing system simulation model. Once the simulation is finished, the data from the multi-agent-based tool can be stored in a knowledge base to allow comparison of the behavior of agents in different scenarios later on. This might support the development of new behavioral rules to implement in the tool.

As this is the final tool, the elements and its manner of operation are described in more detail. The WPM tool consists of worker agents, local environment agents, and a global environment agent. Each worker agent represents a worker entity from the DES model and is linked to a local environment agent that represents the specific environmental conditions at a particular workplace. The WPM tool has its own simulation core which makes it independent from the DES model. The global environment consists of a collection of micro models representing physical and operational environmental factors. As an input, the WPM tool would require static information about the profile of the workforce and historical data about absenteeism, accident rate, and staff turnover. Additional dynamic information from the DES model about the task to conduct and the state of the system is also needed. As an output the WPM tool delivers the values for a set of direct worker performance indicators consisting of activity time, error rate, and dependability for each individual worker. Furthermore, at the end of each simulation run, probability values for indirect worker performance indicators that

reflect worker well-being consisting of absenteeism, accident rate, and staff turnover are estimated and stored in a knowledge base together with the setup data of the workforce.

The DES model would typically be one developed in a visual interactive modeling system that allows some form of programming. Connected to this DES model would be a task database which includes all the predefined standard times for the tasks to be conducted by the workers. At each event that significantly changes the state of the system, and that might therefore have an impact on worker performance, a set of system state data is sent to the WPM tool. This data is sent in combination with a request for a set of worker performance data that reflect worker reactions to the new situation. At the end of each simulation run, the system performance data (e.g., productivity) is stored in a knowledge base together with the setup data of the DES model. Once the knowledge base has been populated with a reasonable quantity of cases, it could be used to try and identify emergent patterns. It might be possible to derive rules that describe how a certain system setup impacts on different workforces or how a certain workforce reacts to different system setups.

Choosing a Worker Agent Architecture

Before considering which factors and state descriptors are needed for a worker agent, a decision must be made as to what underlying agent architecture to use. This is because different architectures require different categories of factors and state descriptors. There are many ways to design the inner structure of an agent, and many different agent architectures have been developed over the years. Wooldridge (1999) classifies architectures for intelligent agents into four groups, as represented in Table 2. Furthermore, the table contains some examples of concrete agent architectures for each class with reference to some of their key contributors.

For the purpose of modeling human behavior, Schmidt (2000) proposes PECS (physique, emotion, cognition, social status) as a new reference model, which aims to replace the BDI (belief, desire, intention) architecture. He argues that the BDI structure is not appropriate for modeling real social systems as it conceives human beings as rational decision makers, while in reality their decisions are controlled by reactive and deliberate behavior. As these are attributes that we want to represent, for the moment the PECS reference model will be considered to be the most suitable one for our purposes.

Worker Agent State Descriptors

In order to design a worker agent, it is necessary to investigate what factors influence direct worker performance. These factors could be used as state descriptors in the form of state constants, state variables, and state transition functions if they are definable, tangible, quantifiable, and can be evaluated. Some models and frameworks that might be of help in giving indications for such factors have been developed in occupational psychology and organizational behavior research, and are applied in job design and human resources. A selection of these are now described and discussed with regards to their usefulness in terms of offering ideas for worker agent state descriptors.

A classical model of person-environment interaction is provided by Lewin (1935), who states that behavior (B) is a function of interactions between the person (P) and the environment (E) at any given time or situation: B=f(P,E). Lewin's theory stresses the importance of understanding behavior within the total situation, and his model accounts for the natural variability of behaviors between different situations. It is now commonly accepted that person and

Table 2. Classification of agent architectures (after Wooldridge, 1999)

Class	Examples of concrete architectures
Logic based agents	Situated automata (Kaelbing, 1986)
Reactive agents	Subsumption architecture (Brooks, 1986)
Belief-Desire-Intention agents	BDI architecture (Bratman et al 1988)
Layered architectures	Touring machines (Ferguson, 1992)

system factors influence work behaviors (Williams & Fletcher, 2002). Unfortunately, the model is of a very general nature, and the challenge is to find the relevant factors for B, P, and E, and their mediation rules.

Support for defining the relevant factors for B, P, and E may be found in modern job design models. Das (1999) has attempted to integrate an extensive list of modern industrial job design factors into a single comprehensive model. Unfortunately, there are some substantial weaknesses in this model: it does not elaborate on how factors interact within the workspace to influence the specific outcomes, and it does not weigh any of the factors pointed out with a degree of importance to define their impact. Das (1999) himself indicated that it will be extremely difficult to collect all the empirical evidence to determine the effects and interactions of the various work design factors.

A model proposed by Parker, Wall, and Cordery (2001) provides categories rather than a universal list of variables in order that specific factors can be identified according to a context. One of the advances of the model is that it considers basic pre-design conditions that may influence the effect of job design on performance outcome. The model that also accounts for the stages in which factors come into play includes a range of mediating mechanisms, antecedent factors, and contingencies that might affect the impact of work characteristics.

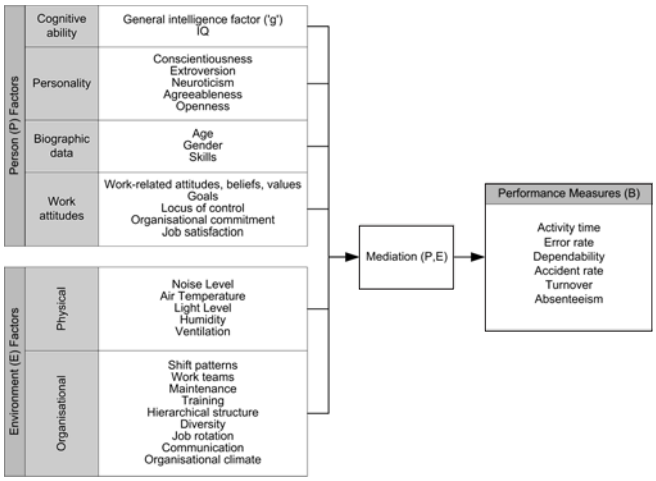
Silverman, Might, Dubois, Shin, and Johns (2001) state that performance moderator functions could be used to increase the realism of human behavior models, and express at the same time that the modeling and simulator communities are finding it difficult to extract performance moderator functions from the behavioral literature. He offers a list of moderators that reflect significant dimensions of individual and group differences, as well as external stressors on individuals and/or groups.

A model presented by Furnham (1992) displays some of the main factors of individual differences that influence occupational behavior and how they relate to one another. He explains that the basic assumption of this model is that any act or behavior pattern of a specific individual can be accurately predicted from the linear addition of scores on various factors like personality, abilities, and temporary states, together with some measures of situational or environmental conditions. These factors are all weighted in accordance with their importance for the specific criterion behavior that is being predicted. This makes a number of assumptions: all relevant variables are specified, correct weightings can be obtained, no situational modifiers exist, and linearity is a given.

The problem with all the models presented so far is that they are either of a very general nature and therefore do not support the decision about relevant factors and variables, or their variables are indefinable and intangible constructs that would be difficult to quantify and evaluate in practice. Moreover, all of the models neglect to fully consider aspects of the physical environment which can be an important consideration within a factory environment.

A theoretical framework has been developed by Hadfield, Fletcher, Mason, Baines, and Ladbrook (2002), who seem to offer a solution to the problem. Figure 6 shows the framework that is based on the Lewin model mentioned earlier and has been specifically tailored to represent factory workers and their impact on system performance. The framework provides a unified approach, and identifies the main

Figure 6. Theoretical framework for worker performance modeling (Hadfield et al., 2002)



factors and performance measures for worker behavior through an extensive review and synthesis of the relevant literature using a criterion-based evaluation concerning: general relevance, contextual relevance, robustness of evidence, and measurability. However, the problem regarding the usability of the theoretical framework for the development of the worker agents is that it lacks a definition of state transition functions. These are key components of an agent framework. Therefore, with respect to its inability to consider time-related changes of the state of the worker throughout the day, it will be a useful help but cannot be used directly in order to define the inner structure of a worker agent.

Conceptual Design of a Worker Agent Framework

Finally, a worker agent framework is presented in Figure 7. It is based on the PECS reference model structure (Schmidt, 2000) and on the theoretical framework for WPM (Hadfield et al., 2002) introduced earlier in the chapter.

The theoretical framework has been enhanced for this purpose by state transition functions that have been derived from discussions with other researchers. In addition to the state variables and state transition functions of the original PECS reference model, the proposed worker agent framework includes state constants which describe factors such as biographic data and personality. Here, changes over time are assumed to be irrelevant, as simulations are usually only run for periods in which these factors do not change drastically. This is a significant development over the original reference model where such constants are not considered.

Problems with Implementing the Agent-Based Approach

Currently, the agent paradigm is not used in manufacturing system simulation for the purpose of modeling human behavior. In addition, multi-agent-based simulation is not yet accepted as a mainstream simulation technique in the manufacturing industry. Many problems and much resistance arise if one wants to adopt the paradigm into the context of manufacturing system simulation.

The main issue that has been identified is the complexity of the task, as worker agents would

not just be used for testing a specific limited hypothesis, which is often the motivation when agents are used in social sciences. Rather they would be used to analyze a variety of manufacturing systems, and therefore they must be designed to be valid in a much broader sense which inflates their complexity.

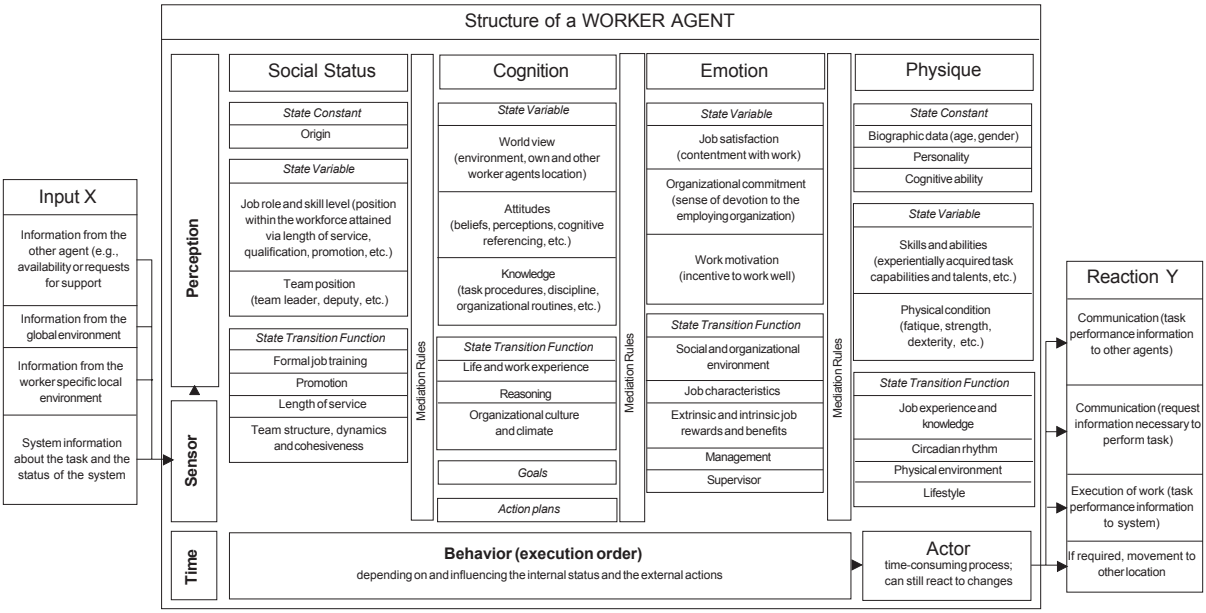
Another major problem still remains concerning the collection of sufficient data to populate the framework. Data that links an individual worker to his or her performance is required, but the old-fashioned industrial engineering approach of using time studies to set pay rates or extract more effort from workers has "poisoned the well for this activity" (Baudin, 2002). Therefore, it will be very difficult to collect the required empirical data in order to populate the worker agents. Another source of information would be social and behavioral science literature. Here, the difficulty is that mainly specific problems including only a very limited amount of factors are investigated, and the required mediation rules that would link all the factors within the framework are not known.

Finally, there are some technical issues. The agent concept is very appropriate for time-driven simulation, but less appropriate for event-driven simulation, which is commonly used in manufacturing systems design. DES models do not support the agent concept of proactiveness, since they are designed as reactive models. However, this is an important aspect when modeling human performance, as humans are able to take initiatives and act without external stimuli.

CONCLUSION

The focus of nature-inspired computing is most often related to problem solving and involves the study of natural phenomena, processes for the development of computational systems, and algorithms capable of solving complex problems. A second, less recognized objective, which has been the focus of this chapter, involves the modeling of natural phenomena and their simulation in computers. The goal in this direction is

Figure 7. Concept of a worker agent



to devise theoretical models that can be implemented in computers and are faithful enough to the natural mechanisms investigated so as to reproduce qualitatively and/or quantitatively some of their functionality.

The first part of the chapter has demonstrated the importance of incorporating HPV models into human-oriented manufacturing system simulation models, thereby enhancing the capabilities for simulation experts to represent the behavior and predict the performance of these systems more accurately. The second part of the chapter has focused on the requirements for a multi-agent-based approach to WPM. After carefully considering the possible development stages of a WPM tool, the choice of a suitable architecture for the worker agents, and the choice of state descriptors for direct workers, a new worker agent framework has been proposed. Finally, problems expected to occur upon implementation of the proposed MAS have been discussed. Two major roadblocks have been identified in the development of the proposed solution: first, the complexity of the task; and second, data collection problems.

In order to tackle the complexity issue, we suggest taking a step back and keeping the KISS (Keep It Simple, Stupid) principle in mind when implementing the proposed worker agent. The required approach is to design a very simplistic agent at first that only considers a few of the relevant state descriptors, and once these are under control, gradually enhance the complexity of the agent. Advances in the development of visual interactive multi-agent simulation platforms now make it a lot easier to design the agents step by step and at the same time allow experimentation with them. Software packages such as AnyLogic (XJ Technologies, 2005) allow the implementation of agents in a very comfortable way, as long as the required state descriptors and mediation rules are known. Notably, this particular package is a multi-paradigm simulation solution which allows the execution of hybrid models consisting of continuous and discrete elements, and so supports the integration of agents into DES models.

With regards to the data collection problems, here is a thought to stimulate discussion. How about an alternative approach to Figure 7 using an artificial neural network (ANN) internally to relate the dependent (performance measures) and independent (person and environment factors) variables? The assumption is that such an approach could somehow help to overcome the problem of defining all the mediation rules required to mediate between different factors. This assumption is based on an explanation of deployment areas of ANN by Francis (2001), who states that "such a data mining tool can fit data where the relationship between independent and dependent variables is nonlinear and the specific form of the nonlinear relationship is unknown." This would be the case for most of the factors listed in the theoretical framework presented in Figure 6. An ANN is also effective in dealing with two additional data challenges: correlated data and interactions (Francis, 2001). Externally this agent-like framework would have the same input and output variables as the worker agent. An example of such an approach applied in the field of industry dynamics is presented by Yildizoglu (2001), who uses ANNs within his firm agents to model the expectations conditioning the R&D decisions of firms.

Finally, it must be recognized that the limitations in representing HPV do not only affect manufacturing systems simulation, but in fact any system simulation where people play a key role. Simulation models in management science and operations research often represent complex systems that involve people, as for example in call centers and hospitals. An advance in modeling these people in terms of their behavior is expected to improve the value of simulation as a decision support tool for these application areas as well.

ACKNOWLEDGMENT

This research was conducted at Cranfield University, sponsored by the Ford Motor Company, and supported by EPSRC through the Innovative Manufacturing Research Center at Cranfield. The author would like to gratefully acknowledge the support and contribution of Prof. Tim Baines, Dr. Val Vitanov, Dr. Linda Hadfield, Dr. Sarah Fletcher, Dr. Steve Mason, Dr. Paul Mason, Mr. John Ladbrook, Mr. Daniel Bradfield, and Mr. Brent Zenobia.

REFERENCES

Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 32-41). London: Pitman.

Bar-Yam, Y. (1997). *Dynamics of complex systems (studies in nonlinearity).* Cambridge, MA: Perseus.

Baudin, M. (2002). *Lean assembly—The nuts and bolts of making assembly operations flow.* New York: Productivity Press.

Bratmann, M. E., Israel, D. J., & Pollack, M. E. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence, 4,* 349-355.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation, 2*(1), 14-23.

Bunting, A. J., & Belyavin, A. J. (1999, June 21-23). Modeling human performance in semi-automated systems. In *People in control: Humans interfaces in control rooms, cockpits and command centers.* IEE, University of Bath, UK.

Carley, K. M., & Gasser, L. (1999). Computational organization theory. In G. Weiss (Ed.), *Multi-agent systems: A modern approach to distributed artificial intelligence.* Cambridge, MA: MIT Press.

Carley, K. M., & Prietula M. J. (1994). ACTS theory: Extending the model of bounded rationality. In K. M. Carley & M. J. Prietula (Eds.), *Computational organization theory* (pp. 55-88). Hillsdale, NJ: Lawrence Erlbaum.

Corker, K. M. (1999, December 5-8). Human performance simulation in the analysis of advanced air traffic management. In P. A. Farrington, H. B. Nemhard, D. T. Sturrock, & G. W. Evans (Eds.), *Proceedings of the 1999 Winter Simulation Conference,* Phoenix, AZ (pp. 821-828). New York: ACM Press.

Das, B. (1999). Development of a comprehensive industrial work design model. *Human Factors and Ergonomics in Manufacturing, 9*(4), 393-411.

De Souza, R., & Zhao, Z. Y. (1997). A case for intelligent representation of dynamic resources in simulation. *International Journal of Production Research, 35*(12), 3289-3302.

Doerr, K. H., Mitchell, T. R., Klastorin, T. D., & Brown, K. A. (1996). Impact of material flow policies and goals on job outcomes. *Journal of Applied Psychology, 81,* 142-152.

Dudley, N. A. (1968). *Work measurement: Some research studies.* London: MacMillan.

Elkind, J. I., Card, S. K., Hochberg, J., & Huey, B. (1990). *Human performance models for computer-aided engineering.* Boston: Academic Press.

Epstein, J. M., & Axtell, R. (1996). *Growing artificial societies: Social science from the bottom up.* Cambridge, MA: MIT Press.

Ferguson, I. A. (1992, August 5-8). Towards an architecture for adaptive, rational, mobile agents. In E. Werner & Y. Demazeau (Eds.), *Decentralized AI 3—Proceedings of the 3rd European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW-91)*, Kaiserslautern, Germany (pp. 249-262). Amsterdam: Elsevier.

Francis, L. (2001). The basics of neural networks demystified. *Contingencies,* (11/12), 56-61.

Furnham, A. (1992). *Personality at work: The role of individual differences in the workplace.* London: Routledge.

Gaylord, R. J., & D'Andria, L. (1998). *Simulating society—A mathematical toolkit for modeling socio-economic behavior.* New York: Springer-Verlag.

Gazendam, H. W. M. (1993, July 4-9). Theories about architectures and performance of multiagent systems. *Proceedings of the 3rd European Congress of Psychology,* Tampere, Finland.

Gilbert, N., & Troitzsch, K. G. (1999). *Simulation for the social scientist.* Buckingham: Open University Press.

Harris, R. M., Glenn, F. A., Iavecchia, H. P., & Zalkad, A. (1986). Human operator simulator. In W. Karwowski (Ed.), *Trends in ergonomics/human factors III* (pp. 31-39). New York: North-Holland.

Hadfield, L., Fletcher, S., Mason, J. P., Baines, T., & Ladbrook, J. (2002, April 3-5). A theoretical framework for human performance modeling. *Proceedings of the 2nd International Conference on Systems Thinking in Management (ICSTM02)*, Salford, UK.

Hendy, K. C., & Farrell, P. S. (1997). *Implementing a model of human information processing in a task network simulation environment.* DCIEM 97-R-71, Ontario, Canada.

Holland, J. (1992). *Adaptation in natural and artificial systems.* Cambridge, MA: MIT Press.

Kaelbing, L. P. (1986). An architecture for intelligent and reactive systems. In M. P. Georgeff & A. L. Lansky (Eds.), *Proceedings of the 1986 Workshop on Reasoning About Actions & Plans* (pp. 395-410). San Mateo, CA: Morgan Kaufmann Publishers.

Kroemer, K. H. E., Snook, S. H., & Meadows, S. K. (1988). Ergonomic models of anthropometry, human biomechanics and operator-equipment interfaces. In *Proceedings of a Workshop of the Commission on Behavioral and Social Sciences and Education.* Washington, DC: The National Academies Press.

Laughery, R. (1998, December 13-16). Computer simulation as a tool for studying human-centered systems. In D. J. Medeiros, E. F. Watson, J. S. Carson & M. S. Manivannan (Eds.), *Proceedings of the 1998 Winter Simulation Conference,* Washington, DC (pp. 61-66). Los Alamitos, CA: IEEE Computer Society Press.

Lewin, K. (1935). *A dynamic theory of personality: Selected papers of Kurt Lewin.* New York: McGraw-Hill.

McCoy, M., & Levary, R. R. (2000). A rule-based pilot performance model. *International Journal of Systems Science, 31*(6), 713-729.

Olenick, S. M., & Carpenter, D. J. (2003). An updated international survey of computer models for fire and smoke. *Journal of Fire Protection Engineering, 13*(2), 87-110.

Orcut, G. H. (1986). Views on micro-analytic simulation modeling. In G. H. Orcut, J. Merz, & H. Quinke (Eds.), *Information research and resource reports: Micro-analytic simulation*

models to support social and financial policy (Vol. 7). Amsterdam: North-Holland.

Parker, S. K., Wall, T. D., & Cordery, J. L. (2001). Future work design research and practice: Towards an elaborated model of work design. *Journal of Occupational and Organizational Psychology, 74,* 413-440.

Pew, R. W., & Mavor, A. S. (Eds.). (1998). *Modeling human and organizational behavior.* Washington, DC: National Academic Press.

Sasou, K., Takano, K., & Yoshimura. S. (1996). Modeling of a team's decision-making process. *Safety Science, 24*(1), 13-33.

Schmidt, B. (2000). *The modeling of human behavior.* Ghent, Belgium: SCS Europe BVBA.

Schultz, K. L., Juran, D. C., & Boudreau, J. W. (1999). The effects of low inventory on the development of productivity norms. *Management Science, 45*(12), 1664-1678.

Schultz, K. L., Juran, D. C., Boudreau, J. W., McClain, J. O., & Thomas, L. J. (1998). Modeling and worker motivation in JIT production systems. *Management Science, 44*(12), 1595-1607.

Sebok, A. L., Hallbert, B. P., Plott, B. M., & Nash, S. S. (1997, June 8-13). Modeling crew behavior and diagnoses in the control room. In D. I. Gertman, D. L. Schurman, & H. S. Blackman (Eds.), *Proceedings of the 1997 IEEE 6th Conference on Human Factors and Power Plants,* Orlando, FL (pp. 7.13-7.17). New York: IEEE Press.

Shannon, R. E. (1975). *Systems simulation, the art and science.* London: Prentice-Hall.

Siebers, P. O. (2004). *The impact of human performance variation on the accuracy of manufacturing system simulation models.* PhD thesis, Cranfield University, UK.

Silverman, B. G., Might, R., Dubois, R., Shin, H., & Johns, M. (2001, May 15-17). Toward a human behavior modeling anthology for developing synthetic agents. *Proceedings of the 10th Conference on Computer-Generated Forces and Behavioral Representation,* Orlando, FL (pp. 277-285).

Skvoretz, J. (2003). Complexity theory and models for social networks. *Complexity, 8*(1), 47-55.

Spencer, M. B. (1987). The influence of irregularity of rest and activity on performance: A model based on time since sleep and time of day. *Ergonomics, 30*(9), 1275-1286.

Trisoglio, A. (1995, August 7-9). *Complexity: The challenges.* Paper presented at the Risk, Policy and Complexity workshop. Laxenburg, Austria.

Weiss, G. (Ed) (1999). *Multiagent systems: A Modern approach to distributed artificial intelligence.* Cambridge, MA: MIT Press.

Williams R., & Fletcher, C. (2002). Performance management and organizational effectiveness. In I. T. Robertson, M. Callinan, & D. Bartram (Eds.), *Organizational effectiveness: The role of psychology* (pp. 135-158). Chichester: John Wiley & Sons.

Wooldridge, M. (1999). Intelligent agents. In G. Weiss (Ed), *Multiagent systems: A modern approach to distributed artificial intelligence* (pp. 27-78). Cambridge, MA: MIT Press.

XJ Technologies. (2005). *Simulation software: AnyLogic.* Retrieved July 25, 2005, from http://www.xjtek.com

Ye, M., & Carley, K. M. (1995). Radar-soar: Towards an artificial organization composed of intelligent agents. *Journal of Mathematical Sociology, 20*(2/3), 219-246.

Yildizoglu, M. (2001). Connecting adaptive behavior and expectations in models of innovation: The potential role of artificial neural networks. *European Journal of Economic and Social Systems, 15*(3), 203-220.

KEY TERMS

Agent-Based Modeling: In the context of this chapter, this is a bottom-up approach that allows the behavior of human beings to be captured in a more realistic fashion. The artificial agents acting as representatives for real factory workers have to be designed to mimic the attributes and behaviors of their real-world counterparts as similarly as possible. The system's macro-observable properties emerge as a consequence of these attributes and behaviors, and the interactions between them.

Artificial White Room: Simulation of a laboratory as it is used by social scientists for data gathering under controlled conditions.

Direct Performance Indicators: Indicators that measure how the individual worker affects the system. Typical indicators are activity time (the actual time it takes a worker to complete a task that is usually a repetitive cycle), error rate (an indication of how well a worker conducts a task), and dependability (given that all conditions for a task to commence are met, when does the operator start the activity in response to a request?).

Direct Workers: Factory workers dedicated to predominately manual routines.

Discrete Event Simulation (DES): Modeling of a real system as it evolves over time by representing the changes as separate events, for the purpose of better understanding and/or improving that system.

Human Performance Variation (HPV): The variation in the time taken to complete a task by a direct worker under normal working conditions.

Indirect Performance Indicators: Indicators that measure how the system affects the workers, which in return might have an effect on the performance of the individual worker. Typical indicators are absenteeism (absence from the workplace for any reason other than official leave or those covered by collective agreements), accident rate (an indication of how safely the worker conducts his or her work), and staff turnover (the number of employees starting or finishing employment at a particular place of work over a given period).

KISS (Keep It Simple, Stupid) Principle: A popular maxim often invoked when discussing a design process as a reminder to avoid the unnecessary complexity that can arise during the design process.

Time and Motion Study: An analysis applied to a job or number of jobs to check the efficiency of the work method, the equipment used, and the worker. Each operation is studied minutely and analyzed in order to eliminate unnecessary motions and thus reduce production time and raise output, which increases productivity.

Worker Performance Modeling (WPM): Modeling of the processes and effects of human behavior within a working environment.

Section VI

Information Systems

Chapter XLIV
Toward an Agent-Oriented Paradigm of Information Systems

Hong Zhu
Oxford Brookes University, UK

ABSTRACT

This chapter presents a meta-model of information systems as a foundation for the methodology of caste-centric agent-oriented software development, which is suitable for applications on the Internet/Web platform and the utilization of mobile computing devices. In the model, the basic elements are agents classified into a number of castes. Agents are defined as active computational entities that encapsulate: (a) a set of state variables, (b) a set of actions that the agents are capable of performing, (c) a set of behaviour rules that determine when the agents will change their states and when to take actions, and (d) a definition of their environments in which they operate. Caste is the classifier of agents and the modular unit of the systems. It serves as the template that defines the structure and behaviour properties of agents, as class does for objects. Agents can be declared statically or created dynamically at runtime as instances of castes. This chapter also illustrates the advantages of agent-oriented information systems by an example.

INTRODUCTION

In recent years we have seen a rapid change in the hardware infrastructure and software platforms on which information systems operate. Notably, the Internet/Web as well as mobile devices, such as notebook computers, PDAs, and 3G mobile phones and wireless networks, are becoming ubiquitous. Proposals for effective utilisation of such flexible devices and the Internet infrastructure have been advanced, such as Web services, semantic Web, grid computing, and so on. These techniques provide a bright vision for the future computer applications, especially for management information systems. However, a big problem remains open: how software should be developed.

In the past two decades, object-orientation (OO) has been a successful mainstream paradigm for the analysis, design, and implementation of software, especially information systems. However, software engineers are currently confronted with a number of challenges

in the development of Web-based information systems, especially in the construction of service-oriented systems, due to the new features of the Internet and World Wide Web. One of the main challenges comes from the autonomous feature of the hardware and software resources on the Internet/Web. It is unnatural to model autonomous resources within the OO meta-model, which considers everything as objects.

In the past two decades, agent technology has been developed mostly as an artificial intelligence endeavour (cf. Huhns & Singh, 1997). It is partly inspired in the observations and modelling of autonomous and emergent behaviours in the societies of human beings or insects and animals. It has long been regarded as a viable solution to the development of complicated applications in dynamic environments such as the Internet (Jennings & Wooldridge, 1998). However, existing agent-based systems have been developed in *ad hoc* methods without proper methodology, language, and tool supports. It is widely recognised that the lack of rigour has hampered the wide adoption of agent technology in IT industry.

In this chapter, we adapt and extend the principles of OO and propose a new meta-model of information systems based on the concept of agents. We will first present a meta-model of such agent-oriented information systems (AOISs), and then demonstrate the features of AOISs with an example and compare agent-orientation (AO) with traditional approaches. The readers are referred to Shan, Shen, Wang, and Zhu (2006) for the aspects on the methodology, languages, and tools that support the development of such AOISs.

The remainder of the chapter is organised as follows. We will first give an informal introduction to AOIS, which is followed by a formal definition of the meta-model. We will then illustrate the features of an AOIS with an example and compares it with traditional approaches to the development of information systems. Finally, we conclude the chapter with a discussion of related works and further work.

BASIC CONCEPTS

In our conceptual model, the basic unit that forms an information system is agent. Because there is no widely accepted definition of the concept of agent and multi-agent systems (MASs), it is worthy clarifying what we mean by agent and MAS, and how such systems work. Our conceptual model can be characterized by a set of pseudo-equations. Each pseudo-equation defines a key feature of a MAS.

The Structures and Operations of Agents and Multi-Agent Systems

Pseudo-equation 1 states that agents are defined as real-time active computational entities that encapsulate data, operations, and behaviours, and situate in their designated environments.

$$Agent = \langle Data, Operations, Behaviour \rangle_{Environment} \quad (1)$$

Here, data represent an agent's state. Operations are the actions that the agent can take. Behaviour is described by a set of rules that determine how the agent behaves in the context of its designated environment. By encapsulation, we mean that an agent's state can only be changed by the agent itself. Figure 1 illustrates the control structure of agent's behaviour.

There is a fundamental difference between objects and agents. In the structure of objects, there is no explicitly programmed behaviour rule. Instead, there is a fix behaviour rule for all objects: to execute a method if and only if it receives a message that invokes the method. In contrast, agents' behaviours are not simply driven by messages, although they can be so.

Figure 1. The control structure of agent's behaviour

```
Begin
    Initialise state;
    Loop
        Perceive the visible actions and states of the agents in its
            environment;
        Take actions and change state according to the situation in
            the environment and its internal state;
    end of loop;
end
```

For example, an agent can have a behaviour rule that enables it to take an action when it has not received any message for a certain period of time. Moreover, when an agent receives a message that requests the agent to take a specific action, the agent can decide whether to do so, for instance, according to its internal state or the origin of the request. In other words, the agent can decide 'when to go' and 'whether to say no' according to an explicitly specified set of behaviour rules. In this sense, an agent can be not only reactive as driven by the external events, but also be proactive, that is, being able to initiate interactions with the outside.

Despite this fundamental difference, objects can be considered as agents in a degenerate form as argued in Zhu (2001a). In particular, object is a special case of agent in the sense that it has a fixed rule of behaviour: "execute the corresponding method when receives a message." Consequently, in our conceptual model, a MAS consists of agents and nothing but agents as stated in pseudo-equation 2.

$$MAS = \{Agent_n\}, n \in Integer \quad (2)$$

Organisations of Multi-Agent Systems

In our conceptual model, the classifier of agents is called *caste.* Caste is the set of basic building blocks in the design and implementation of a MAS. As a modular language facility, a caste serves as a template that describes the structure and behaviour properties of agents. Pseudo-equation 3 states that a caste at time *t* defines a set of agents that have the same structural and behavioural characteristics.

$$Caste_t = \{Agent \mid Structure \,\&\, Behaviour\ properties\} \quad (3)$$

While a collection of castes represent various types of participants in the problem domain, the structure of the problem domain is captured with certain relationships between castes.

Caste Membership and Migration Relations

Agents are classified into various castes in a way similar to how data are classified into types and objects are classified into classes. In other words, agents are instances of castes just like objects are instances of classes. In the development of an AOIS, caste is the modular programming unit that serves as the template of agents so that agents can be created as an instance of a caste at runtime or declared statically. When an agent is created as an instance of a caste, it will have all the structural and behavioural features defined by the caste.

However, different from the notion of class in OO, caste allows dynamic classification. That is, an agent can change its caste membership (called *casteship* in the sequel) at runtime. The weakness of the static object-class relationship in current mainstream OO programming has been widely recognized. Proposals have been advanced, for example, to allow objects' dynamic reclassification (Drossopoulou, Damiani, Dezani-Ciancaglini, & Giannini, 2002). In our model, dynamic classification is an integral part of agents' behaviour capability, which can be naturally represented through agents' behaviour rules to change its casteship. An agent can take an action to join a caste or

retreat from a caste at run-time. When an agent retreats from a caste, it will lose the structural and behavioural features of the caste. When it joins a caste, it will obtain the structural and behavioural features. Dynamic casteship allows users to model the real world with MAS naturally, and to maximize the flexibility and power of agent technology. For example, Zhu and Lightfoot (2003) demonstrated that agents' ability to dynamically change their roles can be naturally represented by dynamic casteship.

A migration relation that represents agents' dynamic casteship can, therefore, be defined on castes. There are two types of migration relations: migrate and participate. A *participate* relation from caste *A* to caste *B* means that agents of caste *A* can join caste *B* without quitting from caste *A*. A *migrate* relation from caste *A* to caste *B* means that agents of caste *A* can join caste *B* and then quit from caste *A*. For example, in a university information system, we would have castes MSc_Student, PhD_Student and Staff to represent the components that collect the information about and deliver the services to various types of users. A participate relation from PhD_Student to Staff can be defined to represent the possibility that a PhD student may be employed as a staff member, such as a teaching assistant, while registered as a student. When an agent of PhD_Student joins the Staff caste, it will obtain all the structural and behavioural features of the caste Staff without losing its original structural and behavioural features defined in the caste PhD_Student. For example, suppose that the Staff caste defines a state variable Salary while PhD_Student does not. When an agent of PhD_Student joins Staff, it will have an additional state variable Salary as defined in the Staff caste. A migrate relation can be defined from caste MSc_Student to PhD_Student since an MSc student may become a PhD student after graduation, but it cannot be an MSc student and a PhD student at the same time. The agent will lose all the structural and behavioural features defined in the caste MSc_Student, but obtain all the structural and behavioural features that are defined by the PhD_Student caste. Suppose that the PhD_Student caste has a state variable Office_Address, while the MSc_Student caste has a state variable Laboratory. When an agent of MSc_Student moves to PhD_Student, it will lose the variable Laboratory and obtain a new variable Office_Address.

Inheritance Relation

Inheritance relations can be specified between castes. Caste *A* inherits caste *B* means that any agents of caste *A* have all structural, behavioural, and environmental features of caste *B*. Our model also allows multiple classifications—that is, an agent can belong to more than one caste at the same time. Consequently, a caste can inherit more than one caste.

Whole-Part Relations

In our model, an agent may contain a number of components that are also agents. The former is called compound agent of the latter. In such a case, there exists a whole-part relationship between the compound and the component agents. We identify three types of whole-part relationships between agents according to the ways a component agent is bound to the compound agent.

The strongest binding between a compound agent and its components is *composite*. A composition relation from caste *A* to caste *B* means that an agent *b* in caste *B* contains an agent *a* in caste *A*. The compound agent *b* is responsible for creation and destruction of its component *a*. If the compound agent *b* no longer exists or quits from caste *B*, the component agent *a* will be destroyed and hence not exist.

The weakest binding is *aggregate*. An aggregate relation from caste *A* to caste *B* means that, although an agent *b* in caste *B* contains a component agent *a* of caste *A*, the component agent *a* is independent of the compound agent *b* in the sense that *b* does not affect *a*'s existence or casteship. If agent *b* is destroyed or quits from caste *B*, agent *a* can still survive and be a member of caste *A*.

The third whole-part relation is called *congregate*. It means that if the compound agent is destroyed, the component agent will still exist, but it will lose the casteship of the component caste. For example, a university consists of a number of individuals as its members. If the university is destroyed, the individuals should still exist. However, they will lose the membership of the university. Therefore, in the university information system, the whole-part relationship between the caste University and the caste University Member is a congregation relation. This relationship is different from the relationship between a university and its departments. Departments are components of a university. If a university is destroyed, its departments will no long exist. The whole-part relationship between the castes University and Department is therefore a composition relation. The composition and aggregation relation in our conceptual model is similar to the composition and aggregation in UML, respectively. However, congregation is a novel concept, which has not been recognized in the research on OO modelling of whole-part relations (cf. Barbier, Henderson-Sellers, Le Parc, & Bruel, 2003).

Communications and Environment

In our conceptual model, an agent's state variables and actions are divided into two kinds: visible ones and invisible (or internal) ones. When an agent takes a visible action, it generates an event that can be observed by other agents in the system. An agent taking an internal action generates an event that can only be perceived by its components, which are also agents. Similarly, the value of a visible state variable can be observed by other agents, while the value of an internal state can only be observed by its components. Notice that our use of the term 'visibility' is different from the traditional concept of scope used in OO languages.

The concept of visibility of an agent's actions and state variables forms the basic communication mechanism in our conceptual model. Agents communicate with each other by taking visible actions and changing visible state variables, and by observing other agents' visible actions and visible states, as shown in pseudo-equation 4.

$$A \rightarrow B = A.Action \,\&\, B.Observation \tag{4}$$

An agent's visible actions are not necessarily observed by all agents in the system. They are only observed by those agents who are interested in the agent's behaviour and regard the agent as a part of their environments. The environment of an agent in a MAS at time moment *t* is a subset of the agents in the system. As illustrated in pseudo-equation 5, from a given agent's point of view, only those in its environment are visible. In particular, from agent *A*'s point of view, agent *B* is visible means that agent *A* can perceive the visible actions taken by agent *B* and obtain the value of agent *B*'s visible state variables at run-time.

$$Environment_t(Agent, MAS) \subseteq MAS - \{Agent\} \tag{5}$$

In our model, the environment of an agent is required to be explicitly defined so that the agents in a MAS are designed and implemented with a *designated environment*. In other words, the environment of an agent is specified but allowed to vary within a certain range when an agent is designed. We can describe the envi-

ronment of a caste, for example, as the set of agents in a number of particular castes. An environment such as that described is neither closed, nor fixed, nor totally open. Once an agent joins a caste, its environment is combined with the specified environment of the caste. Hence, the agents in the caste's environment become visible. The environment also changes when other agents join the caste in the agent's environment.

FORMAL DEFINITION OF THE META-MODEL

We now formally define the conceptual model using mathematical notions and notations. Readers can skip over this section if not interested in the formal treatment of the subject.

Multi-Agent Systems

Agents behave in real time concurrently and autonomously. A time moment is an element in a time index set T, which is defined as a subset of real numbers $[t_0, \infty)$, i.e. $T = \{t \mid t \in R \,\&\, t > t_0\}$, where t_0 can be any real number. The structure of agents consists of four elements—that is, each agent A is 4-tuple $\langle S_{A,t}, \Sigma_{A,t}, R_{A,t}, E_{A,t} \rangle$, where:

a. $S_{A,t}$ *is the state space*. We also write $S^V_{A,t}$ and $S^I_{A,t}$ to denote the visible and internal parts of the state space $S_{A,t}$, respectively. Thus, $S_{A,t} = S^V_{A,t} \times S^I_{A,t}$.

b. $\Sigma_{A,t}$ *is the set of actions* that the agent is capable of performing. We write $\Sigma^V_{A,t}$ and $\Sigma^I_{A,t}$ to denote the sets of visible and internal actions, respectively. Thus, $\Sigma_{A,t} = \Sigma^V_{A,t} \cup \Sigma^I_{A,t}$, where $\Sigma^V_{A,t} \cap \Sigma^I_{A,t} = \varnothing$.

c. $R_{A,t}$ *is the set of behaviour rules* that determine how the agent changes its state and which action to take at what circumstances.

d. $E_{A,t}$ *is the designated environment of the agent*.

In general, the set of agents in a MAS may change during execution as agents may be quit from the system or join the system at runtime. Let $\{A_1, A_2, \cdots, A_n\}_t$ be the set of agents in the system at time moment t. These agents are members of castes $C_1, C_2, \cdots, C_m$. The *casteship* of agent A at time moment t to caste C is denoted by $A \in_t C$. We write $Caste_t(A)$ to denote the set of castes that agent A belongs to at time moment t (i.e., $Caste_t(A) = \{C \mid A \in_t C\}$). An agent can join a caste C by taking the action of $JOIN(C)$ and quit from a caste C by taking the action $QUIT(C)$.

An inheritance relation between castes is defined as partial ordering relation denoted by $\prec$. We have that at all times t, $A \in_t C \wedge C \prec C' \Rightarrow A \in_t C'$. We assume that the set of castes and the inheritance relations between them do not change at runtime. Each caste describes its agents' structure, behaviour, and environment in the form shown in Figure 2.

In Figure 2, the clause '$C \Leftarrow C_1, C_2, \cdots, C_k$' specifies that caste C inherits castes $C_1, C_2, \cdots, C_k$. Therefore, $C \prec C_i, i = 1, \cdots, k$. The VAR clause declares a set of state variables of the caste in addition to what it inherits, where the visible variables $S^V(C)$ and invisible variables $S^I(C)$ of the caste C are:

$$S^V(C) = \{v_1 : T_1, \cdots, v_m : T_m\} \cup \bigcup_{i=1}^{k} S^V(C_i) \qquad (6)$$

Figure 2. Structure of caste description

```
Caste C <= C1, ..., Ck; (* inheritance relationship*)
   ENVIRONMENT E1, ..., Ew; (*description of the environment*)
   VAR      *v1:T1, ..., *vm:Tm; (* visible state variables *)
            u1:S1, ..., ul:Sl; (* invisible state variables *)
   ACTION   *A1(p1,1, ..., p1,n1), ..., *As(ps,1,..., ps,ns); (* visible actions *)
            B1(q1,1,..., q1,m1), ..., Bt(qt,1,..., qt,mt); (* invisible actions*)
   RULES R1, R2, ..., Rh (* Behaviour rules *)
End C.
```

$$S^I(C) = \{u_1 : S_1, \cdots, u_j : S_j\} \cup \bigcup_{i=1}^{k} S^I(C_i). \quad (7)$$

The ACTION clause defines a set of actions of the caste. The visible actions $\Sigma^V(C)$ and invisible actions $\Sigma^I(C)$ of caste C are:

$$\Sigma^V(C) = \{A_1(p_{1,1}, \cdots, p_{1,n_1}), \cdots, A_s(p_{s,1}, \cdots, p_{s,n_s})\} \cup \bigcup_{i=1}^{k} \Sigma^V(C_i) \quad (8)$$

$$\Sigma^I(C) = \{B_1(q_{1,1}, \cdots, q_{1,m_1}), \cdots, B_t(q_{t,1}, \cdots, q_{t,m_t})\} \cup \bigcup_{i=1}^{k} \Sigma^I(C_i) \quad (9)$$

It is assumed that variables and action identifiers are unique in each caste declaration. Duplicated declarations of identifiers in a caste are not allowed.

The environment of the agents of a caste is explicitly specified in the ENVIROMENT clause in the following forms:

a. '**agent name**' indicates a specific agent in the system;
b. '**All: caste-name**' means all the agents of the caste;
c. '**agent-variable: caste-name**' is a variable that ranges over the caste. It can be assigned to any agent in the caste.

Let $ENV_t(C)$ denote the environment for caste C at time moment t:

$$ENV_t(C) = \bigcup_{i=1}^{w} [\![E_i]\!]_t \cup \bigcup_{i=1}^{k} ENV_t(C_i), \quad (10)$$

where $E_i, i = 1, \cdots, w$ are the environment description clauses in caste C's definition, and $[\![E]\!]_t$ denotes the semantics of an environment description clause E at time moment t. $[\![E]\!]_t$ is defined as follows.

$$[\![agent]\!]_t = \begin{cases} \{agent\}, & \text{if } agent \text{ is in the system at time } t; \\ \varnothing, & \text{if } agent \text{ is not in the system at time } t. \end{cases} \quad (11)$$

$$[\![All : Caste]\!]_t = \{X \mid X \in_t Caste\}; \quad (12)$$

$$[\![x : C]\!]_t = \begin{cases} \{A\}, & \text{if } x = A \text{ and } A \text{ is in the system at time } t; \\ \varnothing, & \text{otherwise.} \end{cases} \quad (13)$$

The set of rules $RULE(C)$ that agents of caste C must obey is:

$$RULE(C) = \{R_1, R_2, \cdots, R_h\} \cup \bigcup_{i=1}^{k} RULE(C_i). \quad (14)$$

Let A be any given agent and $Caste_t(A) = \{C_1, C_2, \cdots, C_n\}$; the following equations define the structural and behavioural properties of the agent at each time moment t.

$$S^V_{A,t} = \bigcup_{i=1}^{n} S^V(C_i), \text{ and } S^I_{A,t} = \bigcup_{i=1}^{n} S^I(C_i). \quad (15)$$

$$\Sigma^V_{A,t} = \bigcup_{i=1}^{n} \Sigma^V(C_i), \text{ and } \Sigma^I_{A,t} = \bigcup_{i=1}^{n} \Sigma^I(C_i). \quad (16)$$

$$E_{A,t} = \bigcup_{i=1}^{n} ENV_t(C_i). \quad (17)$$

$$R_{A,t} = \bigcup_{i=1}^{n} RULE(C_i). \quad (18)$$

Dynamic Semantics

A *run r* of a MAS is a mapping from time T to the set $\prod_{i=1}^{n} S_{A_i,t} \times \Sigma_{A_i,t}$. The behaviour of a MAS is defined by the set R of possible runs. For any given run r of the system, a mapping h from T to $S_{A,t} \times \Sigma_{A,t}$ is a run of agent A in the context of r, if $\forall t \in T.h(t) = r_A(t)$, where $r_A(t)$ is the restriction of $r(t)$ on $S_{A,t} \times \Sigma_{A,t}$. In the sequel, we use

$R_A = \{r_A \mid r \in R\}$ to denote the behaviour of agent A in the system. We assume that a MAS has the following properties:

- Actions are instantaneous, that is, for all $t_1, t_2 \in T$, if $t_1 \neq t_2, r_A(t_1)$ is regarded as different from $r_A(t_2)$.
- An agent may take no action at a time moment t, that is, the agent is silent at time t. We use τ to denote silence.
- The actions taken by an agent are separable, that is, for all runs r, all agents A, there exists a real number $\varepsilon>0$ such that for all t, $r_A^C(t) \neq \tau \Rightarrow (\forall x \in T.(t < x \leq t+\varepsilon \Rightarrow r_A^C(x) = \tau))$, where $r_A^C(t)$ denotes the action taken by agent A at time moment t in the run r.

With the above assumptions, we can prove that an agent can take at most a finite number of non-silent actions in any finite period of time and a countable number of non-silent actions in its lifetime.

Notice that the global state S_g of the system at time t is the value of $\prod_{i=1}^{n} S_{A_i,t} \times \Sigma_{A_i,t}$. However, each agent A can only view the visible states and actions of the agents in its environment—that is, the part of S_g in the space $\prod_{X \in E_{A,t}} S_{X,t}^V \times \Sigma_{X,t}^V$.

A behaviour rule R for agent A defines a predicate P_R on the set of all possible execution histories of the agent in the context of the system's runs. Here, the context is the agent's view of the history of the environment. Therefore, such a possible history is a mapping from the time $t \in T$ to the set $S_{A,t} \times \Sigma_{A,t} \times \prod_{X \in E_{A,t}} \left(S_{X,t}^V \times \Sigma_{X,t}^V\right)$. For a possible history $\psi_{A,t}$ of agent A up to time moment t, $P_R(\psi_{A,t}) = true$ means that agent A's behaviour at time moment t satisfies the behaviour rule R. An execution r_A of agent A in the context of a run r is valid, if for all time moment t and all behaviour rules $R \in R_{A,t}$, agent A's behaviour at time t satisfies rule R. A run r of a MAS M is valid, if all agents A's behaviours are valid in r.

The meta-model does not define how a behaviour rule should be defined. In fact, languages at different levels of abstraction can have their own ways of defining behaviour rules. For example, in the formal specification language SLABS, behaviour rules are defined in the following form:

<Pattern> | [<Probability>] → <Action>,
[***if*** <Scenario>];
[***where*** <Pre-condition>]

where <Pattern> defines a pattern of the agent's behaviour so far, <Scenario> specifies the scenario in the environment, <Pre-condition> specifies the pre-condition that the rule applies, <Action> specifies the action to be take by the agent, and <Probability> defines the probability that the agent will take the action as specified. The formal semantics of such behaviour rules and a formal system for reasoning about the behaviour of MAS can be found in Zhu (2005). In modelling language CAMLE, behaviour rules are defined in the form of behaviour diagrams (Shan & Zhu, 2004, 2005). In programming language SLABSp, the behaviour rules take the form of a conditional statement that uses a scenario as the guard condition (Wang, Shen, & Zhu, 2005).

ILLUSTRATIVE EXAMPLE

Generally speaking, information systems are systems that collect, store, process, and use information to fulfil certain tasks and provide certain services. Different paradigms of software and information system development methods differ in the way the information processing functions and storage facilities are structured, organised, and used, and how such systems are constructed and evolved accordingly.

From the structure point of view, traditional *structured methods* separate the storage of information from their processing methods. Functions are hierarchically decomposed into sub-functions and then sub-sub-functions, and so on. In *object-oriented methods,* a set of related data and their processing methods are encapsulated into one computational entity called object. The relationships between objects resemble their counterparts in the real world. The classification of objects into classes represents the structural similarity between objects. Inheritance relation represents specialisation and dissimilarities. Method invocation association represents functional dependences. Whole-part relations represent hierarchical structural decomposition. These relations help the maintenance and evolution of information systems as their real-world environment evolves, as discussed in the introduction of the chapter. However, objects in the mainstream OO paradigm are passive entities. Models of information systems that consist of a large number of active and autonomous information processing components cannot be represented naturally and close to the real-world counterparts in the structure. In the meta-model of *caste-centric AO method* proposed in this chapter, a set of data, their processing methods, and the rules on how the processing methods are to be used are encapsulated into one computational entity called agent to represent an active information processing element. The relationships between such elements and their environments are represented in the classification of agents for the similarity in their structural and behaviour features, in inheritance relations for the dissimilarity and specialisation, in whole-part relations for structural decomposition, in visibility in the environment for communications and collaborations with each other, and so on. This model inherits many features of OO, but further captures the features of active information processing elements in modern information systems enabled by the availability of mobile computing devices and the Internet/Web-based software platform.

In this section, we will present an illustrative example to highlight the differences between traditional approaches (such as structured and OO approaches) and the AO approach.

Suppose that a summer school is organised to teach a number of classes. Some classes are scheduled to be held at the same time, but at different classrooms. Students can choose the classes to attend. Students may be unfamiliar with the site where classes are. An information system is required to help students go to the right classrooms.

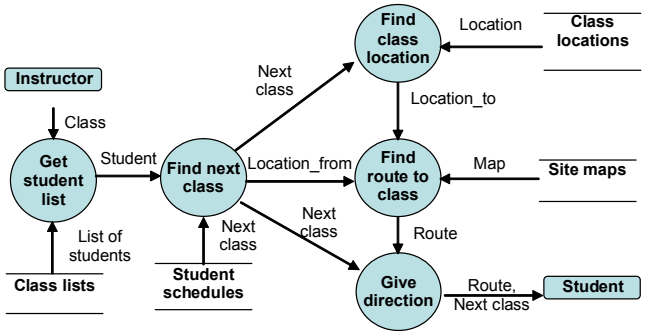
Solution of Structured Methods

In a structured methodology such as stepwise refinement, one would decompose the functionality of the system and find the steps to calculate the results from the input. For example, suppose the system is to be used by an instructor for helping a student in his/her class

Figure 3. Pseudo-code in structured programming

```
Begin
        1. Get the times and locations of the classes
        2. Get the class lists of the students
        3. Input the current class
        4. Get a list of students in the class
        5. For each person on the list Do:
                5.1. Find the next class that student is attending
                5.2. Find the location of the class
                5.3. Find the route from the classroom to the person's next class
                5.4. Tell the student how to get to their next class
End
```

Figure 4. Data flow diagram



to find the next classroom. One might come up with a solution similar to the following.

Using a structured analysis and design method, one might obtain the data flow diagram shown in Figure 4.

Object-Oriented Solution

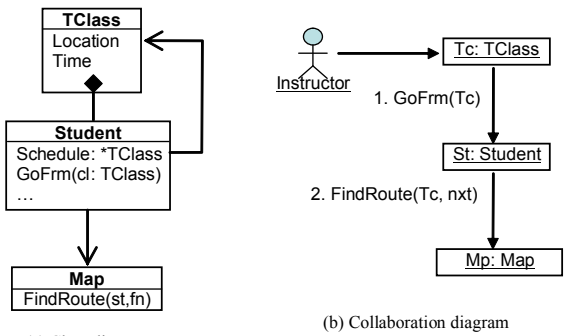
As Shalloway and Trott (2002) pointed out, although the above may be a good solution for computer systems, it is unnatural for a person to do. In other words, the model given in Figure 4 does not represent the real world. A better solution would be that the instructor post directions to go from this classroom to the other classrooms, and then inform everybody that the directions are posted at the back of the classroom and tell everybody to follow the direction and go to the next classroom. The alternative solution can be represented in an OO model shown in Figure 5.

Shalloway and Trott (2002) analysed the differences between the above two solutions and their impacts on software development. In addition to their analysis, we can also identify the following weakness of the OO solution. First, modelling students as objects means that they are passively driven by messages to perform actions of finding out the route and then going to the next classroom. They are controlled by other objects. However, in the real world, students may have autonomous behaviours. They are not directly controlled by anybody. Therefore, the model still does not exactly represent the real world. Second, it is the developer's responsibility to implement all the components of the system although the implementation can be done by programming the code, reusing existing code, using COTS components, and so forth. Once implementation is done, it is not to be changed during execution of the system.

Agent-Oriented Solution

A caste-centric AO solution would contain three castes to represent students, instructors, and the organizers of the summer school as

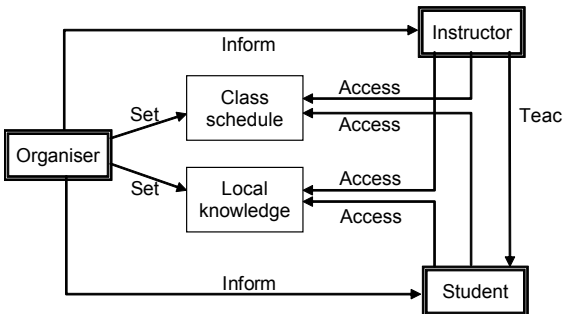
Figure 5. Object-oriented model in UML



(a) Class diagram

(b) Collaboration diagram

Figure 6. Collaboration diagram of AO solution



shown in Figure 6. The organizers will be responsible to set the class schedule and provide the local knowledge. They are also responsible to inform the instructors and the students of the availability of the schedule and local knowledge. The instructors will teach the students. Teachers may also access the class schedules and local knowledge to determine when and where to go to the classrooms and teach which subjects. The students will access the class schedule and local knowledge in order to determine where to go and how to get there. It is worthy noting that the schedule and local knowledge can be objects that are a degenerated form of agents as discussed in the previous section.

In comparison with the OO solution, the AO model is much closer to how the real world works. The main difference is in the release of instructors from the responsibility of controlling the students on when to find out the information about their next classes. The students can access the schedule and local knowledge whenever they like to do so. The impact of this difference is significant because this not only gives the freedom to decide when to access the local knowledge and class schedule, but also the possibility to have freedom in how to access them because other parts of the system need to know many less details of the caste. As far as the local knowledge, such as a map of the campus, and class schedule are represented in a standard format, agents that represent students can be any program that understand them. Therefore, they can be programs running on students' notebook computers, hand-held computers, mobile phones, and so forth. The software running on these computing devices can be different products, each suitable to the device that is owned by the student. This will significantly reduce the complexity of developing the system to make it suitable to run on many different hardware and software platforms. It will also enable the users to interact with the system in an interface that they are familiar with. It is unnecessary to integrate the programs that represent the student agents into the system before it is put into operation, because they can join the system at any time during the execution of the system. The software can be run on other servers on the Internet that provides, for example, services to display a map on the screen, to find routes in an electronic map, and to provide scheduling and event reminder services. Therefore, the complexity and cost to develop the system can be significantly reduced. It is also easier for the system to evolve.

CONCLUSION

In this chapter, we presented a meta-model of AOIS, in which the basic elements are agents. The meta-model enables us to model software systems very close to information system in the physical world so that the software systems are easier to understand, more reusable, and easier to modify. As illustrated in the example in this chapter, AO approach is suitable to the platform of Internet/Web and the utilisation of mobile devices.

In the literature of AO software engineering, there are a number of proposals to AO methodologies aiming at developing MASs

(Zambonelli, Jennings, & Wooldridge, 2003; Bresciani, Perini, Giorgini, Giunchiglia, & Mylopoulos 2004; Burrafato & Cossentino, 2002; Zambonelli & Omicini, 2004). A few formal models of agents and MASs have also been developed and investigated (cf. Myer & Schobbens, 1999). Among the most well-known formal models is the mentalistic model of BDI agents, in which each agent has mental states of belief, desire, and intention (Rao & Georgreff, 1991). The logic properties of mental states were formally studied in the framework of modal logics (e.g., Wooldridge, 2000). Such models are suitable for developing artificial intelligence applications. Application of them to information systems may need a revolutionary change in the software development paradigm. d'Inverno and Luck (2003) formally defined various concepts related to agents and MASs in the formal specification language Z. They regard agents as special cases of objects. This is the opposite to our approach. Semi-formal definitions of meta-models have also been proposed by Bernon, Cossentino, Gleizes, Turci, and Zambonelli (2005) and Odell, Nodine, and Levy (2005). They provide syntactical descriptions of the structures of agents and related notions, but no definition of their semantics. These models tend to include a large number of concepts including, for example, roles, agent groups, agent societies, capabilities, responsibilities, goals, plans, organisations, and so forth. Further research on these concepts and their properties and interrelationships is necessary before they can be language facilities. Our approach is caste-centric, that is, caste plays the central role in our methodology. In comparison with other meta-models, our meta-model is much simpler and clearer. For example, most other methodologies have the notion of roles which is not formally defined and is not a language facility. In our approach, agents that play the same role can be defined by a caste (Zhu, 2001b), which is a well-define language facility that can be implemented in a programming language (Wang et al., 2005). It can also be used to implement other agent concepts naturally, such as agent societies, protocols, and normative behaviours. The most important feature of our approach is that it is a natural evolution of the current mainstream paradigm of OO software development.

We are currently further investigating the design and implementation of AO programming languages based on the meta-model to support the development of service-oriented computing such as Web services (Zhu & Shan, 2005). We are also further developing automated software tools to support the whole development process to bridge the gaps between models, specifications, and their implementations on existing software platforms.

REFERENCES

Barbier, F., Henderson-Sellers, B., Le Parc, A., & Bruel, J.-M. (2003). Formalization of the whole-part relationship in the Unified Modelling Language. *IEEE Transactions of Software Engineering, 29*(5), 459-470.

Bernon, C., Cossentino, M., Gleizes, M.-P., Turci, P., & Zambonelli, F. (2005). A study of some multi-agent meta-models. *Proceedings of AOSE 2004* (LNCS 3382, pp. 62-77). Berlin: Springer-Verlag.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., & Mylopoulos, J. (2004). Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems, 8,* 203-236.

Burrafato, P., & Cossentino, M. (2002, May 27-28). Designing a multi-agent solution for a bookstore with the PASSI methodology. *Proceedings of AOIS-2002,* Toronto, Canada.

d'Inverno, M., & Luck, M. (2003). *Understanding agent systems*. Berlin: Springer-Verlag.

Drossopoulou, S., Damiani, F., Dezani-Ciancaglini, M., & Giannini, P. (2002). More dynamic object reclassifi-cation: FickleII. *ACM Transactions on Programming Language and Systems, 24*(2), 153-191.

Huhns, M., & Singh, M. P. (Eds.). (1997). *Readings in agents*. San Francisco: Morgan Kaufmann.

Jennings, N. R., & Wooldridge, M. J. (Eds.). (1998). *Agent technology: Foundations, applications and markets*. Berlin: Springer-Verlag.

Myer, J.-J., & Schobbens, P.-Y. (Eds.). (1999). *Formal models of agents—ESPRIT project model age final workshop selected papers*. Berlin: Springer-Verlag (LNAI 1760).

Odell, J., Nodine, M., & Levy, R. (2005). A metamodel for agents, roles and groups. *Proceedings of AOSE 2004* (LNCS 3382, pp. 78-92). Berlin: Springer-Verlag.

Rao, A. S., & Georgreff, M. P. (1991). Modelling rational agents within a BDI-architecture. *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning* (pp. 473-484).

Shalloway, A., & Trott, J. (2002). *Design patterns explained*. Reading, MA: Addison-Wesley.

Shan, L., Shen, R., Wang, J., & Zhu, H. (2006). Caste-centric development of agent-oriented information systems In J.-P. Rennard (Ed.), *Handbook of research on nature-inspired computing for economics and management*. Hershey, PA: Idea Group Reference.

Wang, J., Shen, R., & Zhu, H. (2005, July). Agent-oriented programming based on SLABS. *Proceedings of COMPSAC'05* (pp. 127-132), Edinburgh, UK.

Wooldridge, M. (2000). *Reasoning about rational agents*. Cambridge, MA: MIT Press.

Zambonelli, F., & Omicini, A. (2004). Challenges and research directions in agent-oriented software engineering. *Autonomous Agents and Multi-Agents Systems, 9,* 253-283.

Zambonelli, F., Jennings, N. R., & Wooldridge, M. (2003). Developing multi-agent systems: The Gaia methodology. *ACM Transactions on Software Engineering and Methodology, 12*(3), 317-370.

Zhu, H., & Lightfoot, D. (2003, August). Caste: A step beyond object orientation in modular programming languages. *Proceedings of JMLC'2003* (LNCS 2789, pp. 59-62), Austria. Berlin: Springer-Verlag.

Zhu, H. (2001a). SLABS: A formal specification language for agent-based systems. *International Journal of Software Engineering and Knowledge Engineering, 11*(5), 529-558.

Zhu, H. (2001b, July). The role of caste in formal specification of MAS. *Proceedings of PRIMA'2001* (pp. 1-15), Taipei, Taiwan. Berlin: Springer-Verlag (LNCS 2132).

Zhu, H. (2003, July). A formal specification language for agent-oriented software engineering. *Proceedings of AAMAS'2003* (pp. 1174-1175), Melbourne, Australia.

Zhu, H. (2005, July 14-16). Formal reasoning about emergent behaviour in MAS. *Proceedings of SEKE'05*, Taipei, Taiwan (pp. 280-285).

Zhu, H., & Shan, L. (2005, February 2-4). Agent-oriented modelling and specification of Web services. *Proceedings of WORDS'2005*, Sedona, CA (pp. 152-159).

Chapter XLV
Caste-Centric Development of Agent-Oriented Information Systems

Lijun Shan
National University of Defense Technology, China

Rui Shen
National Laboratory for Parallel and Distributed Processing, China

Ji Wang
National Laboratory for Parallel and Distributed Processing, China

Hong Zhu
Oxford Brookes University, UK

ABSTRACT

Based on the meta-model of information systems presented in Zhu (2006), this chapter presents a caste-centric agent-oriented methodology for evolutionary and collaborative development of information systems. It consists of a process model called growth model, and a set of agent-oriented languages and software tools that support various development activities in the process. At the requirements analysis phase, a modelling language and environment called CAMLE supports the analysis and design of information systems. The semi-formal models in CAMLE can be automatically transformed into formal specifications in SLABS, which is a formal specification language designed for formal engineering of multi-agent systems. At implementation, agent-oriented information systems are implemented directly in an agent-oriented programming language called SLABSp. The features of agent-oriented information systems in general and our methodology in particular are illustrated by an example throughout the chapter.

INTRODUCTION

In Zhu (2006), we presented a vision of future information systems through an agent-oriented meta-model. The promising features of the meta-model were illustrated in the context of software development on the Internet/Web platforms and the utilisation of mobile computing devices. In this chapter, we address the problem of how to develop such agent-oriented information systems (AOISs). Based on the meta-model introduced in Zhu (2006), we propose a methodology for developing an AOIS which consists of a process model that guides the development activities, along with a set of languages and software tools that support various development activities in the process.

The chapter is organised as follows. We begin by describing an information system used as the running example in the chapter. We then propose an evolutionary development process model for AOIS and outline the caste-centric agent-oriented modelling language and environment, CAMLE. The next section reviews the formal specification language SLABS, which stands for a Specification Language for agent-based systems. The focus then turns to implementation issues, and the SLABSp experimental programming language is briefly described. We conclude the chapter with a discussion of related work and further work.

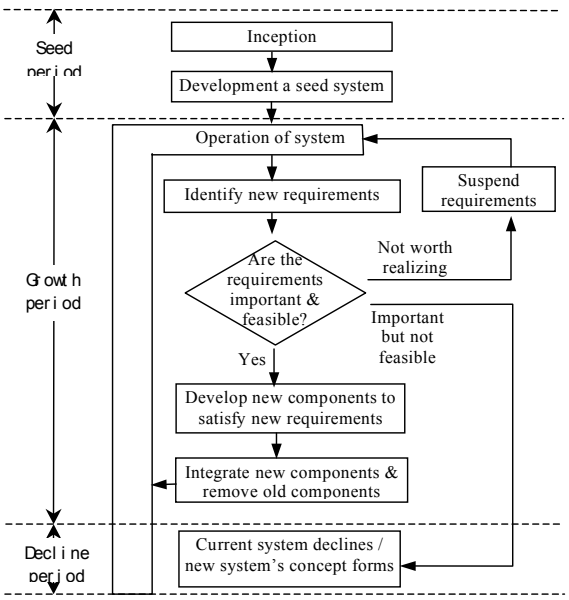
DESCRIPTION OF THE RUNNING EXAMPLE

We will use a simple, but non-trivial, information system to illustrate our methodology as a running example throughout the chapter. The example was proposed and used as a case study by FIPA's AUML Technique Committee (2004) to study agent-oriented modelling methods and notations. It was inspired by the procedure of the United Nations Security Council to pass a resolution. The description of the system follows.

The United Nation Security Council (UNSC) consists of a number of members, some permanent and others elected from UN members. Members become the chair of the Security Council in turn monthly. To pass a UNSC resolution, the following procedure is followed:

1. At least one member of UNSC submits a proposal to the current chair.
2. The chair distributes the proposal to all members of UNSC and sets a date for a vote on the proposal.
3. At a given date that the chair sets, a vote from the members is made.
4. Each member of the Security Council can vote either FOR or AGAINST or ABSTAIN.
5. The proposal becomes a UNSC resolution, if the majority of the members voted FOR and no permanent member voted AGAINST.
6. The members vote one at a time.
7. The chair calls members in a given order to vote, and the chair is always the last one to vote.
8. The vote is open (in other words, when one votes, all the other members know the vote).
9. The proposing member(s) can withdraw the proposal before the vote starts, and in that case no vote on the proposal will take place.
10. All members vote on the same day, one after another, so that the chair does not change within the vote call; but it is possible for the chair to change from one member to another between the time a proposal is submitted until it goes into vote. In this case the earlier chair must forward the proposal to the new one.
11. A vote is always finished in one day and no chair change happens on that day. The date of the vote is set by the chair.

Figure 1. Growth model of software development



In the remainder of the chapter, we will use the above as the initial requirements specification to demonstrate how agent-oriented information systems are analysed, modelled, designed, and implemented in our methodology.

DEVELOPMENT PROCESS

As discussed in Zhu (2006), one of the most attractive potential features of agent-oriented information systems is its strong support of the evolution of information systems and their collaborative developments. To realise this, we proposed a lifecycle model of software systems as shown in Figure 1 (Zhu, Greenwood, Huo, & Zhang, 2000; Zhu, 2002, 2004).

The lifecycle model is called *growth model* because it views information systems' lifecycle as a process of growth. From this point of view, a software system's lifecycle can be divided into three periods: the seed period, the growth period, and the decline period. When an information system is initially constructed and put into operation, it is relatively weak and small in terms of the services it provides, the volume of information it contains, and other non-functional attributes such as performance, security, and so forth. During the operation, the system grows in many directions and dimensions. New components may be integrated into the system to provide new services, while current components may be modified to improve the systems' functional or non-functional properties as users' new requirements are identified and implemented. The system gradually goes to the decline period, and dies when it cannot sustain more modification to meet new requirements. It is worth noting that different types of software systems may be suitable to different lifecycle strategies. For example, software systems of Lenman's S-type (1990, 2001) are more suitable to having a strong seed system and little modifications during the rest of its lifecycle, because such systems' requirements are well understood and well specified. The modifications are mostly corrections of errors in the software systems. However, Zhu (2004) argued that most information systems are Lenman's E-type systems whose requirements are changing, and hence they are evolutionary by nature. They are best developed following a growth strategy with the emphasis on the growth period. In comparison with other strategies that guide information system development, the growth strategy has a number of advantages. The first is the lower risk, because only the best understood requirements are implemented and integrated into the system. The investment in each step of the growth is smaller than implementing a huge system in one big bang. Second, it is more likely to have a shorter time delay from the recognition of a well-understood requirement to the delivery of the functionality. Complicated interactions between requirements can also be reduced and abated. Third, the developers can learn from previous develop-

ment experiences and improve their performance in the follow-up development of new components. They can gain confidence during the development process and see their results earlier than other development strategies. Finally, and most importantly, users' feedback can be obtained much earlier than other strategies, as each step of the growth process takes a much shorter period of time. This enables the users to clarify their requirements easily and guide the direction that the system develops. In fact, this strategy differs from the so-called staged development process model in its emphasis on taking users' feedback to guide the direction of software evolution.

To support the growth strategy, we designed and implemented a set of languages and tools for modelling, specification, and programming agent-oriented information systems. These languages and tools support various activities in the development process.

The modelling language and environment CAMLE supports:

- requirements elicitation and analysis by representing the current information system and the required system in agent-oriented models; and
- feasibility study of the requirements by analysing the required modifications to the existing system.

The formal specification language SLABS and its formal reasoning logic Scenario Calculus support:

- formal description of the requirements of the system under development so that new functionalities and services can be implemented as new components in the form of castes/agents; and
- formal reasoning of the design of the system/new components to ensure that the system will meet the requirements and that the new components can be integrated into the systems as expected.

The agent-oriented programming language SLABSp and its runtime support environment are used for:

- the implementation of the system/components according to the semi-formal specification in the CAMLE model and/or the formal specification in SLABS; and
- the testing of new components and the integration into the existing system.

In the following sections, we will describe each of these languages and tools, and illustrate their uses with the running example described earlier in the chapter.

MODELLING AND ANALYSIS

Modelling plays a crucial role in the development of the seed system and its evolution as the main tool of requirements analysis and system/component design. This section presents the modelling process, and the diagrammatic modelling language and environment of CAMLE (Shan & Zhu, 2003, 2004a, 2004b, 2005; Zhu & Shan, 2005).

Process of Modelling

In our methodology, modelling aims at representing the users' requirements with a set of agents at various granularities and organizing the agents into an information system. The key activities in the modelling and analysis phase include:

- Identify the agents and castes of agents in the system as well as the relationships between them, such as the *is-a* relation (inheritance), membership-shift relation (migration or participation), and whole-

part relation (aggregation, congregation, or composition). The artefact produced in this activity is a caste model for the system from the perspective of system architecture.

- Identify the agents' interaction patterns in various scenarios, and produce a set of collaboration models for the system from the perspective of dynamic behaviour. In order to specify the system in sufficient detail, an agent may be decomposed into a number of components, which are also agents. Then the interaction modelling proceeds to capture the interactions between the components. Eventually, the collaboration model is refined into a hierarchy, where collaboration models at various granularities specify the interactions between component agents at various abstraction levels. Along with agent decomposition, the caste model is enriched with further details to present the caste of agents at various granularities and the structural dependencies between them.
- For each caste, elaborate and specify how its agents perform actions and/or change states in typical scenarios so that a set of behaviour rules can be assigned to the caste. The artefact produced in this activity is a set of behaviour models, each associated to a caste in the system.

The result of the modelling is a system model comprising a set of diagrams that represent the system from various views and at different levels of abstraction. For example, in the UNSC system, a caste diagram is constructed to capture the organization structure, which comprises one *chair* and a number of *UNSC members*—either *permanent member* or *elected member*. Collaboration diagrams describe the typical scenarios of the interaction between UNSC members and the chair. Behaviour diagrams respectively for UNSC member and chair define their specific behaviour rules. More details are given in the next subsection.

During the growth phase of an existing agent-oriented information system, new components for providing new functions, services, and features are developed in the context of the existing system, which will be the operating environment of the new components. Therefore, the model of the existing system is the basis for the representation of the new requirements and the analysis of their feasibility. For example, if the organization of the United Nations Security Council is to be reformed to add a new type of member whose power on resolution is between elected member and permanent member, the UNSC information system can be modified accordingly by adding a new caste representing the new type of members.

The Modelling Language

CAMLE employs the multiple views principle to model complicated systems. There are three

Figure 2. Caste diagram: Notation and the UNSC example

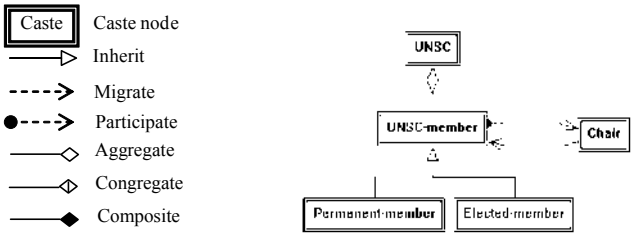
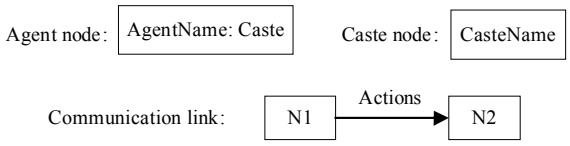


Figure 3. Notation of collaboration diagra



types of models in CAMLE: caste models, collaboration models, and behaviour models. Each model may consist of one or more diagrams.

A caste model usually consists of one caste diagram. Figure 2 shows the notation of caste diagrams and an example caste diagram of UNSC system. A caste diagram comprises a set of caste nodes representing various types of agents in the system, and a set of links representing various relationships between agents of the castes.

In the UNSC caste diagram, caste UNSC represents the organization, which is composed of a number of members represented by caste UNSC-member. The aggregate link between the UNSC-member and the UNSC denotes the part-whole relationship between the members and the organization. The rule that members take the role of Chair in rota is described by participate and migrate relation between caste UNSC-member and caste Chair. Two types of members are represented by two sub-castes of UNSC-member, Permanent-member and Elected-member, respectively.

The collaboration models capture agents' interaction patterns that represent dynamic behaviours of the system. The notation of collaboration diagrams is shown in Figure 3. A collaboration model may consist of a set of scenario-specific collaboration diagrams that represent the interactions between agents in specific scenarios, and a general collaboration diagram that summarises the communications between agents.

For example, Figure 4 depicts the collaboration model of UNSC. Figures 4a and 4b describe the interactions between agents in the

Figure 4. Collaboration model of UNSC information system

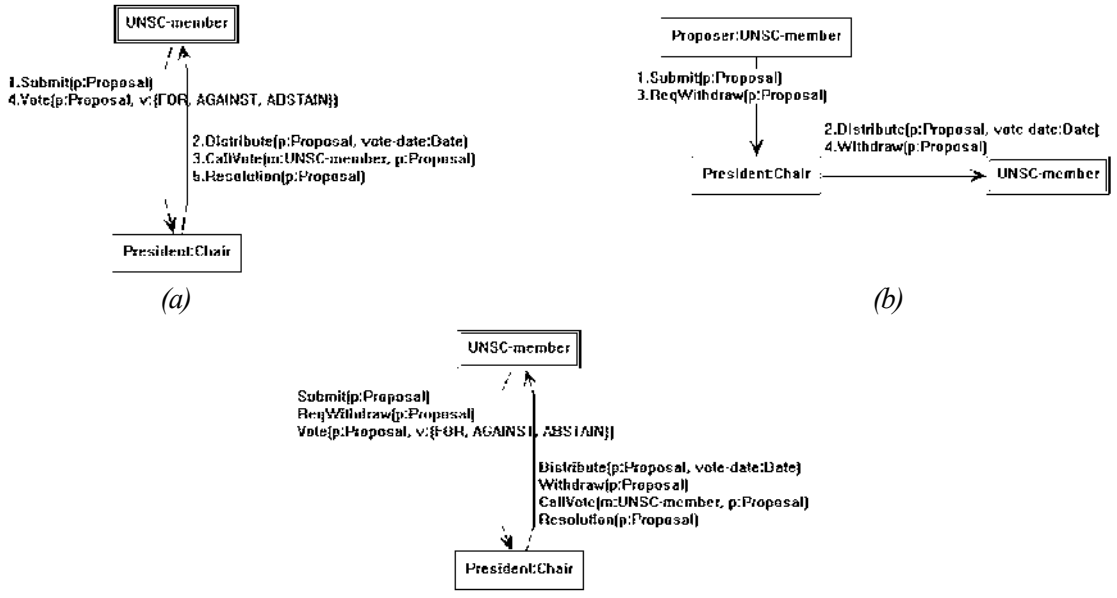
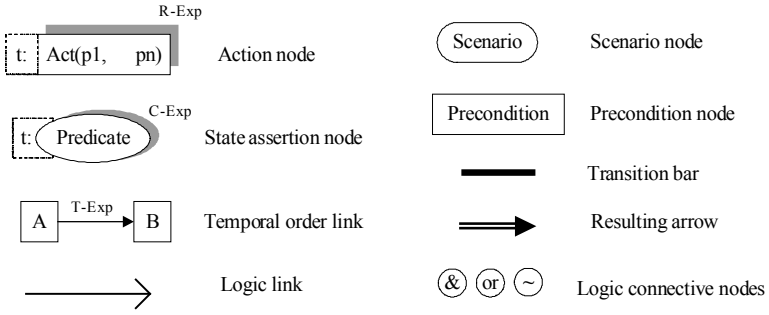


Figure 5. Notation of behaviour diagram



scenarios of voting on a proposal and withdrawal of a proposal, respectively. The general collaboration diagram, such as Figure 4c, describes all possible communications between all agents that may occur during the system's execution.

a. Scenario-specific diagram: Voting
b. Scenario-specific diagram: Withdraw
c. General collaboration diagram

Note that when an agent is decomposed into components, the interactions between the component agents also need to be specified. This results in a hierarchy of collaboration models defining the dynamic behaviours of agents at various granularities. Readers are referred to Shan and Zhu (2004b) for details about the process of collaboration modelling, the hierarchical structure of collaboration models, as well as examples.

While caste and collaboration models describe multi-agent systems at the macro-level from the perspective of an external observer, behaviour modelling adopts the internal or first-person view of each agent. It describes an agent's behaviour in terms of how it acts in certain scenarios of the environment at the micro-level. The notation of behaviour diagrams is shown in Figure 5. Readers are referred to Shan and Zhu (2003) for detailed explanation of the notation.

Each caste is associated with a behaviour diagram that describes the behaviour rules of its agents. In the UNSC example, there are two behaviour diagrams: one for caste Chair and the other for caste UNSC-member. Figure 6 depicts the behaviour diagram for caste Chair. The behaviour of a Chair agent is defined by four behaviour rules describing its actions under various circumstances, namely to distribute a proposal when some member submits the proposal, to withdraw the proposal when requested by the proposing member(s), to call all the members to vote on a proposal, and to quit from Chair when its turn finishes.

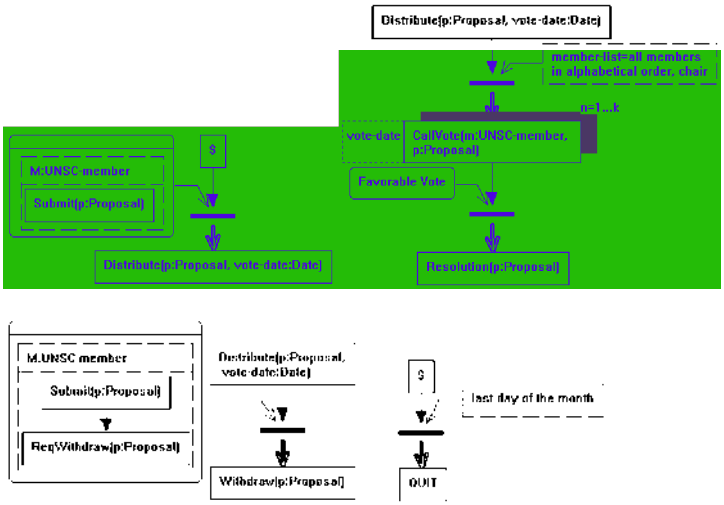
The castes Permanent-member and Elected-member inherit the behaviour rules of UNSC-member. They have no additional behaviour rules, thus require no different behaviour diagram than that of the UNSC-member.

The Modelling Environment

A software environment to support the process of analysis and modelling in CAMLE has been designed and implemented. It integrates the following set of tools:

- **Model Construction and Management Tools:** A set of interactive diagram edi-

Figure 6. Behaviour diagram of the chair in UNSC information system



tors with graphic user interface are provided to enable the creation, editing, and modification of various diagrams in CAMLE models. These diagrams are organized and managed into development projects. Reuse of models from other projects is enabled. Figure 7 shows a screen snapshot of the CAMLE environment's interface.

- **Consistency Checkers:** A set of consistency constraints is defined on the CAMLE language to ensure that a set of diagrams form a meaningful model of an information system. The consistency of a model is checked by a set of tools to identify any violence of the constraints. Details of the consistency constraints and the implementation of the checkers can be found in Shan and Zhu (2004a).
- **Specification Generator:** It transforms a well-defined model into a formal specification in SLABS. Details of the transformation algorithms can be found in Zhu and Shan (2005).

Figure 7. CAMLE's graphical user interface for model construction

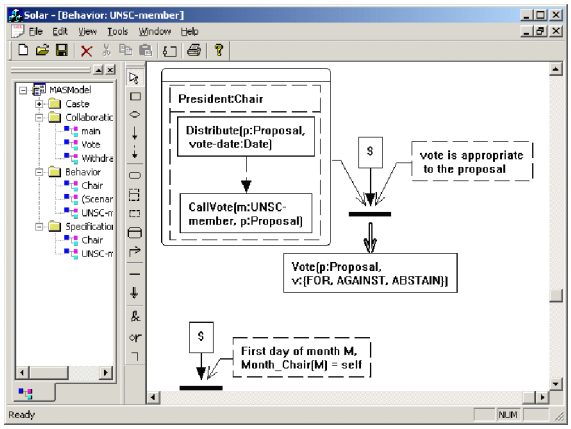
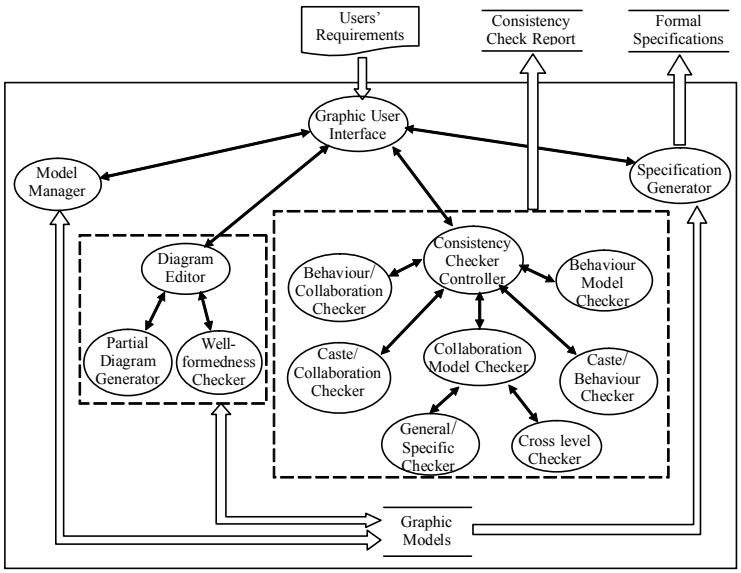


Figure 8 shows the architecture of the modelling environment. Readers are referred to Zhu and Shan (2005) for detailed description of the architecture and functionality of the CAMLE environment.

SPECIFICATION

One of the most appealing features of agent technology is its natural way to modularise

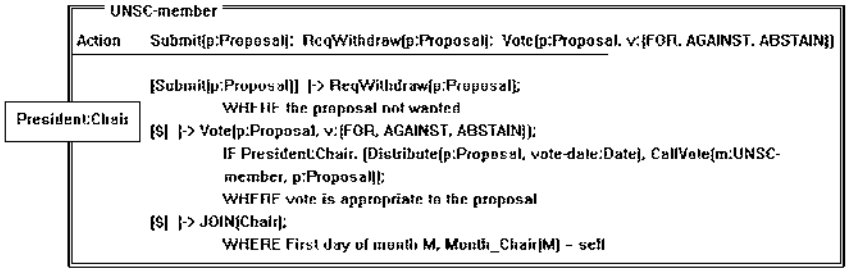
Figure 8. The architecture of CAMLE environment



complex systems in terms of multiple interacting autonomous components. This feature is supported by the language facility caste in SLABS for modular and composable specification of multi-agent systems. It bridges the gap between graphic modelling and implementation in the AOIS development process. The output of the modelling phase—a system model in CAMLE—is further analysed at the specification phase, which involves the following two main activities:

- **Generation of Formal Specifications:** As for all software developments, it is necessary to analyse the design of an agent-based system before the developers are committed to costly implementation. It is particularly true during the evolution of a system when new components are to be integrated into the existing system. Formal analysis of the new components in the context of the system is therefore highly desirable. However, the manual production of formal specifications of multi-agent systems is labour intensive, costly, time consuming, and error prone. With the help of the CAMLE modelling environment, formal specifications in SLABS can be automatically generated from graphic models in CAMLE.
- **Formal Analysis of the System:** Formal analysis can be applied on formal specifications in SLABS to prove the properties of the specified system. We have been devising a formal system Scenario Calculus to reason about the behaviours of multi-agent systems, especially their most complicated behaviours such as emergent behaviours (Zhu, 2005). If the formal reasoning about the system/new components based on the formal specification reveals that the system model is unsatisfactory on certain properties, the flow of the process goes back to the modelling phase to rectify the design. Thus, the process iterates the modelling and specification stages until a satisfactory model/specification is achieved.

Figure 9. Specification of UNSC-member caste in SLABS



The formal definition of the SLABS language and its meta-model can be found in Zhu (2001, 2003). A formal logic for reasoning about MASs' behaviours based on SLABS can be found in Zhu (2005).

Figure 9 shows an example of caste specification. It is the UNSC-member caste generated by the CAMLE environment's specification generator from the CAMLE model of the UNSC system.

IMPLEMENTATION

A distinctive feature of our agent-oriented development methodology of information systems is that we aim at the direct implementation of information systems with a novel agent-oriented programming language that is based on the meta-model of the agent-oriented information system described in Zhu (2006). Such a programming language can significantly narrow the gap between specification and implementation. This section presents our research on the design and implementation of the agent-oriented programming language SLABSp and illustrates the style of programming through the running example.

SLABSp is designed to support the caste-centric approach to agent-oriented software development methodology by extending the object-oriented programming language Java (Shen, Wang, & Zhu, 2004; Wang, Shen, & Zhu, 2005a, 2005b, 2005c). As shown in Figure 10, it extends Java with three key concepts and language facilities: caste, scenario, and environment descriptions.

These language facilities become the dominant language facilities in the implementations of an AOIS and significantly change the styles of programming. In particular, caste becomes the basic program unit from which a complicated software system is built. Although class in object-orientation can still be used in the programming, it is now mainly used to define encapsulated data types that agents manipulate and use to represent agent states. Other Java constructs, such as Import statements, Expressions, Statements, and so on, are still legal language facilities, but they are extended to include identifiers to refer to agent states and actions, which are represented by preceding '#' and '~', respectively. There are also the additional join and quit statements to enable agents to dynamically join into and quit from castes.

Another significant change of programming style is the result of the introduction of the scenario description language facility. The syntax of scenario description is given in Figure 10, where an expression in the form of (a) describes the situation that a specific agent behaves in a certain pattern, where the agent is referred to by its name or keyword self. Ex-

Figure 10. Syntax of SLABSp in EBNF

```
Caste ::= { Java-Import }
      'caste' Name [ ':' { Name / ',' }+ '{'
            { Environment }
            { State | Action | Rule }
      '}'
Agent ::= { Java-Import }
      'agent' Name [ ':' { Name / ',' }+ ] '{'
            { Environment }
            { State | Action | Rule }
      '}'
Environment ::= Name Id ';'

State ::= [ 'internal' ] Type '#'Id '(' Formal-Parameters ')' '{'
            { Java-Definition }
            'get' ':' Statement
            [ 'set' ':' Statement ]
      '}'
Action ::= [ 'internal' ] '~'Id '(' Formal-Parameters ')' '{'
            { Statement }
      '}'
Rule ::= 'rule' Id '(' Formal-Parameters ')'
      [ 'when' '(' Scenario ')' ]
      [ 'where' '(' Conditional-Expression ')' ]
      'do' '{' { Statement } '}'

Scenario ::= ( Name | 'self' ) Pattern                          (a)
      |  '<' [Number] ':' [Number] '>' Name Pattern             (b)
      |  Count-Conditional-Expression                           (c)
      |  Scenario ',' Scenario                                  (d)
      |  Scenario '|' Scenario                                  (e)
      |  '!' Scenario                                           (f)
      |  '(' Scenario ')'                                       (g)
Pattern ::=
      '\' { ( Action-Pattern | State-Assertion )  / ',' } '\'
Action-Pattern ::=
      ('~' | '*' | '~'Id '(' Parameters ')' ) ['^' Number]
State-Assertion ::= Conditional-Expression
```

pressions in the form of (b) describe the situation that the number of agents of a caste that behave in a certain pattern is within a specified interval, where the interval's boundaries are optional. The default value of the left boundary (i.e., the lower number) is 'zero'. When the right boundary is absent, it means 'all'—that is, the size of all the caste. The *count-conditional-expression* in the form of (c) is an extension of Java *conditional-expression* with count-expression. The result of evaluating a count-expression is the number of agents in a caste that behave in the pattern. Expressions in the form of (d), (e), and (f) are the logic 'and', 'or', and 'not' combination of scenarios in the above forms, respectively. Expressions in the form of (g) are used to change the preference of the logic combinations. The uses of scenario descriptions in conjunction with agents' visible actions and environment descriptions enable communication and collaboration among agents to be described at a high level of abstraction and in the same style of conditional expressions in structured programming.

For example, the UNSC system can be implemented in SLABSp as shown in Figure 13. Based on the specification of the UNSC system, caste Member has three behaviour rules. Rule Withdraw will enable the agent to request the proposal to be withdrawn when the agent regards the proposal as inappropriate. Rule Vote will guide the agent to vote on the proposal with a specific attitude when the chair calls the agent to vote. Rule AlternateJoin will trigger the agent to join caste Chair when it is its turn. Caste Chair extends caste Member with four additional rules. Rule Distribute will guide the Chair agent to distribute the proposal and schedule a voting date for each submitted proposal. Rule CallVote will direct the Chair agent to call the members to vote on the proposal on the voting date. Rule AlternateQuit will ask the current chair to quit from caste Chair when its turn finishes. Rule Resolution will define how a decision should be made based on the members' votes.

A runtime environment for the execution of multi-agent systems has been implemented as an extension of Java runtime environment. In particular, an automaton called the pattern process machine is designed and implemented to process patterns and scenarios. A compiler has been developed to translate SLABSp programs into Java and to execute in the runtime environment. More details can be found Shen et al. (2004) and Wang et al. (2005a, 2005b, 2005c).

The design and implementation of SLABSp demonstrate that caste and scenario are feasible as programming language facilities. Our experiences and experiments with the language clearly show that they provide power abstractions for AO programming. In particular, the

Figure 11. Fragments of UNSC system in SLABSp

```
import java.util.Calendar;

caste Member {

      ~Submit(Proposal proposal){
            // ...
      }

      ~ReqWithdraw(Proposal proposal){
            // ...
      }

      ~Vote(Proposal proposal, Attitude attitude){
            // ...
      }

      rule Withdraw(Proposal proposal)
      when ( self\~Submit(?proposal)\ )
      where ( proposal.isInappropriate() )
      do {
            ~ReqWithdraw(proposal);
      }

      rule Vote(Proposal proposal)
      when ( Chair\~CallVote(self, ?proposal)\  )
      do {
            // think about the proposal
            Attitude attitude = Attitude.FOR ; // or AGAINST, or ABSTAIN
            ~Vote(proposal, attitude);
      }

      rule AlternateJoin()
      when ( self\~\ )
      where (// the first day of a month
            ChairOfMonth(self, Calendar.getInstance().get(Calendar.MONTH))
        && (Calendar.getInstance().get(Calendar.DAY_OF_MONTH) ==
        Calendar.getInstance().getActualMinimum(Calendar.DAY_OF_MONTH)) )
      do {
            join Chair;
      }
}


// the caste for permanent members
caste PermanentMember : Member {
      // ...
}
```

```
import java.util.Date;
import java.util.Calendar;

caste Chair : Member {
      ~Distribute(Proposal proposal, Date date){ // ...
      }

      ~Withdraw(Proposal proposal){ // ...
      }

      ~Callvote(Member member, Proposal proposal){ // ...
      }

      ~Resolution(Proposal proposal){ // ...
      }

      rule Distribute(Proposal proposal)
      when ( self\~\, <1:1>Member\~Submit(?proposal)\ )
      do { // schedual voting a week later
            Date date = Calendar.getInstance()
                  .add(Calendar.DAY_OF_MONTH, 7).getTime();
            ~Distribute(proposal, date);
      }

      rule CallVote(Proposal proposal, Date date)
      when ( self\~Distribute(?proposal, ?date)\ )
      where ( Calendar.getInstance().getTime().equals(date) )
      do { // call all members (except the Chair) to vote, the call the Chair
            Collection members = Member#agents();
            members.remove(self);
            for(Member member: members)
                  ~CallVote(member, proposal);
            ~CallVote(self, proposal);
      }

      rule AlternateQuit()
      when ( self\~\ )
      where (// the last day of a month
            Calendar.getInstance().get(Calendar.DAY_OF_MONTH) ==
        Calendar.getInstance().getActualMaximum(Calendar.DAY_OF_MONTH) )
      do { quit Chair; }

      rule Resolution(Proposal proposal)
      when (  <:>Member\~Vote(?proposal, *)\, // all members have voted
            *Member\~Vote(proposal, FOR)\ > Member#population() / 2,
            !<1:>PermenantMember\~Vote(proposal, AGAINST)\ )
      do { ~Resolution(proposal); }
}
```

caste facility enables the modularity in the concept of agents to be realized directly and in full strength. An obvious advantage of using scenarios to define agents' behaviours is that it can significantly reduce the unnecessary, explicit, message-based communications among agents. This also enables AO programming at a very high level of abstraction.

CONCLUSION

We now conclude the chapter with a summary of our main ideas and research results, and a comparison of our work with related works.

Summary

Our caste-centric methodology of agent-oriented information systems is based on a well-defined meta-model presented in Zhu (2006). It consists of a process model called the *growth model,* a set of languages including a *modelling language CAMLE* for the requirements analysis and design, a *formal specification language SLABS* and a *programming language SLABSp*, and a set of support tools including *CAMLE's modelling environment,* a formal reasoning system *Scenario Calculus*, and a *runtime support environment* of agent-oriented programs. A number of case studies

on modelling, formal specification and verification, and programming have been conducted to develop the heuristics of using the languages and tools. Our methodology has the following features.

The methodology aims at modern information systems, especially those running on the Internet and the Web platforms. As argued in Zhu (2004), such systems belong to Lenman's E-type and are by nature evolutionary. The agent-oriented approach is very suitable for the development of such systems as we have shown in Zhu (2006). Moreover, the growth process model explicitly reflects the evolutionary characteristics of such systems and encourages the growth strategy, that is, the sustainable long-term evolution strategy of their lifecycle. This strategy is also strongly supported by the languages and tools.

The set of languages designed for use at different phases in the development and evolution of AOISs are based on a well-defined meta-model. The gaps between requirements specification, system and component design, and implementation are much smaller than their counterparts in other existing paradigms and approaches. In particular, the key concepts of agents and castes can be directly implemented in the agent-oriented programming language.

Our methodology is an extension of the current mainstream paradigm (the object-orientation) of information system development. In our model, object is a special degenerate form of agent. Agent-orientation provides a better metaphor for modelling the information systems in the real world than object-orientation. It can directly represent active and autonomous elements in information systems such as humans, independent information processing components such as Web services, and so on. It enables the design and implementation of computerised information systems in a structure that is closer to the structure of the system in the real world than object-orientation.

Finally, our approach to agent-orientation is caste-centric. In other words, caste plays the central role in our methodology. It is not just an abstract concept, but also a language facility that can be directly implemented in a programming language. It is the basic form of program unit from which complicated systems are constructed. It realises the kind of modularity inherent in the concept of agents. Our case studies show that caste can be used in a nice and straightforward way to model and implement various useful notions developed in agent technology, such as roles, agent society, collaboration protocols, normative behaviours, and so forth.

Related Work

Since Jennings (1999) advocated the notion of agent-oriented software engineering as a paradigm for building complex systems, a number of methodologies for agent-oriented software development have been proposed, such as MaSE (Wood & DeLoach, 2000), Gaia (Wooldridge, Jennings, & Kinny, 2000; Zambonelli, Jennings, & Wooldridge, 2003), Tropos (Bresciani, Perini, Giorgini, Giunchiglia, & Mylopoulos, 2004), and PASSI (Burrafato & Cossentino, 2002). A survey and analysis of the current state of the art in the research on agent-oriented software engineering can be found in Zambonilli and Omicini (2004).

As in early work on MAS engineering methodology, MaSE provides a development process covering the phases from capturing goals down to assembling agent classes and system design. Notations for representing system specifications in various stages and an environment supporting MAS development are developed (Wood & DeLoach, 2000). Gaia provides guides for analysis and design of agent-based systems with the view that a multi-agent system is a computational organization consisting of various interacting roles (Wooldridge et al., 2000). Role is adopted as the key concept, which is

associated with responsibilities, permissions, activities, and protocols. The new version Gaia methodology advocates computational organization abstractions as the key abstraction of agent-based computing (Zambonelli et al., 2003). Tropos methodology emphasizes the use of the notion of agent and all the related mentalistic notions in all phases of software development, and purports to cover the very early phases of requirements analysis (Bresciani et al., 2004).

Despite the subtle differences in the research aims and focuses, the above works hold the same beliefs that the concept of agent is on a higher level of abstraction than object, thus agent-orientation will bring more efficiency to software engineering than object-orientation. Most of the existing methodologies attempt to exploit agents' advantages, such as autonomy and sociality using the mentalistic notions including goal, plan, role, and so on. Our work distinguishes from them in that the notions of caste and scenario, instead of the mentalistic notions, are the basic concepts for embodying agents' power.

Further Work

There is still a long way to go before agent-orientation become a mature development paradigm of information systems. There are many issues remaining for future work. On the top of our research agenda is further investigation of the languages and tools in industrial context. We will connect the languages and tools with the ongoing development of Web technologies such as Web services, grid computing, and peer-to-peer computing. Another aspect of development methods that has not been discussed in depth in this chapter is testing, verification, and validation. We will further develop the formal reasoning system scenario calculus for analysing SLABS specifications and reasoning about the properties of emergent behaviours. We are also investigating software tools to support the formal reasoning. Automatic transformation from SLABS specifications to executable system is also in our agenda.

ACKNOWLEDGMENT

The work reported in this chapter is partly supported by the National Key Foundation Research and Development Program (973) of China under Grant No. 2005CB321802, the National High Technology R&D (863) Programme of China under grants No. 2002AA116070 and No. 2005AA113130, and the Program for New Century Excellent Talents in University.

REFERENCES

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., & Mylopoulos, J. (2004). Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems, 8*(3), 203-236.

Burrafato, P., & Cossentino, M. (2002). Designing a multi-agent solution for a bookstore with the PASSI methodology. *Proceedings of AOIS'02 at CAiSE'02.*

FIPA Agent UML Technique Committee. (2005). *Case studies of agent modelling: The Security Council of United Nations*. Retrieved May 30, 2005, from http://www.auml.org/auml/documents/

Jennings, N. R. (1999, June/July). Agent-oriented software engineering. In F. J. Garijo & M. Boman (Eds.), *Multi-Agent System Engineering, Proceedings of 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World,* Valencia, Spain (pp. 1-7). Berlin: Springer-Verlag (LNAI 1647).

Lehman, M. M., & Ramil, J. F. (2001). Rules and tools for software evolution planning and management. *Annals of Software Engineering, Special Issue on Software Management, 11*(1), 15-44.

Lehman, M. M. (1990). Uncertainty in computer application. *Communications of the ACM, 33*(5), 584-586.

Shan, L., & Zhu, H. (2003, October). Modelling and specification of scenarios and agent behaviour. *Proceedings of the IEEE/WIC Conference on Intelligent Agent Technology* (IAT'03) (pp. 32-38), Halifax, Canada.

Shan, L., & Zhu, H. (2004a, September). Consistency check in modelling multi-agent systems. *Proceedings of COMPSAC'04*, Hong Kong (pp. 114-121).

Shan, L., & Zhu, H. (2004b). Modelling cooperative multi-agent systems. *Proceedings of the 2nd International Workshop on Grid and Cooperative Computing* (LNCS 3033, pp. 994-1001), Shanghai, China. Berlin: Springer-Verlag.

Shan, L., & Zhu, H. (2005). CAMLE: A caste-centric agent-oriented modelling language and environment. In R. Choren, A. Garcia, C. Lucena, & A. Romanovsky (Eds.), *Software engineering for multi-agent systems III: Research issues and practical applications* (LNCS 3390, pp. 144-161). Berlin: Springer-Verlag.

Shen, R., Wang, J., & Zhu, H. (2004). Scenario mechanism in agent-oriented programming. *Proceedings of APSEC'04*, Busan, Korea (pp. 464-471).

Wang, J., Shen, R., & Zhu, H. (2005a, July 25-28). Agent-oriented programming based on SLABS. *Proceedings of COMPSAC'05*, Edinburgh, UK (pp. 127-132).

Wang, J., Shen, R., & Zhu, H. (2005b, September). Caste-centric agent-oriented programming. *Proceedings of the 1st International Workshop on Integration of Software Engineering and Agent Technology at QSIC'05,* Melbourne, Australia.

Wang, J., Shen, R., & Zhu, H. (2005c, July 27-29). Towards an agent-oriented programming language with caste and scenario mechanisms. *Proceedings of AAMAS'05,* Utrecht, The Netherlands.

Wood, M. F., & DeLoach, S. A. (2000). An overview of the multiagent systems engineering methodology. *Proceedings of AOSE 2000* (pp. 207-222).

Wooldridge, M., Jennings, N. & Kinny, D. (2000). The Gaia methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems, 3*(3), 285-312.

Zambonelli, F., Jennings, N., & Wooldridge, M. (2003). Developing multiagent systems: The Gaia methodology. *ACM Transactions on Software Engineering and Methodology, 12*(3), 317-370.

Zambonilli, F., & Omicini, A. (2004). Challenges and research directions in agent-oriented software engineering. *Autonomous Agents and Multi-Agent Systems, 9,* 253-283.

Zhu, H., & Shan, L. (2005). Caste-centric modelling of multi-agent systems: The CAMLE modelling language and automated tools. In S. Beydeda & V. Gruhn (Eds.), *Model-driven software development, research and practice in software engineering II* (pp. 57-89). Berlin: Springer-Verlag.

Zhu, H. (2001). SLABS: A formal specification language for agent-based systems. *International Journal of Software Engineering and Knowledge Engineering, 11*(5), 529-558.

Zhu, H. (2002). A growth process model and its supporting tools for developing Web-based software. *Acta Electronica Sinica, 30*(12A), 2090-2093.

Zhu, H. (2004, September). Cooperative agent approach to quality assurance and testing Web software. *Proceedings of COMPSAC'04 (Workshop Papers and Fast Abstracts), the Workshop on Quality Assurance and Testing of Web-Based Applications (QATWBA'04)*, Hong Kong (pp. 110-113).

Zhu, H. (2005, July). Towards formal reasoning about emergent behaviours of MAS. *Proceedings of SEKE'05*, Taipei (pp. 280-285).

Zhu, H. (2006). Towards an agent-oriented paradigm of information systems. In J.-P. Rennard (Ed.), *Handbook of research on nature inspired-computing for economics and management.* Hershey, PA: Idea Group Reference.

Zhu, H., Greenwood, S., Huo, Q., & Zhang, Y. (2000, July 30). Towards agent-oriented quality management of information systems. *Proceedings of the 2nd International Bi-Conference Workshop on Agent-Oriented Information Systems at AAAI'2000*, Austin, TX (pp. 57-64).

Chapter XLVI
Evolving Learning Ecologies

Jon Dron
University of Brighton, UK

ABSTRACT

This chapter describes the application of self-organising principles to the field of e-learning. It argues that traditional managed approaches to e-learning suffer from deficiencies both in cost and adaptativity that are addressed through the application of nature-inspired processes such as stigmergy and evolution. Such systems, primarily those employing social navigation, are built to generate structure through the dialogue-like interactions of individual learners within them. The result is emergent control of the learning process, adapting dynamically to learner needs, with limited teacher involvement. The chapter describes some example applications and explores some of the remaining challenges in the field, most notably in encouraging pedagogically useful structures to evolve.

INTRODUCTION: THE PROMISE OF E-LEARNING

The capacity to learn is a central and defining characteristic of most organisations, both large and small (Seely Brown & Duguid, 2000; Vaill, 1996; Wenger, 1998). Many formal and informal approaches have been developed to assist the process of learning within an organisation. Until recently, the predominant formal approach has been face-to-face training, be it through classroom or apprenticeships. However, this is changing. Driven largely by cost considerations, the benefits of anytime, any place delivery and a just-in-time approach, computer-based training (CBT), computer-aided learning (CAL), and more recently, Internet-based or e-learning, have become vital tools to help provide skills and disseminate knowledge within organisations. The financial arithmetic is compelling: for example, Kelly and Nanjiani (2005) report that in 2003, for every dollar spent at Cisco on e-learning, there was a $16 return on investment. Such success stories rely on a range of factors, most notably organisational commitment, but it is central to virtually all forms of e-learning that there is some involvement of a teacher or mentor, be it in the production of learning materials or the ongoing support and evaluation of learners. Carefully designed CBT or CAL learning materials suffer from two main disadvantages, inasmuch as their production is:

1. labour intensive, and
2. time consuming.

Apart from the cost, the net result of this is that, in a dynamic and fast-changing environment, such learning resources may not be perfectly adapted to the current state of the business ecosystem nor to the needs of the learners. For these and other reasons, it is therefore common to supplement or replace these carefully designed resources with some form of learning community, whether face-to-face, asynchronous (typically using discussion forums), or synchronous (e.g., through video/audio/text conferencing or through Web meeting software). Again, this can be relatively costly to manage and requires highly skilled and dedicated personnel to operate successfully.

This chapter explores an alternative approach. It describes a range of nature-inspired technologies and processes that enable groups of individuals in learning communities to help each other and help themselves. In particular, it describes how evolutionary and stigmergic mechanisms can generate an emergent structure that is adapted to the needs of specific groups of learners and is, potentially, relatively inexpensive. Teachers, mentors, and subject experts are not excluded from the process. On the contrary, such systems are predicated on the assumption that information networks and the Internet in particular are rich and deep pools of resources for learning. However, such resources are often freely available or very cheap due to economies of scale.

FINDING STUFF

Perhaps the most obvious characteristic of the Internet is that it is huge. In December 2004, Google™ had indexed over 8 billion Web pages (http://www.Google.co.uk/corporate/timeline.html), a number that continues to grow. This vast number only hints at the far greater quantity of information available on the Invisible Web (Lackie, 2003), stored in databases, hidden behind intranets, in protected content management systems, or simply not yet indexed. Beyond the Web, e-mail, file-sharing, Usenet newsgroups, and other forms of electronic communication and information perhaps exceed the amount of data available on the Web. Within this wealth of stored and communicated knowledge, it is hard to imagine that for any conceivable learning need, there is not something that might be of value. For example, I teach networking, one tiny aspect of which is Ethernet. A search on Google (June 2005) for "Ethernet tutorial" reveals around 701,000 potentially valuable pages, illustrating that the Internet is not so much an information superhighway but a 'stuff swamp' (Crawford, 1999). The problem is to identify what might be of greatest value and, equally, of identifying the useless or positively harmful. Google is a form of implicit collaborative filter or recommender system. Its PageRank™ algorithm (Brin & Page, 2000) and others of its ilk go part of the way, making use of latent human annotation (Kleinberg, 1998) to provide some hint that others have found these pages valuable. However, as a teacher I am able to recognise potentially valuable tutorials right down to the final page of Google's results (about 790 pages into the search, as it happens), but equally I can find tutorials that might be far from relevant, timely, or effective for my teaching needs on the first page.

Identifying the quality of information returned by a search is essential. Google uses implicit recommendations provided by hyperlinks in Web pages, making decisions based on this single dimension of value. When seeking high-quality information, there are many other relevant criteria we might use. For example,

Valovic (1994) suggests that we should consider time value, scope, authenticity, and dissemination value, while Hofman and Worsfield (1996) identify five main categories and 23 subcategories of criteria. Various educational systems have been proposed or developed that explicitly use collaborative filtering techniques (Anderson et al., 2003; Chislenko, 1997; Recker, Walker, & Wiley, 2000; Terveen, Hill, Amento, McDonald, & Creter, 1997), but most assume that resources can be explicitly or implicitly rated on a single scale, like Google. This is not enough. In e-learning, even Hofman and Worsfield's extensive criteria only scratch the surface of learner needs. This is not just a question of content in the correct subject area and of suitable quality. Different learners have different abilities, different needs, different starting points, different learning styles, different preferences. To make matters worse, learners do not stand still. By its nature, learning is a process of transformation: as we learn, we change and, if our learning was successful, what was useful before is unlikely to be useful now. A teacher typically assembles resources for learners taking a wide range of issues into account, including learning objectives, narrative, perceived learning needs, projected outcomes, and many other tangible and intangible factors. In an attempt to capture this, a number of standards for metadata relating to learning objects, learners, and teachers have been developed, including those of the IMS, AICC, SCORM, Ariadne, and IEEE LOM, as well as other less well-defined standards such as EduRSS and EML. Although the standards are slowly converging, the wide range of overlapping and often competing standards attests to the wicked nature of the problem. While in principle it might be possible to cater for most conceivable needs, the greater the range of metadata available, the more unwieldy they become, and before long, a similar difficulty to that of finding relevant information in the first place arises¾there is simply too much of it, and incentives to provide it are seldom compelling. Worse still, even where there is agreement on relevant metadata, there are differences of interpretation as to how they should be applied. For example, the word "beginner" may have many different shades of meaning depending on the context in which it is used and the group or groups of learners to which it applies. A beginner in Web development with no experience of computers is quite different from one with a background in programming. The knowledge of a teacher, much of it tacit, is not easily embodied in a machine. If machine intelligence is not the answer, then one approach to solving the problem is to make use of the intelligence of learners themselves, to which we turn in the next section.

SOCIAL SOFTWARE

In recent years developments such as blogs and wikis, as well as popular sites such as furl (http://www.furl.net/index.jsp), del.icio.us (http://del.icio.us/), Shadows™ (http://www.shadows.com/), and flickr™ (http://www.flickr.com) have attracted attention as instances of social software, perhaps defined most succinctly by Shirky (2003) as software that embodies and alters social patterns. Shirky observes that a common characteristic of such software is that the group is a first-class object within the system. In other words, the results of the collective behaviour of the group is different in kind from the individual actions of its members. Control and structure are emergent properties. For example, furl, Shadows, and del.icio.us are, from an individual's perspective, useful personal repositories for bookmarks. However, as part of the process of bookmarking, sites are 'tagged' with arbitrary metadata, classifica-

tions that suit the individuals' needs when organising their collections of bookmarks. These collections of bookmarks may be private, but more often they are sharable with others. Similar tags are used again and again, with overlapping content. By aggregating the links classified with these tags, users are able to discover other relevant sites that have been similarly classified. An emergent structure or "folksonomy" forms, based not on the top-down decisions of professional taxonomists but on the direct perceived needs of the users themselves. Because the taxonomies of others influence and inspire the taxonomies of individuals as much as vice versa, they collectively generate a kind of information ecosystem, with feedback loops interacting to generate a complex dynamic system. Del.icio.us, for instance, shows the most popular links, with the option to add those links to your own bookmarks, a self-reinforcing process that rewards success and leads to clusters of shared bookmarks. It is possible to view the bookmarks of others who have bookmarked the same links, encouraging the discovery of related links and people with shared interests, a process further amplified by features such as e-mail and discussion forums, made more useful by the option to bundle bookmarks together. Some systems (described below) have been developed to take advantage of this sort of feature within an educational setting.

Social Navigation

Social software that makes use of the past or present behaviour of others to influence or determine the paths that current users take employs a process known as social navigation. When such systems embody the past actions of others, they are sometimes described as *history-rich,* while systems that provide some indication of the real-time presence of others are sometimes known as *aware-ware.* History-rich systems may be compared with traditional books, which tend to fall open on well-read pages, or be annotated, or have page corners turned down. The simplest examples are hit counters that indicate the number of unique visitors to a page. Aware-ware systems provide an experience more akin to the experience of being in a crowded street or building, where the movements of others may influence us positively or negatively to follow or avoid the areas where they congregate. The simplest examples are Web sites that provide an indication of the number of current visitors.

Social navigation is inherently stigmergic, whether direct (sematectonic) or indirect (sign-based). The behaviour of others necessarily leaves signs in the environment which influence others. This makes them particularly interesting from an educational perspective. Stigmergy is an exception to the general rule of the natural world that the large and slow-moving parts of a complex system (e.g., an ecosystem) play a larger role in influencing the behaviour of the small and fast moving (Brand, 1997). As Churchill put it, "We shape our dwellings and afterwards our dwellings shape our lives." A similar dynamic underlies Senge's (1993) observation that structure influences behaviour. Stigmergy causes structure to arise from behaviour, as much as vice versa. In a sense, structure arises from communication which might be described, loosely, as a form of dialogue. This is a significant observation if stigmergic principles are applied to educational systems, especially those where learning occurs in situations where the learner and teacher are physically separated.

The distance learning theorist Michael Moore (Moore, 1983; Moore & Kearsley, 1996) has shown that there is an inverse relationship between structure (the control of a teacher) and dialogue (control negotiated between

learner and teacher). The more there is of one, the less there will be of the other. In an educational transaction where the learning activities are entirely determined by the teacher, options for dialogue will be limited, whereas it is inevitable that dialogue will lead to behaviour that cannot be exactly determined in advance. As well as being intuitively compelling, this has been experimentally confirmed by Saba and Shearer (1994). An emergent system that derives its form from a circular relationship between structure and dialogue of the sort found in stigmergy might therefore be a promising means of constructing a self-organising learning environment. It appears to provide both structure and dialogue, if not at the same time then certainly in the same environment. While the mediated dialogue of stigmergy is impoverished when compared with the richness of even an SMS message, it does contain the essential elements of dialogue. Bakhtin says that words are a "territory shared by both addresser and addressee, by the speaker and his interlocutor" (Morris, 1994). In a meaningful sense, stigmergic signs provide a means of (quite literally) establishing that territory.

EXAMPLES

The following examples are selected to help illustrate a few of the ways in which social navigation is used to create e-learning environments and to provide a context for the discussion that follows. Each of these systems generates a kind of structure through the use of stigmergic processes, bringing together existing resources in ways that may be valuable for learners. In each case this is a participatory process of construction, which may be of educational value in itself. These are merely examples of the genre that serve to illustrate the principles involved. There are many others (e.g., Anderson et al., 2003; Donath, Karahalios, & Vigas, 1999; Edmons, 2000; Miettinen, Kurhila, Nokelainen, & Tirri, 2005; Semet, Lutton, & Collet, 2003; Small, 2001; Susi & Ziemke, 2001; Yang, Han, Shen, Kraemer, & Fan, 2003).

CoFIND

CoFIND is a collaborative bookmark database predating furl and del.icio.us, but embodying many of the same concepts. Like its successors, it allows users to add bookmarks and to tag them with shared, arbitrarily named tags. However, CoFIND is solely designed for groups, not individuals. Although it preferentially displays a user's own bookmarks, all bookmarks are shared and their authors are anonymous to others in the system. CoFIND makes extensive use of many social navigation cues to help learners explore the system. It uses both implicit and explicit ratings to structure the environment, which is constantly evolving, in a literal and Darwinian sense.

As well as simple categorisations (topics and groups of topics), CoFIND uniquely employs metadata known as *qualities* to guide learners to resources. Like everything in the system, these are entered by the users, so their precise nature varies from one instance to the next. Typically they include words such as "useful," "good for beginners," "clear," or "detailed." They are used to provide explicit ratings for resources. They are words or phrases that indicate not only *that* a resource is valuable, but also *how* it is valuable. As there are no limits on the kinds of qualities that may be entered, it is possible for users to enter negative qualities, but such usage is relatively rare as qualities are usually employed to recommend resources.

New qualities can inherit ratings from old ones, so that ratings for existing resources are

copied to the new quality. Once the ratings have been copied, each new quality leads an independent existence from its parent. This is a direct analogue of reproduction with variation, albeit in a Lamarckian more than a Darwinian sense.

Users may select the qualities that are valuable to them at the time they are needed, rather than the more traditional route of embedding such preferences within a user model (Brusilovsky, 2001). Qualities thus provide a kind of disembodied user model, discardable attributes that capture what is valuable at a given moment in an educational trajectory, leaving a kind of pedagogic trail for those who come after.

In a direct analogue of Darwinian natural selection, resources and metadata (topics, groups of topics, and qualities) compete for prominence within the system. Success for both resources and metadata is measured primarily using list position and font size, determined by implicit usage statistics and explicit ratings. The evolutionary winners are those that are most prominent. The losers dwindle at the periphery of the system and are seldom used. In earlier versions they were actually 'killed,' but this was found to be pedagogically harmful, reducing the incentive of users to contribute when their metadata or resources vanished altogether. In the current system, an extra weighting is given to a user's own data and metadata, meaning they will always see their own resources. Numerous other social navigation cues such as novelty icons, flags of recent visits, displays of recent changes, and numeric counters combine to influence behaviour and thus structure the environment dynamically. Each combination of topic and quality represents a small, connected ecosystem in which resources struggle for survival, boosted or thwarted by stigmergic and other navigation cues. In some ecosystems they flourish; in others they fail.

Knowledge Sea

The Knowledge Sea combines AI techniques with social navigation, representing clusters of resources in varying shades of blue that indicate their relative popularity, based on implicit usage statistics (Brusilovsky, Chavan, & Farzan, 2004). Similar resources are clustered, allowing users to identify those that are most relevant by subject, then to identify those that are most relevant by popularity. The stigmergic effects of this are to create deep pools of deeper blue around current topics of interest, drawing users into them. There is a danger with such a system that there might be insufficient negative feedback to draw users away from the area of current interest, but this is compensated for in the way that the system is used to support a traditional institutionally taught course, which provides a further layer of structure forcing users to periodically change topic. As with for all such self-organising environments, context is a crucial element and it is not meaningful to separate the computer system from the other systems with which it interacts.

EDUCO

EDUCO uses both history-rich and real-time aware-ware social navigation features (Kurhila, Miettinen, Nokelainen, & Tirri, 2002). Historically popular documents receive greater emphasis, primarily through shading. Users are shown as dots clustered around documents, making others more inclined to join them in real time. To increase their attractiveness, EDUCO provides a chat system, allowing each chat to be associated with specific documents. This combination of both implicit and explicit dialogue, sign-based and sematectonic stigmergy, creates structure through dialogue in a constantly changing information environment, adapting rapidly according to the changing interactions of its users. The combination of

different social navigation cues that act at different temporal scales, the immediate and the historic, allow for the possibility of a richer kind of structure. This provides (largely latent) potential for the development of an ecosystem that more closely resembles that of natural systems, where the large and slow-moving provide the context for the smaller and faster elements.

Jasper

Jasper (Crossley, Davies, McGrath, & Rejman-Greene, 1999) uses a direct metaphor of a cultivated garden rather than a natural ecosystem, presented using a full three-dimensional interface. Flowers in the garden are clustered together according to similarity of subject matter and move when "touched" (i.e., a user has looked at the chunk of knowledge associated with that flower). Flowers that are most used grow strongest and tallest, while those that are least used wither and die, or may be pruned. Inhabitants of the garden may take cuttings of flowers for their own gardens, and there is an iconic representation of their levels of activity within the system which helps to identify those who are most worth following. Using both sematectonic and sign-based stigmergic cues, Jasper is a rich system that combines both user-created and emergent structure at a number of scales, both temporal and physical. Sadly it appears to no longer be under active development.

THE WISDOM OF CROWDS, THE STUPIDITY OF MOBS

The benefits of systems such as those described above include a cost-effective means of utilising the collective knowledge of the many to the benefit of all, a low-threshold approach to collaboration, an increased sense of community, and a release from some of the isolation of e-learning. Social navigation offers many potential benefits, but with it come potential hazards. In this section we will explore some of these hazards and identify nature-inspired approaches to overcoming them.

A potentially damning criticism of a self-organised approach to learning that uses social navigation is that it may lead to the blind leading the blind. There are at least two distinct potential, though partial, answers to this objection.

1. While any individual learner may have insufficient knowledge to adopt a self-organised learning strategy, the fact that a learning community is made up of adults with a rich variety of perspectives means that the sum of knowledge available is greater than that of an individual. Each learner may contribute the answer to part of the puzzle (e.g., by contributing or rating resources, or simply by leaving a trail of footprints), but it is the emergent collective that generates the structure of the whole.
2. The systems are part of much wider systems and are predicated on the assumption that there are already suitable learning resources available and an existing or actively developing learning community. The systems' contribution is in finding and effectively using those resources that are most relevant.

Although these answers go part of the way to dealing with the problem, a number of thorny issues remain. Examples such as bank runs show clearly how the wisdom of crowds can, through stigmergy, run out of control and become the stupidity of mobs. Without sufficient checks, there is a fine line between a self-organising stigmergic system and one that op-

erates according to the Matthew Principle (most eloquently expressed by Billie Holliday as "them that's got shall get, them that's not shall lose"). Surowiecki (2004) writes of rioting mobs: "As more people riot, more people decide that they are willing to riot, too." For Surowiecki, the answer to the problem lies in maintaining a balance between information that is shared and information that is private. Shared information can strongly influence the behaviour of those who have access to it. As more people share that information, its influence is reinforced. One approach, typified by the Knowledge Sea, is thus to ensure that those with sufficient knowledge and authority provide the most influence on the system, but this reintroduces the objections to traditional e-learning voiced earlier, albeit in a less resource-intensive and more adaptive manner. Another alternative is to aggregate privately held information, withholding the results until sufficient information has been accrued. Google's latent human annotation works this way, due to the innate delays between updates to its indexes. However, there are other approaches that may be borrowed from nature that may help.

Natural Selection

Evolutionary processes of natural selection can provide sufficient negative feedback to halt out-of-control processes. CoFIND, for instance, allows negative as well as positive ratings to provide a limit on harmful growth. Because users are intelligent and goal directed, structures that get in the way of their goals can be removed. For example, when a student deliberately manipulated the system to encourage others to visit his own (tangentially useful at best) site, the negative ratings of others eventually pushed it off the map (Dron, Mitchell, & Boyne, 2003).

Purpose and Clustering

The example of the manipulative student referred to in the previous paragraph highlights a significant aspect of self-organising learning environments. If learners use them with the intention of learning, then they become learning environments. If not, they become something else. For this reason, they work best with relatively small groups of similarly minded learners, with clear and shared learning goals. As a corollary, where a system has evolved to suit the needs of a particular group, it may not suit a newcomer to it quite as well and, for better or worse, the newcomer may provide a disruptive influence by interacting with the system differently.

If used in a larger organisation, it is important to design systems that have a small-world structure, where small, weakly connected clusters can evolve relatively independently of each other. For instance, this can be provided by the parcellation caused by tagging in environments such as CoFIND or the clusters of the Knowledge Sea. These mechanisms can also help to limit damaging effects to a small area, much as parcellation affords opportunities for isolated pockets of evolution in natural systems (Calvin, 1997). Equally, the weak connections between these isolated pockets can act like isthmuses or shifting land masses in the natural evolutionary environment, spreading useful structures into other nearby ecosystems. The principle of tagging as opposed to more traditional hierarchical structures allows resources to be shared between tags, and tags to be shared between clusters of resources. Another way this can be achieved is through the use of multiple scales, as seen in Jasper and EDUCO.

Creativity and Adaptation

Termites will never build mock-Tudor cottages, and stigmergy does not naturally lead to original

solutions. This is independent of the intelligence of the agents involved. The stigmergic structure of money markets exhibits no more intelligence and arguably less than the formation of termite mounds. Having said that, the kinds of stigmergy that drive the systems described here are subtly different from those underpinning natural and simple human systems. For termites building termite mounds, the signs provided by pheromones are unambiguous. Even in human systems such as money markets, where there may be more choices in how to respond (to invest or not to invest), the increases and decreases in the value of currencies are largely unequivocal signals. Social navigation cues, especially when used in combination, are usually far less straightforward. For example, without prior knowledge it is not clear whether the different shades of blue used in the Knowledge Sea imply positive or negative ratings. The presence of others in Educo does not always act as an incentive to visit a page. Users who do not always click on the top element of a list nor the largest item on the screen, may consider different parts of the screen as more significant than others and are affected by the historical, visual, and conceptual context of what they are viewing (Dron, 2005). Above all, by far the most important signal that is used in such systems is the labelling and accompanying text (if any) of the resources themselves. While diluting the overall effectiveness of the social navigation system as a whole, such ambiguities and differences in interpretation can provide the variation that is necessary for the system as a whole to evolve. Google itself is a stigmergic system (Gregorio, 2003) with a strong tendency towards the Matthew Principle, but this does not stop new items from rising to the top of its rankings, partly because of the greater importance of semantic information, partly because it is far from the only source of information available, partly due to the delays between implicit recommendations and their aggregation, and partly because there is always a relatively small but significant number of people choosing to select items that are not in the first page of results.

Sometimes the ambiguities caused by the interactions of different social navigation cues may take on the character of what Gould and Lewontin (1979) describe as 'exaptions'. These features, which are side effects of other adaptations, can lead to unexpected behaviour. For example, in an experiment undertaken by the author comparing the effects of different navigation cues on behaviour, both font size and list position were consistently shown to positively or negatively influence behaviour. However, when combined, the results were less intuitively obvious. For instance, a small item at the bottom of a list of large items attracted a disproportionately large number of clicks, despite combining the two least attractive features of both font size and list position. It was then hypothesised that users were attracted by the fact that it was different from the rest. However, an otherwise identical example in which the item with the smaller font appeared in the middle of the list did not show this behaviour at all (Dron, 2005).

Complex Dynamics, Signposts, and Fenceposts

To provide a learning environment where evolution can occur, it is necessary to develop systems that are capable of co-evolving at the edge of chaos (Kauffman, 1995). If the structure is too prescriptive, they will fall into an unchanging Stalinist regime, while if it is too dynamic, they will fall into a Red Queen regime, always running to stay in the same place, their components never reaching high levels of fitness. In design terms, this means that the structure that is presented to the learners should

not be too constraining, nor should it provide so many choices that the learner is bewildered and unable to choose. Similarly it should not change so rapidly that the learner becomes lost. Learning is by definition a move into the unknown and may thus be frightening (Rogers, 1969), so a learning environment should not make it more so. The central rule should be to build signposts, not fenceposts. Learners should receive guidance, but be free to stray from the path. If appropriate feedback loops can capture and amplify that behaviour, then the system will be capable of change and adaptation.

The Rules of Change

In an ideal system the rules themselves would be subject to evolutionary mechanisms of change. Unfortunately, in a learning environment, this is hard to achieve. Poor adaptations not only fail within the system, but as part of a broader ecology in which other competing alternatives are available, the system itself will not be selected as a suitable environment for learning. This extreme variation of the cold start problem would never get off the ground in the first place. In systems like CoFIND the nearest equivalent to adapting the rules of behaviour lies in the rapid application development method that is used, whereby the system designer chooses ever more effective algorithms based on feedback and use of the system.

CONCLUSION

E-learning software that explicitly employs social navigation is still a research backwater. However, general purpose social software that employs a subset of its principles (blogs, link-sharing software, wikis, and so on) is becoming commonplace. Increasingly the bottom-up design and emergent structures that such software supports develop according to principles more closely akin to evolution than to design. Even the venerable hierarchical asynchronous discussion forum contains social navigation cues such as depth of nesting, implicit indicators of levels of activity, and posting dates that indicate currency. Although these cues are only barely structural and are not designed with educational purposes in mind, they can nonetheless be influential and stigmergic, encouraging or discouraging use. As such systems are ubiquitous tools in e-learning environments, it is important that research continues in this area.

There are still unexplored avenues where principles such as stigmergy and evolution may fruitfully be applied. A particularly wicked problem is that of adaptive sequencing of an open corpus (Masthoff, 2004). The current generation of tools provides a structure that recommends particular resources clustered by similarity and/or popularity. Any sequence that arises is an emergent phenomenon that may be visible in retrospect (though the systems presented here do not explicitly provide this information), but cannot be predicted. However, within traditional planned learning, part of the reassurance and much of the effectiveness is often the result of a planned narrative. While it is simple enough to capture and reinforce learners' paths through a body of resources, ensuring that those paths are useful and worth repeating is a harder task.

As systems such as CoFIND, the Knowledge Sea, and Educo reach maturity, it becomes increasingly important to consider ways of allowing them to interoperate, share knowledge and users, and perhaps even to compete. Tentative steps are being made in this area (Brusilovsky, 2004) which offer a range of new and fascinating challenges, with the promise of opportunities for a rich and variegated worldwide ecosystem of self-organising communities for learning.

REFERENCES

Anderson, M., Ball, M., Boley, H., Greene, S., Howse, N., Lemire, D., & McGrath, S. (2003). RACOFI: A rule-applying collaborative filtering system. *Proceedings of COLA'03,* Halifax, Canada (pp. 13-24).

Brand, S. (1997). *How buildings learn.* London: Phoenix Illustrated.

Brin, S., & Page, L. (2000). *The anatomy of a large-scale hypertextual Web search engine*. Retrieved from http://www-db.stanford.edu/pub/papers/Google.pdf

Brusilovsky, P. (2001). Adaptive hypermedia. *User Modelling and User-Adapted Interaction, 11,* 87-110.

Brusilovsky, P. (2004, May 17-22). KnowledgeTree: A distributed architecture for adaptive e-learning. *Proceedings of WWW 2004,* New York (pp. 104-113).

Brusilovsky, P., Chavan, G., & Farzan, R. (2004). Social adaptive navigation support for open corpus electronic textbooks. *Proceedings of AH 2004,* Eindhoven (pp. 24-33).

Calvin, W. H. (1997). The six essentials? Minimal requirements for the Darwinian bootstrapping of quality. *Journal of Memetics, 1*. Retrieved June 26, 2006, from http://www.fbm.mmu.ac.uk/jom-emit/1997/vol1/calvin_wh.html.

Chislenko, A. (1997). *Collaborative information filtering and semantic transports.* Retrieved November 19, 1999, from http://www.lucifer.com/~sasha/articles/ACF.html

Crawford, W. (1999). The card catalog and other digital controversies. *American Libraries, 30*(1), 52-52.

Crossley, M., Davies, J., McGrath, A., & Rejman-Greene, M. (1999). The knowledge garden. *BT Technology Journal, 17*(1), 76-84.

Donath, J., Karahalios, K., & Vigas, F. (1999). Visualizing conversation. *Journal of Computer Mediated Communication, 4*(4). Retrieved June 26, 2006, from http://www.ascusc.org/jcmc/vol4/issue4/donath.html

Dron, J. (2005). Discovering the complex effects of navigation cues in an e-learning environment. *Proceedings of E-Learn 2005,* Vancouver (pp. 2026-2033).

Dron, J., Mitchell, R., & Boyne, C. W. (2003). Evolving learning in the stuff swamp. In N. Patel (Ed.), *Adaptive evolutionary information systems* (pp. 211-228). Hershey, PA: Idea Group Publishing.

Edmons, B. (2000). *Supporting collective intelligence on the Web: Design, implementation and test of a self-organising, collaborative knowledge system.* Retrieved August 5, 2000, from http://www.cpm.mmu.ac.uk/~bruce/bsi/FET-project.html

Gould, S. J., & Lewontin, R. C. (1979). The Spandrels of San Marco and the Panglossian paradigm: A critique of the adaptationist paradigm. *Proceedings of the Royal Society of London, Series B, 205*(1161), 581-598.

Gregorio, J. (2003). *Stigmergy and the World Wide Web.* Retrieved December 13, 2003, from http://bitworking.org/news/Stigmergy/

Hofman, P., & Worsfield, E. (1996). *Specification for resource description methods, part 2: Selection criteria for quality controlled information gateways.* Retrieved August 1, 2001, from http://www.ukoln.ac.uk/metadata/DESIRE/quality/report.rtf

Kauffman, S. (1995). *At home in the universe: The search for laws of complexity.* London: OUP.

Kelly, T., & Nanjiani, N. (2005). *The business case for e-learning.* Indianapolis: Cisco Press.

Kleinberg, J. M. (1998). Authoritative sources in a hyperlinked environment. *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms* (pp. 668-677).

Kurhila, J., Miettinen, M., Nokelainen, P., & Tirri, H. (2002). Use of social navigation features in collaborative e-learning. *Proceedings of E-Learn 2002,* Montreal, Canada (pp. 1738-1741).

Lackie, R. J. (2003). The evolving 'invisible Web': Tried-and-true methods and new developments for locating the Web's hidden content. *College & Undergraduate Libraries, 10*(2), 65-71.

Masthoff, J. (2004). Group modelling: Selecting a sequence of television items to suit a group of viewers. *User Modelling and User Adapted Interaction, 14,* 37-85.

Miettinen, M., Kurhila, J., Nokelainen, P., & Tirri, H. (2005). OurWeb—Transparent groupware for online communities. *Proceedings of the Conference on Web-Based Communities 2005,* Algarve, Portugal (pp. 53-61).

Moore, M. G. (1983). On a theory of independent study. In D. Sewart, D. Keegan, & B. Holmberg (Eds.), *Distance education: International perspectives* (pp. 68-94). London; Canberra: Croom Helm.

Moore, M. G., & Kearsley, G. (1996). *Distance education: A systems view.* Belmont: Wadsworth.

Morris, P. (Ed.). (1994). *The Bakhtin reader.* London: Edward Arnold.

Recker, M. M., Walker, A., & Wiley, D. (2000). An interface for collaborative filtering of educational resources. *Proceedings of the 2000 International Conference on Artificial Intelligence,* Las Vegas, NV. Retrieved June 26-2006, from http://www.cs.wright.edu/~mcox/ic-ai/recker.doc

Rogers, C. (1969). *Freedom to learn.* Columbus, OH: Merrill.

Saba, F., & Shearer, R.L. (1994). Verifying key theoretical concepts in a dynamic model of distance education. *The American Journal of Distance Education, 8*(1), 36-59.

Seely Brown, J., & Duguid, P. (2000). *The social life of information.* Boston: Harvard Business School Press.

Semet, Y., Lutton, E., & Collet, P. (2003). Ant colony optimisation for e-learning: Observing the emergence of pedagogic suggestions. *Proceedings of IEEE SIS 2003,* Indianapolis, IN (pp. 46-52).

Senge, P. M. (1993). *The fifth discipline—The art and practice of the learning organisation.* Chatham: Century Business.

Shirky, C. (2003). *A group is its own worst enemy.* Retrieved June 29, 2005, from http://www.shirky.com/writings/group_enemy.html

Small, P. (2001). *A self-organizing, living database for volatile data*. Retrieved from http://www.stigmergicsystems.com/stig_v1/papers/page1.html

Surowiecki, J. (2004). *The wisdom of crowds.* London: Little, Brown.

Susi, T., & Ziemke, T. (2001). Social cognition, artefacts, and stigmergy: A comparative analysis of theoretical frameworks for the understanding of artefact-mediated collaborative activity. *Cognitive Systems Research, 2*(4), 273-290.

Terveen, L., Hill, W., Amento, B., McDonald, D., & Creter, J. (1997). PHOAKS: A system

for sharing recommendations. *Communications of the ACM, 40*(3), 59-62.

Vaill, P. (1996). *Learning as a way of being: Strategies for survival in a world of permanent white water.* San Francisco: Jossey-Bass.

Valovic, T. (1994). Quality of information—Part 2. *Telecommunications Magazine.* Retrieved June 26, 2006, from http://www.telecoms-mag.com/marketing/articles/feb95/usa_editorial.html

Wenger, E. (1998). *Communities of practice: Learning, meaning and identity.* New York: Cambridge University Press.

Yang, F., Han, P., Shen, R., Kraemer, B. J., & Fan, X. (2003). Cooperative learning in self-organising e-learner communities based on a multi-agents mechanism. *Proceedings of the 16th Australian Conference on AI,* Perth, Australia (pp. 490-500).

Chapter XLVII
Efficient Searching in Peer-to-Peer Networks Using Agent-Enabled Ant Algorithms

Prithviraj Dasgupta
University of Nebraska, USA

ABSTRACT

In this chapter we describe a mechanism to search for resources in unstructured peer-to- peer (P2P) networks using ant algorithms implemented through software agents. Traditional resource search algorithms in P2P networks use an uninformed or blind search among the various nodes of the network. In contrast, the resource search algorithm described in this chapter performs an informed search using the ant-based heuristic. In our algorithm, ants, implemented as software agents, are created in response to a user's resource search query. An ant reinforces the route that yields a successful search for directing ants in the future towards nodes with higher probability of locating resources. We describe and compare different reinforcement strategies used by ants to perform efficient resource search in P2P networks.

INTRODUCTION

The advent of the Internet over the past decade has enabled humans to interact and exchange information with each other in various formats including text-based communication, audio, and video. Recently, file-sharing networks such as Napster, Kazaa, and BitTorrent have become an attractive paradigm for online users to exchange resources such as data, information, and services with one another. Most of these file-sharing networks are based on a peer-to-peer (P2P) network architecture. Besides file sharing, P2P systems are currently being used in applications including data sharing and information management for digital libraries (Walkerdine & Rayson, 2004), future combat systems, and large-scale computing for searching for extra-terrestrial life (Anderson, Cobb, Korpela, Lebofsky, & Werthimer, 2002). With

the recent research and commercial interest in P2P networks, it is clear that P2P networks are becoming an important technology for managing large-scale information systems. Therefore, it makes sense to develop mechanisms that make management of P2P networks more efficient. In this chapter, we focus on techniques to improve an essential feature of P2P networks—searching for resources within P2P networks. We envisage that understanding and addressing the issues related to P2P search algorithms will enable us to improve information management in P2P networks.

A P2P network consists of users located on nodes that are interconnected with each other. In the client-server paradigm used commonly over the Internet, a client computer requests access to information or services that are available with a server computer, giving rise to a master/slave-like hierarchy between the server and the client. In contrast, in a P2P network, all nodes behave as peers with similar or comparable capabilities. Every peer node can provide as well as access information and services from other peer nodes. The peer-based model makes P2P networks decentralized and distributed. Consequently, in contrast to a centralized server-based architecture, P2P networks can grow in size without worrying about problems such as congestion at the central server node and scalability of the network. However, the decentralized nature of P2P networks also introduces several challenges such as rapid searching of resources in the network, issues related to security and trust of nodes in network, and enforcing fair sharing of resources among nodes in the P2P network. Several mechanisms including structured overlays (Ratnaswamy et al., 2001), reputation and referral-based mechanisms (Yu, 2003), and game-theoretic approaches (Marti & Garcia-Molina, 2004) have been proposed to address these challenges. In contrast to these approaches, in this chapter we discuss nature-inspired computing techniques to address the problem of rapidly searching for resources in a P2P network.

P2P NETWORK CLASSIFICATIONS

In a P2P network, a user at a node makes different resources such as text, audio, and video files, and even computational resources such as CPU time and networked storage, available to other users. A user on another node can access and acquire these resources, and in return is expected to share resources that it possesses. Consequently, one of the major operations performed by users in a P2P network is to search for resources on other nodes of the network. Searching for resources rapidly and efficiently is a challenging problem in P2P networks because of the absence of a central server node that would maintain information about the contents shared by all nodes in the network. The decentralized nature of P2P networks also introduces challenges such as load balancing, authenticating the identity of nodes, and monitoring the actions performed by different nodes.

Various types of P2P networks that modify some of the features of the peer-based model have been proposed to address some of these challenges. One class of P2P networks is the structured P2P networks where nodes are arranged on a pre-determined topology such as a hypercube or a ring. Distributed hash table (DHT)-based techniques are used to facilitate rapid searching of resources in the network. Several structured P2P networks such as Chord (Stoica, Morris, Karger, Kaashoek, & Balakrishnan, 2001), Pastry (Rowstron & Druschel, 2001), Tapestry (Zhao, Kubiatowicz, & Joseph, 2001), Content Addressable Net-

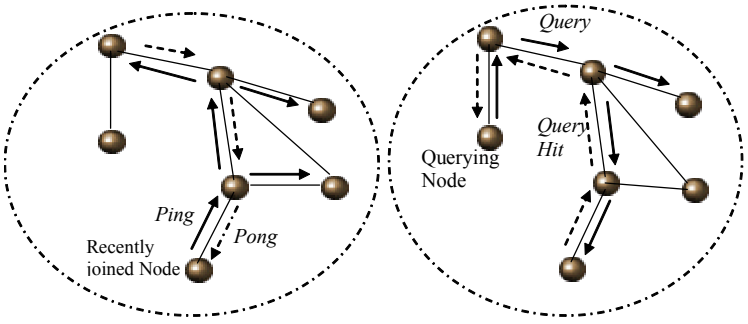
work (CAN) (Ratnaswamy et al., 2001), and Symphony (Manku, Naor, & Manber, 2004) have been developed that employ clever network topologies and hashing algorithms. However, in a structured P2P network, node and resource placement algorithms determine the location of a node and the resources stored on that node. A user at a node in a structured P2P network does not have the discretion to select resources for storing and sharing from the node at which the user is located. In contrast, unstructured P2P networks do not use deterministic node and resource placement algorithms, and a user in an unstructured P2P network can decide which resources he/she is willing to host and share. Unstructured P2P networks are also relatively easy to setup, and their protocols are simple to implement. Consequently, unstructured P2P networks have become an attractive paradigm for implementing commercial P2P systems such as Kazaa (http://www.kazaa.com), Limewire (http://www.limewire.com), and Morpheus (http://www.morpheus.com). In unstructured P2P networks, network setup and resource search rely on a packet flooding mechanism. This gives rise to considerable traffic, ensuing congestion within the network that degrades the network performance. The network protocols for unstructured P2P networks are described in the following section.

UNSTRUCTURED P2P NETWORK PROTOCOLS

Unstructured P2P networks are by far the most attractive P2P network architecture for commercially deployed P2P systems owing to their simplicity. In this section we provide an overview of the unstructured P2P network protocols used for network setup and resource search.

In an unstructured P2P network, a newly joining node uses the node discovery protocol to discover other nodes in the network. As shown in Figure 1, a new node wishing to join the network sends a *ping* message that gets flooded across the network. Nodes receiving the ping message respond with a *pong* message that contains information about the responding node including its IP address, available bandwidth, and networking resources. The pong message is routed back to the joining node that originated the ping message. The joining node then selects its neighboring nodes based on the information contained in the different pong messages.

Figure 1. (a) Node discovery and (b) resource discovery protocols in a P2P network



(a) *(b)*

After a node has joined the P2P network, it uses the resource discovery protocol to search for resources within the network. The resource discovery protocol comprises a *query* message that gets forwarded to nodes along the network paths established during node discovery, until the resource is discovered or the lifetime of the query message expires. If the resource is found on a node along the search path of the query message; a *queryHit* message is sent back to the node originating the query. The node originating the query then selects one of the nodes that responded with a queryHit message and downloads the resource from that node. In this chapter, we concentrate on the P2P resource discovery protocol and assume that appropriate mechanisms for node discovery are used to set up the P2P network. Lin, Marzullo, and Masini (1999) and Portmann and Sevairatne (2002) provide techniques to improve the basic node discovery protocol for unstructured P2P networks.

INFORMED SEARCH IN A P2P NETWORK

The *resource discovery* protocol in an unstructured P2P network performs an uninformed or blind search to locate the resource being searched for. Instead of using a blind search over all the nodes in the network, the efficiency of the search can be improved by using a heuristic that directs the search towards nodes that are more likely to contain the resource being searched for. Here, we describe a heuristic inspired by the foraging activity of social insects such as ants (Bonabeau, Dorigo, & Theraulaz, 1999; Heck & Ghosh, 2002) to locate food.

In several ant species, ants searching for food leave behind a pheromone trail along the path from their nest to the food. Ants searching for the food later on use the trail as a positive reinforcement to lead themselves to the food. Ants also show a high affinity to an established trail and are reluctant to deviate to trail-less but more efficient paths towards the food even if one exists.

In our resource discovery algorithm, software agents (Weiss, 1998) implement the actions of ants. In the rest of the chapter, the terms *agent* and *ant* are used interchangeably. To enable our resource discovery algorithm, an ant is created in response to a search query. Every node within the network is associated with a pheromone value that represents the probability of locating the resource being searched for on that node. An ant searches for a resource on different nodes of the network. Ants get attracted to nodes with higher pheromone values. If an ant locates the resource it is searching for on a node, the pheromone value of that node is reinforced. The ant then retraces the route back to its origin while reinforcing pheromone on each node along its return route. On the other hand, if the resource is not found by an ant, it does not reinforce pheromone along the nodes of its return route. Later on, when another resource needs to be searched, an ant uses an already established pheromone trail to direct the search towards nodes where resources had been located previously.

ANT ALGORITHM FOR P2P SEARCH

We consider an unstructured P2P network. Each node in the network contains certain resources inside a resource table, and a resource can be identified on a node with a unique resource name. Nodes join and leave the network at random using the P2P node discovery protocol. Each node maintains a forwarding table containing the addresses of its neighbor

nodes returned by the node discovery protocol. Each address in the forwarding table is associated with a normalized value that corresponds to the pheromone associated with that node. The pheromone associated with a node in the forwarding table gets updated when an ant selects to move to it.

A user at a node initiates a search by providing a set of keywords corresponding to the identifier of the resource(s) he or she wishes to locate. The node originating the query creates an ant to search for the resource. At each node, the ant selects a neighboring node to move to for the next hop, with a probability given by the pheromone corresponding to the node within the forwarding table of the current node. Before migrating to the selected node, the ant updates the pheromone corresponding to that node in the forwarding table. The pseudocode for the ant algorithm for resource discovery is given in Figure 2.

Pheromone Update

The updatePheromone method in Figure 2 is used by an ant to update the pheromone of the node it selects to visit next, from the forwarding table of the node it is currently visiting. Every ant is provided with a pheromone value π_o when it starts its search for a resource from its origin node *o*. While visiting node *i*, an ant selects the node *j* that has the highest pheromone value within node *i*'s forwarding table and updates the pheromone $p_{j,i}$ according to equation 1 given below:

$$\pi_i \leftarrow \pi_o / [h_{d,i}]^{\alpha},$$

$$p_{j,i}^{\ t} \leftarrow p_{j,i}^{\ t-1} + \pi_i \ (1 - p_{j,i}^{\ t-1}), \qquad (1)$$

where, π_i denotes the pheromone deposited by the ant within node *i*'s forwarding table, $h_{d,i}$ denotes the number of hops made by the ant to reach from node *d* to node *i*, α is a control parameter, and, $p_{j,i}^{\ t}$ is the pheromone associated with node *j* in node *i*'s forwarding table during time *t*. The node *d* corresponds to the ant's origin *o* during its forward journey and to the node on which the search terminated during the ant's return journey. In the above pheromone update equation, the amount of pheromone deposited in node *i*'s forwarding table is proportional to the distance of node *i* from the ant's origin (or from the node on which the search terminated during the ant's return journey). This ensures that nodes further away from the search origin are given less pheromone when the search is unsuccessful so that later ants do not get considerable reinforcement from pheromone on distant nodes. Also, in Equation 1, the amount by which $p_{j,i}^{\ t}$ gets reinforced is inversely proportional to the amount of pheromone $p_{j,i}^{\ t-1}$ already existing for node *j* in node *i*'s forwarding table since time *(t-1)*. This ensures that nodes that already have sufficient amounts of pheromone do not receive excessive reinforcement and overshadow the reinforcement from other nodes in its neighborhood.

After each pheromone update in the forwarding table of node *i*, the pheromone on all nodes in node *i*'s forwarding table are renormalized according to equation 2 given below:

$$p_{j,i}^{\ t} \leftarrow p_{j,i}^{\ t} / \sum_j p_{j,i}^{\ t} \qquad (2)$$

Improvements on the Basic Ant Algorithm

The basic *ant* algorithm described in Figure 2 is suitable to search for resources in a static environment. However, resource search in P2P networks is dynamic in nature because different resources are located on different nodes, and nodes join and leave the network in an ad-

Figure 2. Basic algorithm used by an ant for P2P resource discovery

```
void antMove(String resourceName, Node origin)
{
N_i: set of neighbors of node i specified in the forwarding table of node i;
p_{j,i} : pheromone associated with neighbor j in forwarding table of node i, j ∈ N_i

Stack s; // stack containing nodes visited by ant
Node i;
int hopCount;
boolean success;

s ← {empty};
i ← origin;
hopCount ← 0;
success ← false;

while ((search(resourceName,i) == false) && (hopCount <= maxHops))
{
        // select the neighbor of node i that has the highest pheromone
        j ← arg max_j p_{j,i}
        updatePheromone(p_{j,i});

        // remember the node just visited
        s.push(i);

        // visit selected node next
        hopCount ← hopCount + 1;
        i ← j;
}


// set the success flag if resource was found before maxHops.
if (hopCount <= maxHops) success ← true;

// Ant retraces route.
while (s.isEmpty() == false)
{
j ← s.pop();
if (success == true) updatePheromone(p_{i,i});
i ← j;
}

}
```

hoc manner. To address these issues, the basic ant algorithm needs to be modified to make it suitable for P2P resource discovery. Resource discovery in P2P networks is different from the foraging behavior in an ant society in the following ways:

- **Selective Exploration:** In the traditional ant algorithm (Bonabeau et al., 1999), pheromone reinforces the attraction of ants to the route leading to the food. Such a mechanism is successful for locating food, whose physical location does not change with time. In contrast, in a P2P network, different resources are present on different nodes, and users search for different resources at different times. If ants blindly follow the pheromone trail within the P2P network established by an earlier ant, all ants are likely to end up at the node on which the earlier ant ended its search. Such a technique would not be successful when different resources being searched for are located on different nodes.
- **Node Exit:** In traditional ant algorithms, the ants cease to move when the pheromone trail decays to a sufficiently small

quantity and ants do not receive any positive reinforcement to direct their search. In a P2P network, nodes leave the network in an ad-hoc manner. This might result in breaking of a pheromone trail that passed through an exited node. Traditional ants encountering a broken trail do not receive any pheromone reinforcement to continue their search, and would cease their movement and revert to their origin, if a route to the origin exists. However, with such a response to broken trails, ants would not be able to find resources that are located on other nodes in the network and would adversely affect the search efficiency.

- **Node Entry:** New nodes joining the P2P network do not have a pheromone trail leading to them. Traditional ants that only follow the pheromone trail established by earlier ants would not be able to include newly joined nodes in their search path. Therefore, ants using the basic ant algorithm would fail to discover resources located on newly joined nodes.

To address these challenges for P2P resource discovery, we extend the traditional ant algorithm, as described below.

Anti-Pheromone Ant Algorithms

In the traditional ant algorithm, ants exhibit a high affinity to established pheromone trails. This affinity can be mitigated by enabling ants to explore newly joined nodes or nodes that are not along a successful search path. A suitable mechanism to enable exploration of nodes that are not reinforced with pheromone without degrading the search performance is provided by using *anti-pheromone*. Unlike positive reinforcement provided by pheromone, anti-pheromone provides negative reinforcement, and ants are repelled from a node marked with anti-pheromone. Anti-pheromone can be used to mark nodes along a trail that was established long ago, but has not yielded successful results recently. The negative reinforcement provided by the anti-pheromone repels ants from continuing to follow the established trail and directs the ants to explore unvisited nodes.

We employ four different types of anti-pheromone in our P2P resource discovery algorithm:

1. subtractive pheromone (SP) ants,
2. preferential anti-pheromone (PAP) ants,
3. anti-pheromone APE (APE) ants, and
4. preferential anti-pheromone APE (PAPE) ants.

Subtractive Pheromone (SP) Ants

The *subtractive pheromone* (SP) algorithm (Montgomery & Randall, 2002; Schoonderwoerd, Holland, Bruten, & Rothkrantz, 1996) was developed in order to remove poor solutions created in the past by normal ants. If an ant has a successful search, then the ant behaves as a normal ant during its return journey. On the other hand, if the search was unsuccessful and the ant reaches its search boundary, the ant removes pheromone from the nodes along its return journey. SP ants use the pheromone update equation given in equation 1 during their forward journey. During their return journey, the pheromone update equation for SP ants is given by:

$$\pi_i \leftarrow \pi_o / [h_{d,i}]^{\alpha},$$

$$p_{j,i}^{t} \leftarrow p_{j,i}^{t-1} + \pi_i (1 - p_{j,i}^{t-1}), \text{ if search was successful,}$$

$$p_{j,i}^{t} \leftarrow (1 - \pi_i) p_{j,i}^{t-1}, \text{ if search was unsuccessful,} \quad (3)$$

where d is the node on which the search terminates and all other parameters have the same meaning as in equation 1. Each pheromone update is followed by a re-normalization according to equation 2.

Preferential Anti-Pheromone (PAP) Ants

The *preferential anti-pheromone (PAP)* ants algorithm (Montgomery&Randall, 2002; Iredi, Merkle, & Middendorf, 2001) is used to solve bi-criterion optimization problems. It uses different pheromones for different criteria to allow multiple criteria to be improved in each iteration process. A PAP ant uses two types of pheromone: it deposits pheromone on nodes along a successful search path, while it deposits anti-pheromone along an unsuccessful search path. A PAP ant is associated with a parameter $\lambda \in [0,1]$ that denotes its probability to get attracted to pheromone. PAP ants use a variant of the ant algorithm described in Figure 2. While selecting the next node $j \in N_i$ to visit from node i, a PAP ant selects a node j using the criteria:

$$j \leftarrow arg\ max_j\ p_{j,i}^{\ t}, \text{ with probability } \lambda, \text{ and,}$$

$$j \leftarrow arg\ max_j\ p'_{j,i}^{\ t}, \text{ with probability } (1 - \lambda)$$

where $p_{j,i}^{\ t}$ and $p'_{j,i}^{\ t}$ are the amount of pheromone and anti-pheromone associated with node j in node i's forwarding table respectively at time t. The parameter $\lambda \in$ is determined experimentally to be 0.8 following results reported in Dasgupta (2004) to minimize the search latency.

The pheromone update equation used by PAP ants for a successful search path is identical to equation 1. For unsuccessful search paths, a PAP ant deposits anti-pheromone on nodes along the return route to its origin using the following equation:

$$\pi_i \leftarrow \pi_o / [h_{d,i}]^{\alpha},$$

$$p_{j,i}^{\ t} \leftarrow p_{j,i}^{\ t-1} + \pi_i\ (1 - p_{j,i}^{\ t-1}),\ \text{if search was successful,}$$

$$p'_{j,i}^{\ t} \leftarrow p'_{j,i}^{\ t-1} + \pi_i\ (1 - p'_{j,i}^{\ t-1}),\ \text{if search was unsuccessful,}$$

where d is the node on which the search terminates, $p'_{j,i}^{\ t}$ is the amount of anti-pheromone associated with node j in node i's forwarding table, and all other parameters have the same meaning as in equation 1. Each pheromone update is followed by a re-normalization according to equation 2.

Anti-Pheromone Explorer (APE) Ants

The *anti-pheromone explorer (APE)* ants algorithm (Montgomery & Randall, 2002) takes a different approach from the first two anti-pheromone algorithms by not reducing the pheromone value associated with poor solutions. Instead, a small number of ants—called anti-pheromone explorer ants—have their pheromone preference reversed and get attracted to nodes with low pheromone. APE ants are identical to the SP ants, and they use the pheromone update equations given in equation 3. However, APE ants' attraction to pheromone is reversed. Consequently, their selection criterion for a node j to visit from node i is given by:

$$j \leftarrow arg\ min_j\ p_{j,i}^{\ t},$$

where $p_{j,i}$ is the pheromone associated with node j in node i's forwarding table at time t.

PAP-APE Ants

The *PAP-Explorer (PAPE)* ants algorithm combines the behavior of PAP and APE ants.

PAPE ants behave like APE ants and get attracted to nodes with lower pheromone values. However, like PAP ants, PAPE ants also use pheromone to mark successful search paths and anti-pheromone to mark unsuccessful search paths.

EXPERIMENTAL RESULTS

We have used the JavaSwarm platform (SwarmWiki, 2005) as the simulation environment for experiments of our ant-based resource discovery algorithm. In each simulation, we have used normal ants and ants with four different anti-pheromone algorithms (SP, PAP, APE, and PAPE) to evaluate their relative performances. The performance is measured as "% of successful ants," which is given by the number of ants that succeeded in locating the resource they were searching for divided by the number of total ants used in each experiment. The P2P network used for our simulation contains between 280-310 nodes, the exact number of nodes being determined randomly. There are 30 ants initialized on different nodes for different resource queries. For all our experiments we have used the following constants: π_o=5.0, maxHops and α = 2.0. We use a metric called resource availability to simulate the distribution of resources over the nodes of the P2P network. Resource availability is defined as the percentage of nodes that contain at least one resource. The simulations are run for different resource availabilities representing the number of files available on each node. The different values of resource availability used are 5, 15, 40, and 65. All results were averaged over five simulation runs. Our objective across all these experiments is to determine the different types of ants and their proportions with respect to each other that yield a good search performance.

Sequences of Operations in Swarm Simulation Model

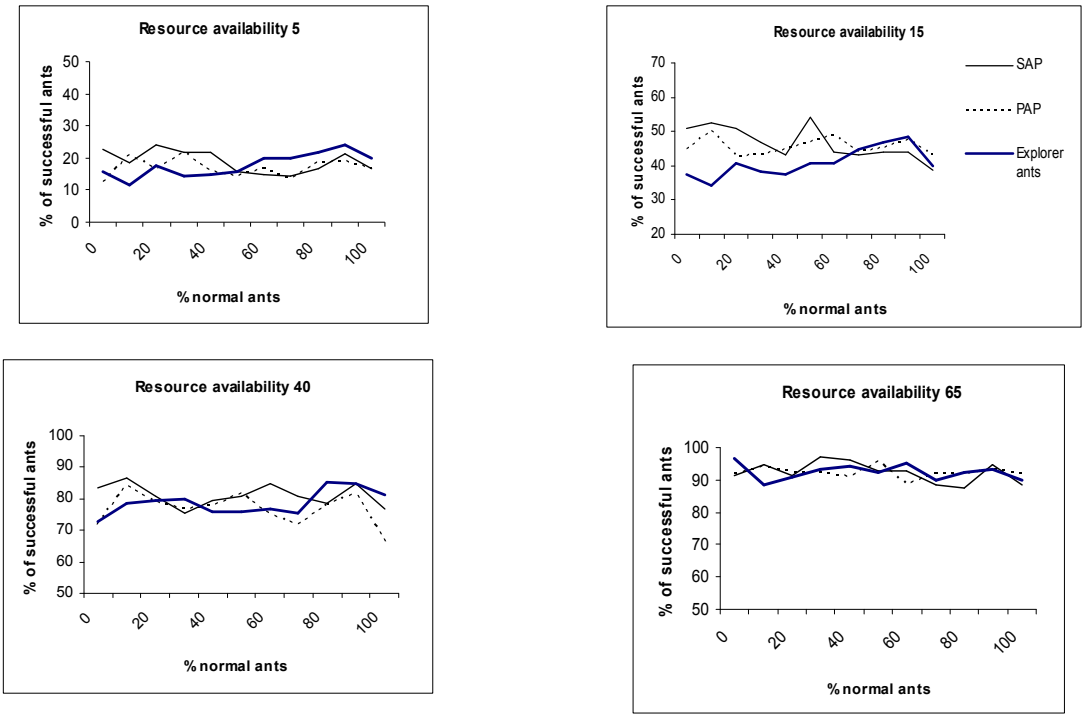
The sequences of operations for the swarm simulation of the P2P network are given below:

1. Create N nodes, where N ≅ 300.
2. Create a forwarding table for each node, and randomly select a number of neighbors (n), where n = U [5,…,10].
3. In the forwarding table for each node, initialize pheromone of each neighbor node to 1/n.
4. Create a resource table on each node. The size of the resource table is drawn from U [0,…,R_n], where R_n = 5 (very low resource availability), 15 (low resource availability), 40 (medium resource availability), and 65 (high resource availability).
5. Select a node at random and create an ant on it. An ant is specified by its startNodeID, maxHops, resource query (containing file name to search for), and ant's type.
6. Repeat step 5 for 30 times, each time selecting a new node randomly.
7. Each ant uses its own ant algorithm (normal, SP, PAP, APE, and PAP APE).

Simulation-I

Figure 3 shows the performance of the resource discovery algorithm when normal ants are used with SP, PAP, and APE ants respectively. At resource availabilities of 5 and 15, SP ants outperform PAP and APE when the percentage of normal ants is below 50%. This is because a strong proportion of anti-pheromone assists the search to remove poor solutions. At resource availabilities of 40 and 65, we could not find any distinction between the performances of different types of ants. This is because at high resource availability of 40-

Figure 3. Search performance when SP, PAP, and APE ants are used with normal ants for resource availabilities of 5, 15, 40, and 65



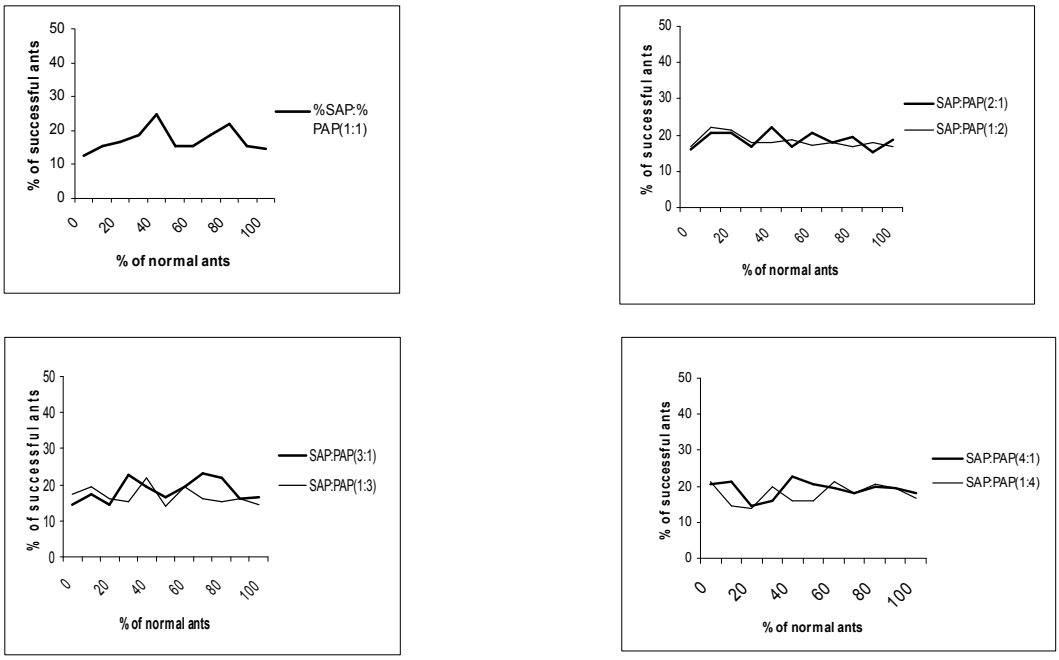
65%, most searches (with maxHops=10) are successful, and even a random walk among the nodes is likely to lead to a successful search. Therefore, different pheromone update strategies produce similar solutions.

In the results for PAP ants, there are no significant differences across all the experiments. This is because PAP ants behaves like normal ants except for their preference towards pheromone, λ. For the experiments shown, we have chosen $\lambda = 0.8$ as the preference towards normal pheromone, while only $(1-\lambda) = 0.2$ is used as the preference towards anti-pheromone. A low affinity towards anti-pheromone makes PAP ants behave like normal ants.

APE ants perform well when normal ants are between 70-90%, at resource availabilities of 5, 15, and 40. This is because APE ants assist in exploring those solutions that normal ants cannot reach. At resource availability 65, there is no significant difference. This is once again because of high resource availability.

By comparing the performance of SP, PAP, and APE ants at resource availability less than 40, we observe that SP ants have the best performance when normal ants are between 0-50%. This is because SP ants deposit anti-pheromone during their return journey from an unsuccessful search and prevent later ants from visiting the same nodes that returned a poor solution. Therefore, a higher number of SP ants than normal ants enable better search performance. On the other hand, APE ants seem to produce the best results when normal ants are between 50-100%. This is because APE ants are attracted to poorer search areas. Therefore, a high percentage of normal ants produce successful trails and the small percent-

Figure 4. Search performance when different ratios of SP and PAP ants are used with normal ants for resource availability=5



age of APE ants assist in exploring nodes that are not visited by the normal ants. Thus, a small proportion of APE ants are successful in yielding a good search performance. Finally, PAP ants produce the worst results among the others. This is because when resource availability is low, few nodes contain the resource being searched for, resulting in a low success rate for resource searches. In such a scenario, most nodes are marked with anti-pheromone by PAP ants. It is quite likely that nodes that do not contain a particular resource might contain a different resource. However, a PAP ant marks nodes from a previous unsuccessful search with anti-pheromone and consequently does not visit those nodes for later searches. Therefore, PAP ants perform the worst when resource availability is low. SP and APE ants do not have this problem because APE ants explore nodes marked with anti-pheromone, while SP ants use only one type of pheromone. Therefore, we observe that for this set of experiments, SP and APE ants perform better than PAP ants.

Simulation-II

This simulation compares the performance between SP and PAP ants. The experiment was simulated under different ratios of 1:1, 2:1, 3:1, 4:1, 1:2, 1:3, and 1:4 between SP and PAP ants.

Figure 4 shows the results at resource availability 5. This simulation produces equal performance for the combination of SP and PAP with normal ants, and is due to a low resource availability to distinguish the performance of the different types of ants.

Figure 5 shows the experiments with different ratios of SP and PAP ants for a resource availability of 15. The combination of SP and PAP ants in the proportion of 2:1 produces slightly better results than with a proportion of

Figure 5. Search performance when different ratios of SP and PAP ants are used with normal ants for resource availability=15

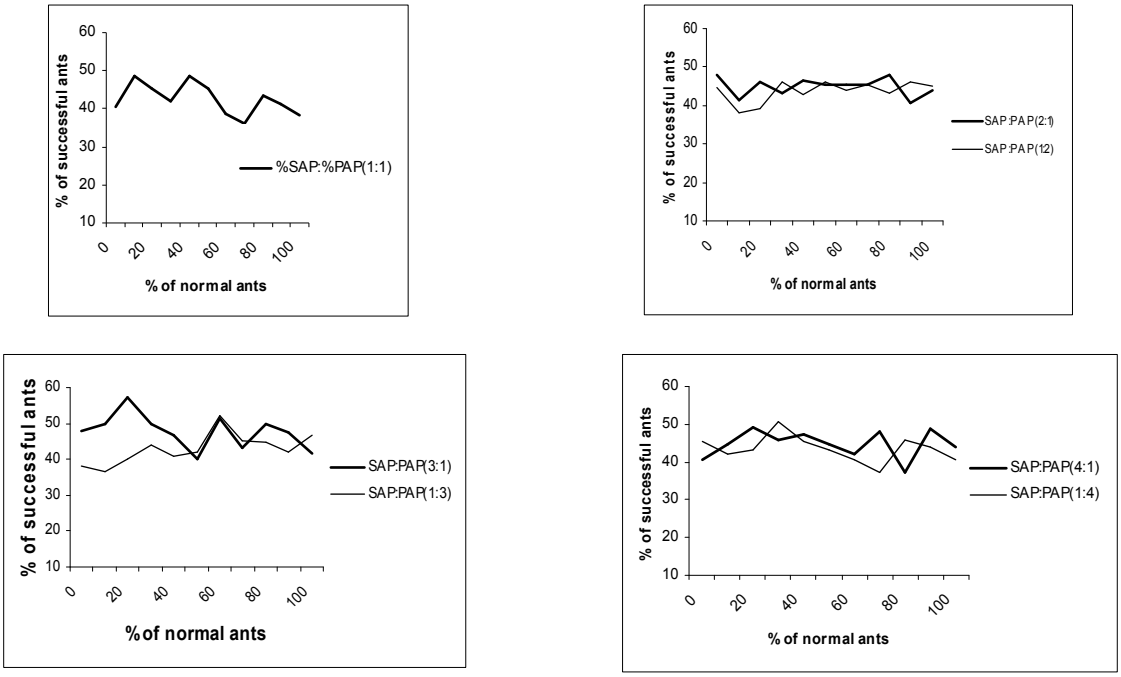


Figure 6. Search performance when different ratios of SP and PAP ants are used with normal ants for resource availability=40

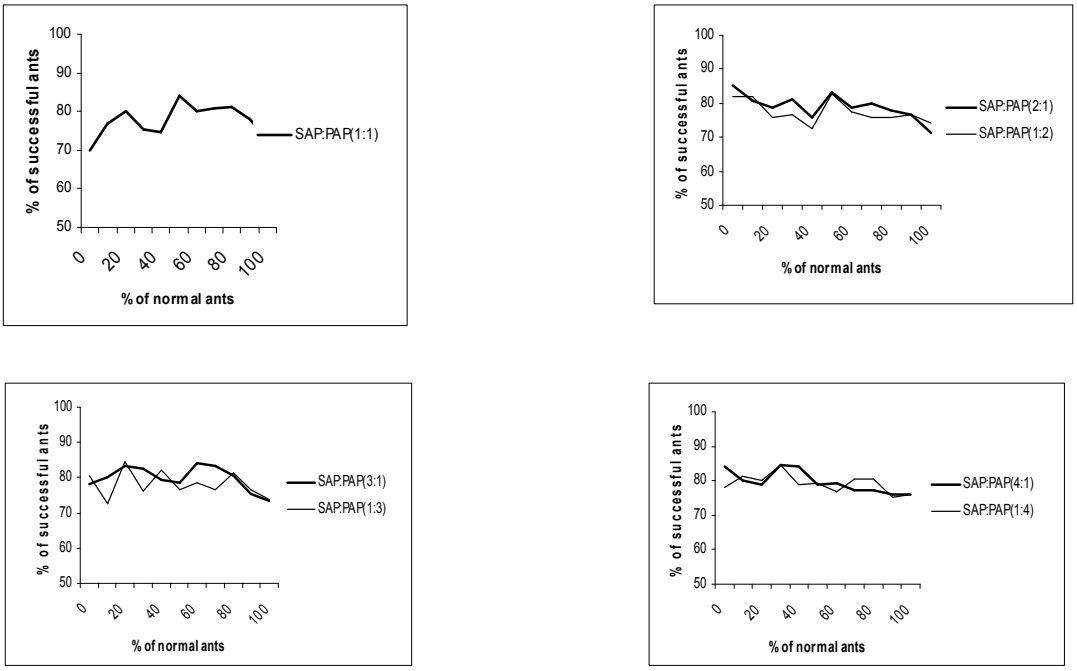
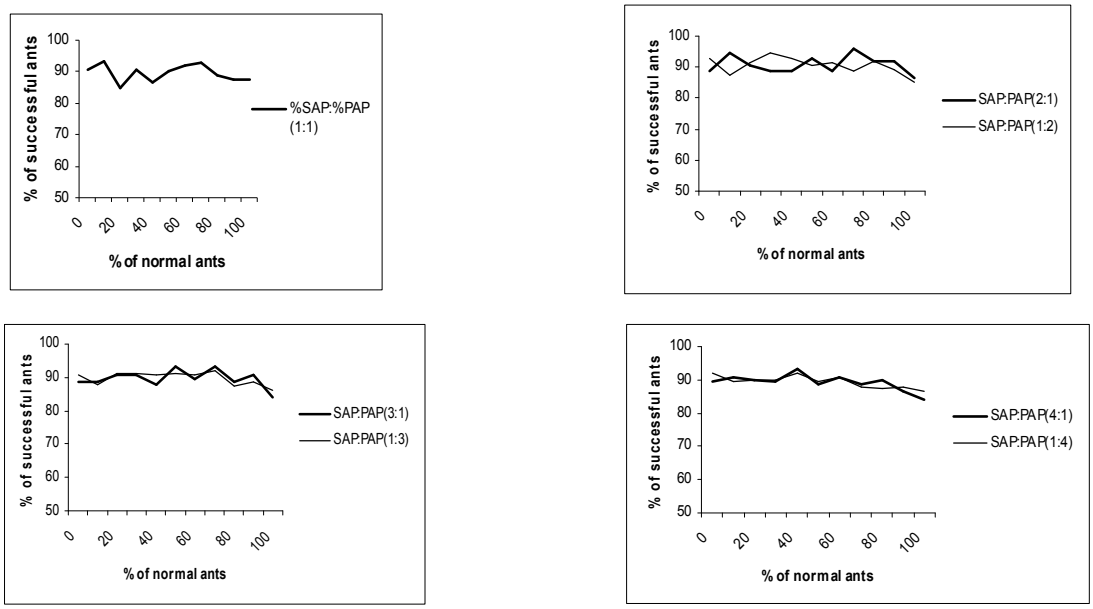


Figure 7. Search performance when different ratios of SP and PAP ants are used with normal ants for resource availability=65



1:2 between SP and PAP ants when normal ants are between 0-80%. Likewise, SP and PAP ants at a ratio 3:1 give a better result than with a ratio of 1:3 between SP and PAP when normal ants are between 0-50%. Finally, the combination of SP and PAP in the ratio of 4:1 performs slightly better than with a SP:PAP ratio of 1:4 when normal ants are between 0-80%. Therefore, this simulation suggests that a combination of more SP than PAP ants when normal ants are 0-50% will yield a good search performance at resource availability 15.

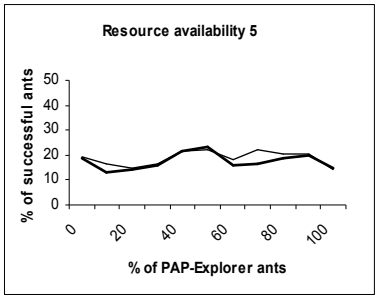
Similar results are obtained for the search performance when the resource availability is varied for different proportions of SP and PAP ants with normal ants, as illustrated in Figures 6 and 7.

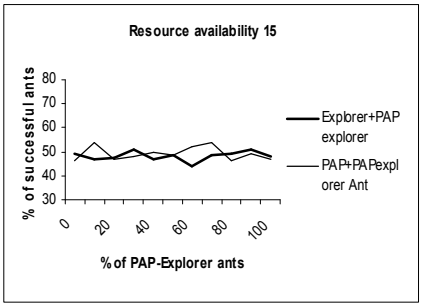
In general, the results from simulation-II indicate that SP ants produce slightly better results than PAP ants. This can be attributed to the fact that PAP ants used a pheromone preference of λ=0.8 for normal pheromone and (1-λ)=0.2 for anti-pheromone. This value of λ makes PAP ants behaves like normal ants. Therefore, SP with a strong preference of anti-pheromone assists the search to remove poor solutions and produce better performances than PAP.

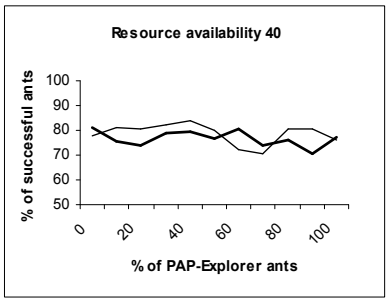
Simulation-III

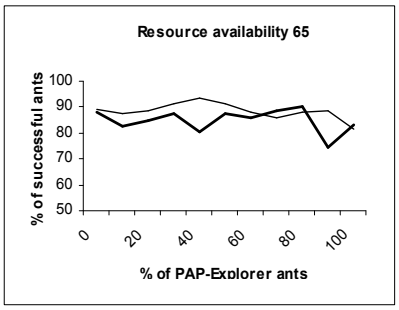
This experiment shows the performance comparison between PAP with PAPE ants and APE with PAPE ants. As shown in Figure 8, at resource availabilities 5 and 15, they have similar performance because there are too few resources among the nodes of the network. At resource availabilities of 40 and 65, PAP with PAPE ants produces better results. This is because APE with PAPE ants have a strong preference for anti-pheromone which distracts the search too much and prevents these ants from searching for the solution.

Figure 8. Results of Simulation-III for PAP, APE ants, and PAP APE ants at resource availabilities 5, 15, 40, and 65









In the above set of experiments, we have simulated resource discovery in P2P networks using different proportions of ants implementing different ant algorithms. The results obtained illustrate the appropriate proportion of ants required to perform an efficient search varies with the availability of resources in the network, the percentage of normal ants, and the different proportions of various anti-pheromone-enabled ants used in the ant population.

RELATED WORK

Ant algorithms have already been applied to several applications (Bonabeau et al., 1999; Dasgupta, 2004) including dynamic programming, the traveling salesman problem, and routing in telecommunication networks (Schoonderwoerd et al., 1996). However, resource discovery in P2P networks is different from each of these applications because the node on which the resource will be discovered is not known *a priori* and the topology of the P2P network can change dynamically as nodes join and leave. Extensions to ant algorithms using anti-pheromone for the traveling salesman problem have been studied in Montgomery and Randall (2002). The Anthill framework (Babaoglu, Meling, & Montresor, 2002) employs ant-based algorithms for load balancing in a P2P network, and ants backtrack along the path they traveled to update routing tables at each node. In contrast, our algorithm uses different types of pheromone and ants with different behavior to make P2P resource discovery more efficient.

CONCLUSION AND FUTURE WORK

In this chapter, we have described an informed search algorithm using an ant-based heuristic

for P2P resource discovery. We are currently investigating extensions to the algorithm described in this chapter using multiple ants to enable parallel search queries. We are also exploring evolutionary algorithms to enable ants to rapidly discover routes to resourceful nodes. We also propose to develop a cooperative multi-agent framework that allows ants from different nodes to exchange trail information with each other to locate resources rapidly. We envisage that ant algorithms implemented through software agents provide a useful direction for further exploring challenges and issues of management of P2P networks for future research.

REFERENCES

Anderson, D., Cobb, J., Korpela, E., Lebofsky, M., & Werthimer, D. (2002). SETI@home: An experiment in public-resource computing. *Communications of the ACM, 45*(11), 56-61.

Babaoglu, O., Meling H., & Montresor, A. (2002). Anthill: A framework for the development of agent-based peer-to-peer systems. *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*, Vienna, Austria (pp. 15-22).

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems*. Oxford, UK: Oxford University Press.

Dasgupta, P. (2004). *Improving peer-to-peer resource discovery using mobile agent based referrals* (LNCS 2872, pp. 186-197). Berlin: Springer-Verlag.

Gnutella. (2005). Retrieved from http://www.gnutella.com

Heck, P., &. Ghosh, S. (2000). A study of synthetic creativity through behavior modeling and simulation of an ant colony. *IEEE Intelligent Systems, 15*(6), 58-66.

Iredi, S., Merkle, D., & Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms. *Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization (EMO'01)*, Zurich (pp. 359-372).

Lin, M., Marzullo, K., & Masini, S. (1999). *Gossip versus deterministic flooding: Low message overhead and high reliability for broadcasting on small networks*. Technical Report CS1999-0637, University of California, San Diego, USA.

Manku, G., Naor, M., & Manber, U. (2004). Know thy neighbor's neighbor: The power of lookahead in randomized P2P networks. *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, Chicago, IL (pp. 54-63).

Marti, S., & Garcia-Molina, H. (2004). Limited reputation sharing in P2P systems. *Proceedings of the 5th ACM Conference on E-Commerce*, New York (pp. 91-101).

Montgomery, J., & Randall, M. (2002). *Anti-pheromone as a tool for better exploration of search space* (LNCS 2463, pp. 100-110). Berlin: Springer-Verlag.

Portmann, M., & Seneviratne, A. (2002). Cost-effective broadcast for fully decentralized peer-to-peer networks. *Computer Communication, Special Issue on Ubiquitous Computing, 26*(11), 1159-1167.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., & Shenker, S. (2001). A scalable content-addressable network. *Proceedings of ACM SIGCOMM* (pp. 161-172), San Diego, CA.

Rowstron, A., & Druschel, P. (2001). Pastry: Scalable, distributed object location and routing

for large-scale peer-to-peer systems. *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)* (pp. 329-350), Heidelberg, Germany.

Schoonderwoerd, R., Holland, O., Bruten, J., & Rothkrantz, L. (1996). Ant-based load balancing in telecommunications networks. *Adaptive Behavior, 2,* 169-207.

Search for Extra-Terrestrial Intelligence (SETI)@Home. (2005). Retrieved from *http://setiathome.ssl.berkeley.edu*

Stoica, I., Morris, R., Karger, D., Kaashoek, F., & Balakrishnan H. (2001). Chord: A peer-to-peer lookup service for Internet applications. *Proceedings of the ACM SIGCOMM*, San Diego, CA.

SwarmWiki. (2005). Retrieved from http://www.swarm.org

Walkerdine, J., & Rayson, P. (2004). P2P-4-DL: Digital library over peer-to-peer. *Proceedings of the IEEE Conference on Peer-to-Peer Computing* (pp. 264-265).

Weiss, G. (Ed.). (1998). *Multi-agent systems: A modern approach to distributed artificial intelligence.* Cambridge, MA: MIT Press.

Yang, B., & Garcia-Molina, H. (2003). Designing a super-peer network. *Proceedings of the 19th International Conference on Data Engineering*, Bangalore, India (pp. 49-60).

Yu, B., & Singh M. (2003). Searching social networks. *Proceedings of the 2nd International Conference on Autonomous Agents and Multi-Agent Systems*, Bologna, Italy (pp. 65-72).

Zhao, B., Kubiatowicz, J., & Joseph, A. (2001). *Tapestry: An infrastructure for fault resilient wide-area location and routing.* Technical Report UCB CSD-01-1141, University of California, Berkeley, USA.

KEY TERMS

Ant Algorithm: Essentially, a reinforcement learning algorithm. In an ant algorithm, points that are good candidates for a solution to a problem get reinforced over time. When the algorithm terminates, the points that have been reinforced the most are selected as the solution(s) to the problem.

Anti-Pheromone: Unlike pheromone, anti-pheromone is used to provide negative reinforcement in an ant algorithm. Points in the solution space that are marked with anti-pheromone are usually bad solutions to the problem. The negative reinforcement provided by the anti-pheromone ensures that these points are not included in successive iterations of the ant algorithm.

Node Discovery: The mechanism used by a newly joining node in a P2P network to determine the IP addresses of existing nodes in the network and selectively choose them as neighbors.

Peer: A node in a P2P network. A peer can provide services to other peers and also receive services from other peers. Because it can behave both as a server and a client, a peer is also known as a servant.

Peer-to-Peer Network: A network consisting of an overlay or logical network between nodes that are interconnected by an underlying physical network. Users are located on nodes of the P2P network. A user at a node can store resources such as data files on its node and can share these resources with other nodes, as well as acquire new resources from other nodes in the network.

Pheromone: The metric used in an ant algorithm to reinforce points in the solution space of a problem. Pheromone decays over time.

Resource Discovery: The mechanism used by a node to search for resources among other nodes of the network. A user at a node provides a search query containing a set of keywords corresponding to the resource being searched for. A resource discovery protocol is then used to forward the search query to other nodes and search for the resource on those nodes. When a resource is located at a node, a success message is sent to the node that originated the search query.

Structured P2P Network: In such a network, the network growth follows a specific topology as a ring or a hypercube. Resources are placed on nodes using a distributed hash table (DHT)-based algorithm. In structured P2P networks, operations such as determining the location of a newly joining node within the network and searching for resources are performed using deterministic algorithms.

Unstructured P2P Network: Networks that do not use a pre-determined topology and grow in an ad-hoc manner. A node in an unstructured P2P network makes the decision to accept another node as a neighbor based on the number of resources shared by the latter node.

Section VII

Commerce and Negotiation

Chapter XLVIII
An Annealing Protocol for Negotiating Complex Contracts

Mark Klein
Massachusetts Institute of Technology, USA

Peyman Faratin
Massachusetts Institute of Technology, USA

Hiroki Sayama
University of Electro-Communications, Japan

Yaneer Bar-Yam
New England Complex Systems Institute, USA

ABSTRACT

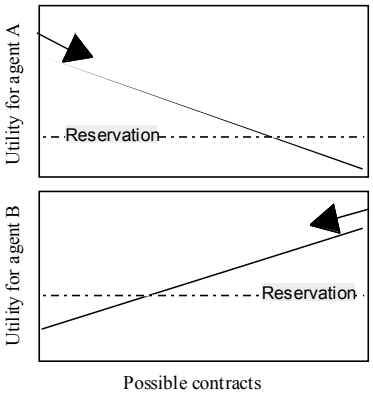
Work to date on negotiation protocols has focused almost exclusively on defining contracts consisting of one or a few independent issues and a relatively small number of possible contracts. Many real-world contracts, by contrast, are much more complex, consisting of multiple interdependent issues and intractably large contract spaces. This chapter describes a simulated annealing-based approach appropriate for negotiating such complex contracts that achieves near-optimal social welfare for negotiations with binary issue dependencies.

INTRODUCTION

Work to date on negotiation protocols has focused almost exclusively on negotiating what we can call *simple* contracts—that is, contracts consisting of one or a few independent issues (Faratin, Sierra, & Jennings, 2000; Ehtamo, Ketteunen, & Hamalainen, 2001; Fisher, Ury, & Patton, 1991; Raiffa, 1982). These protocols work, in general, via the iterative exchange of proposals and counterproposals. An agent starts with a contract that is optimal for that agent and makes concessions, in each subsequent proposal, until either an agreement is reached or the negotiation is abandoned because the utility of the latest proposal has fallen below the agents' reservation value (see Figure 1).

This is a perfectly reasonable approach for simple contracts. Since issues are independent,

Figure 1. The proposal exchange model of negotiation, applied to a simple contract. Each point on the X axis represents a possible contract. The Y axis represents the utility of a contract to each agent.



the utility of a contract for each agent can be calculated as the weighted sum of the utility for each issue. The utility function for each agent is thus a simple one, with a single optimum and a monotonic drop-off in utility as the contract diverges from that ideal. Simple contract negotiations thus typically progress as in Figure 2.

As we can see, the proposals from each agent start at their own ideal, and then track the Pareto frontier until they meet in the middle with an optimal agreement. This happens because, with linear utility functions, it is easy for an agent to identify the proposal that represents the minimal concession: the contract that is minimally worse than the current one is "next" to the current one in the contract space and can be found by moving in the direction with the smallest aggregate utility slope. The simplicity of the utility functions, moreover, makes it feasible for agents to infer enough about their opponents that they can identify concessions that are attractive to each other, resulting in relatively quick negotiations.

Real-world contracts, by contrast, are generally much more complex, consisting of a large number of inter-dependent issues. A typical contract may have tens or even hundreds of

Figure 2. A typical negotiation for a simple contract. The contract consisted in this case of 40 binary issues. Each agent was required to reduce the Hamming distance (number of issues with different values) between successive proposals until an agreement was reached. The Pareto frontier was estimated by applying an annealing optimizer to differently weighted sums of the two agents' utility functions.

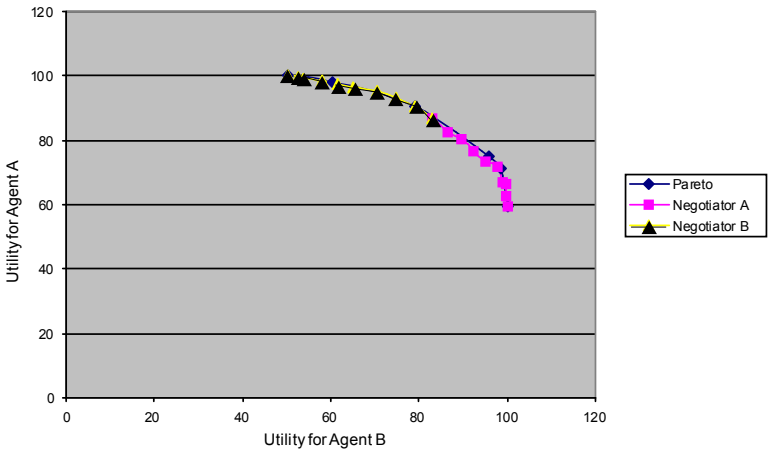
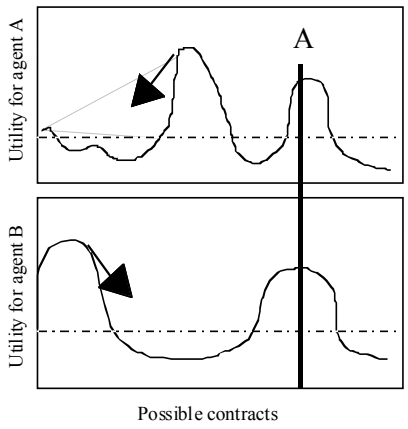


Figure 3. An example of proposal exchange applied to a complex contract

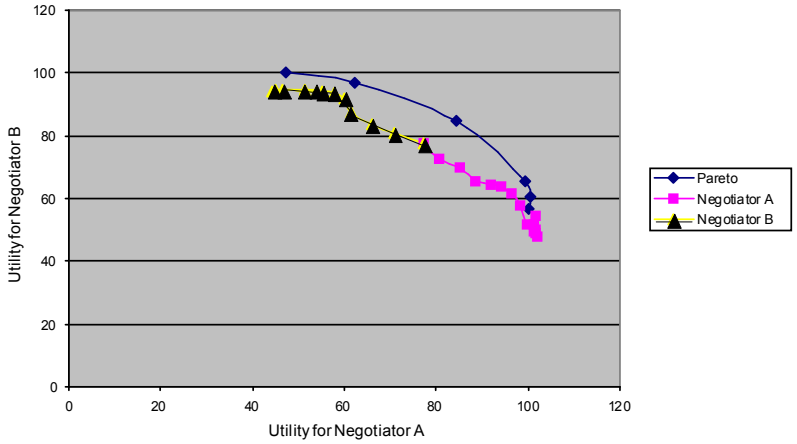


distinct issues. Even with only 40 issues and two alternatives per issue, we encounter a search space of roughly 1 trillion possible contracts, too large to be explored exhaustively. The value of one issue selection to an agent, moreover, will often depend on the selection made for another issue. The value to me of a given couch, for example, depends on whether it is a good match with the chair I plan to purchase with it. Such issue interdependencies lead to nonlinear utility functions with *multiple* local optima (Bar-Yam, 1997).

In such contexts, an agent finding its own ideal contract becomes a nonlinear optimization problem, difficult in its own right. Simply conceding toward the other agents' proposals can result in the agent's missing contracts that would be superior from both their perspectives (e.g., the contract labeled "A" in Figure 3). Standard negotiation techniques thus typically produce the behavior in Figure 4 when applied to complex contract negotiation.

The agents start with an approximation to their ideal contract and diverge increasingly from the Pareto frontier as they converge upon an agreement. The degree of sub-optimality depends on the details of the utility function. In our experiments, for example, the final contracts averaged 94% of optimal. This is a substantial decrement when you consider that agents can average 50% of optimal simply by choosing random contracts. The utility functions we used for each agent were, moreover, individually quite easy to optimize: a simple steepest ascent search averaged final utility

Figure 4. A typical negotiation for a complex contract. This example differs from Figure 2 only in that a nonlinear utility function was used by each agent (details follow in text).



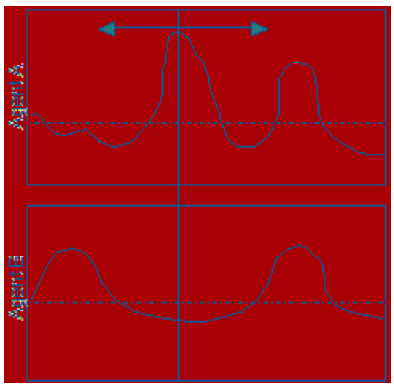
values of roughly 97% of those reached by a nonlinear optimization algorithm. It is striking that such relatively forgiving multi-optima utility functions lead to substantially sub-optimal negotiation outcomes.

These sub-optimal outcomes represent a fundamental weakness with current negotiation techniques. The only way to ensure that subsequent proposals track the Pareto frontier, and thus conclude with a Pareto optimal result, is to be able to identify the proposal that represents the minimal concession from the current one. But in a utility function with multiple optima, that proposal may be quite distant from the current one, and the only way to find it is to exhaustively enumerate all possible contracts. This is computationally infeasible, however, due to the sheer size of the contract space. Since the utility functions are quite complex, it is in addition no longer practical for one agent to infer the other's utility function. Complex contracts therefore require different negotiation techniques which allow agents to find 'win-win' contracts in intractable multi-optima search spaces in a reasonable amount of time. In the following sections we describe a family of negotiation protocols that make substantial progress towards achieving these goals.

MEDIATED SINGLE-TEXT NEGOTIATION

A standard approach to dealing with complex negotiations in human settings is the mediated single-text negotiation (Raiffa, 1982). In this process, a mediator proposes a contract that is then critiqued by the parties in the negotiation. A new, hopefully better proposal is then generated by the mediator based on these responses. This process continues, generating successively better contracts, until some agreed-upon stopping point (e.g., the reservation utility value is met or exceeded for both parties). We can visualize this process as shown in Figure 5.

Figure 5. Single-text negotiation



Here, the vertical line represents the contract currently proposed by the mediator. Each new contract moves the line to a different point on the X axis. The goal is to find a contract that is sufficiently good for both parties.

We defined a simple experiment to help us explore computationally how well this approach actually works. In this experiment, there were two agents negotiating to find a mutually acceptable contract consisting of a vector S of 40 Boolean-valued issues, each issue assigned the value 0 or 1. This formulation is fully general, because we can model Nary issues (issues that have more than two possible values) simply by treating several bits as corresponding to a single issue. This defined a space of 2^40, or roughly 10^12, possible contracts. Each agent had a utility function calculated using its own 40x40 influences matrix H, wherein each cell represents the utility increment or decrement caused by the presence of a given pair of issues, and the total utility of a contract is the sum of the cell values for every issue pair present in the contract:

$$U = \sum_{i=1}^{40} \sum_{j=1}^{40} H_{ij} S_j S_j$$

The influence matrix therefore captures the dependencies between issues, in addition to the value of any individual contract clause. For our experiments, the utility matrix was initialized to have random values between –1 and +1 in each cell. A different influences matrix was used for each simulation run, in order to ensure our results were not idiosyncratic to a particular configuration of issue inter-dependencies.

The mediator proposes a contract that is initially generated randomly. Each agent then votes to accept or reject the contract. If both vote to accept, the mediator mutates the contract (by randomly flipping one of the issue values) and the process is repeated. If one or both agents vote to reject, a mutation of the most recent mutually acceptable contract is proposed instead. The process is continued for a fixed number of proposals. Note that this approach can straightforwardly be extended to a N-party (i.e., multi-lateral) negotiation, since we can have any number of parties voting on the contracts.

We defined two kinds of agents: *hill-climbers* and *annealers*. The hill-climbers use a very simple decision function: they accept a mutated contract only if its utility to them is greater than that of the last contract both agents accepted. Annealers are more complicated, implementing a Monte Carlo machine (Kalos & Whitlock, 1986). Each annealer has a virtual 'temperature' T, such that it will accept contracts worse than the last accepted one with the probability:

$$P(\text{accept}) = \max(1, e^{\Delta U/T})$$

where ΔU is the utility change between the contracts. In other words, the higher the virtual temperature and the smaller the utility decrement, the greater the probability that the inferior contract will be accepted. The virtual temperature of an annealer gradually declines over time so eventually it becomes indistinguishable from a hill-climber. Annealing has proven effective in single-agent optimization, because it can travel through utility valleys on the way to higher optima (Bar-Yam, 1997). This suggests that annealers can be more successful than hill-climbers in finding good negotiation outcomes.

Table 1. Annealing vs. hill-climbing agents

	Agent 2 hill-climbs	Agent 2 anneals
Agent 1 hill-climbs	[.86] .73/.74	[.86] .99/.51
Agent 1 anneals	[.86] .51/.99	[.98] .84/.84

THE PRISONER'S DILEMMA

Negotiations with annealing agents did indeed result in substantially superior final contract utilities, but as the payoff table (Table 1) shows, there is a catch.

In this table, the cell values are laid out as follows:

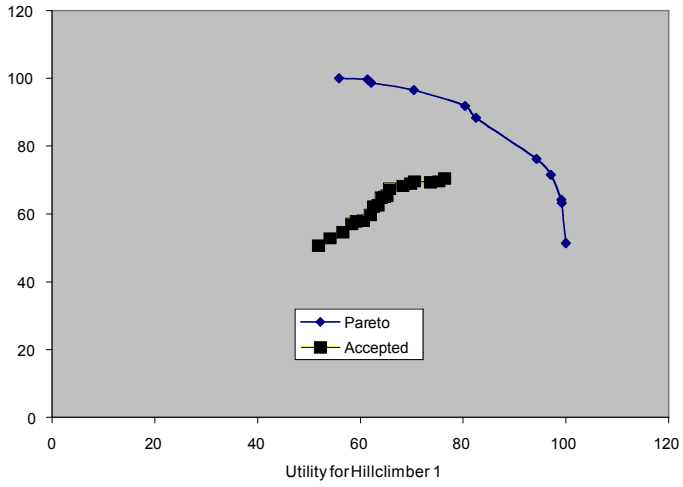
[<social welfare optimality>]
<agent 1 optimality >/<agent 2 optimality >

The social welfare of a contract is given by the sum of the agent's utilities for that contract; social welfare optimality thus represents how close the social welfare of a contract comes to the highest achievable social welfare (i.e., the Pareto contract with the highest social welfare).

As expected, paired hill-climbers do relatively poorly while paired annealers do very well. If both agents are hill-climbers, they both get a poor payoff, since it is difficult to find many contracts that represent an improvement for both parties. A typical negotiation with two hill-climbers is shown in Figure 6.

As we can see, in this case the mediator was able to find only two contracts that increased

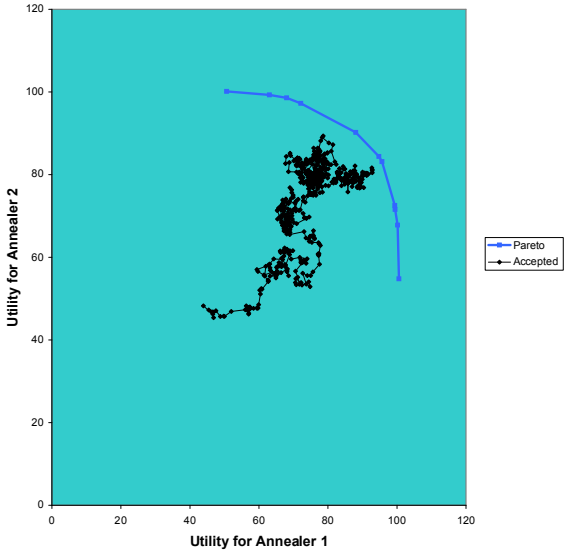
Figure 6. A typical negotiation with two hill-climbers



the utility for both hill-climbers, and ended up with a poor final social welfare.

Near-optimal social welfare can be achieved, by contrast, when both agents are annealers, willing to initially accept individually worse contracts so they can find win-win contracts later on (see Figure 7).

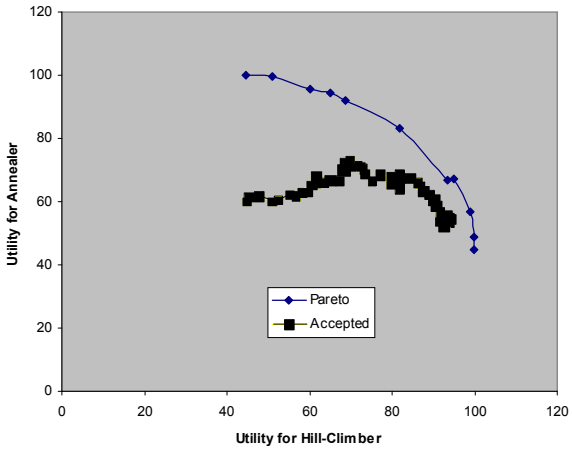
Figure 7. A typical negotiation with two annealers



The agents entertain a much wider range of contracts, eventually ending very near the Pareto frontier.

If one agent is a hill-climber and the other is an annealer, however, the hill-climber does extremely well but the annealer fares correspondingly poorly (see Figure 8). This pattern can be understood as follows. When an annealer is at a high virtual temperature, it becomes a

Figure 8. A typical negotiation with an annealer and hill-climber



chronic conceder, accepting almost anything beneficial or not, and thereby pays a "conceder's penalty." The hill-climber *drags* the annealer towards its own local optimum, which is not very likely to also be optimal for the annealer:

This reveals a dilemma. In negotiation contexts we typically cannot assume that agents will be altruistic, and we must as a result design protocols such that the individually most beneficial negotiation strategies also produce the greatest social welfare (Rosenschein & Zlotkin, 1994). In our case, however, even though annealing is a *socially* dominant strategy (i.e., annealing increases social welfare), annealing is not an *individually* dominant strategy. Hill-climbing is dominant, because no matter what strategy the other agent uses, it is better to be a hill-climber (see Table 1). If all agents do this, however, then they forego the higher individual utilities they would get if they both annealed. Individual rationality thus drives the agents towards the strategy pairing with the *lowest social welfare*. This is thus an instance of the prisoner's dilemma. It has been shown that this dilemma can be avoided if we assume repeated interactions between agents (Axelrod, 1984), but we would prefer to have a negotiation protocol that incents socially beneficial behavior without that difficult-to-enforce constraint. Several straightforward approaches to this problem, however, prove unsuccessful. One possibility is to simply reduce the annealer's willingness to make concessions. This can indeed eliminate the conceder's penalty, but at the cost of achieving social welfare values only slightly better than that achieved by two hill climbers. Another option is to have agents switch from being an annealer to a hill-climber if they determine, by observing the proposal acceptance rates of their opponents, that the other agent is being a hill-climber. We found, however, that it takes too long to determine the type of the other agent: by the time it has become clear, much of the contract utility has been committed, and it is too late to recover from the consequences of having started out as an annealer. See Klein, Faratin, and Bar-Yam (2002) for details.

THE ANNEALING MEDIATOR

We were able to develop a negotiation protocol that avoids the prisoner's dilemma entirely in mediated single-text negotiation of complex contracts. The trick is simple: rather than requiring that the negotiating agents anneal and thereby expose themselves to the risk of being dragged into bad contracts, we moved the annealing into the mediator itself. In our original protocol, the mediator would simply propose modifications of the last contract both negotiating agents accepted. In our refined protocol, the mediator is endowed with a time-decreasing willingness to follow up on contracts that one or both agents rejected (following the same inverse exponential regime as the annealing agents). Agents are free to remain hill-climbers and thus avoid the potential of making harmful concessions. The mediator, by virtue of being willing to provisionally pursue utility-decreasing contracts, can traverse valleys in the agents' utility functions and thereby lead the agents to win-win solutions. We describe the details of our protocol, and our evaluations thereof, below.

In our initial implementations each agent gave a simple accept/reject vote for each proposal from the mediator, but we found that this resulted in final social welfare values significantly lower than what we earlier achieved using annealing agents. In our next round of experiments, we accordingly modified the agents so that they provide additional information to the mediator in the form of vote strengths: each agent annotates an accept or reject vote as being *strong* or *weak*. The agents were

designed so that there are roughly an equal number of weak and strong votes of each type. This maximizes the informational content of the vote strength annotations. When the mediator receives these votes, it maps them into numeric values (strong accept = 1, weak accept = 0, weak reject = -1, strong reject = -2) and adds them together to produce an aggregate score. A proposal is accepted by the mediator if the score is non-negative—that is, if both agents voted to accept it or if a weak reject by one agent is overridden by a strong accept from the other. The mediator can also accept rejected contracts (i.e., those with a negative aggregate score) using the annealing scheme described above. This approach works surprisingly well, achieving final social welfare values that average roughly 99% of optimal despite the fact that the agents each supply the mediator with only two bits of information. This additional bit of information is critical, however, because it allows the system to pursue social welfare-increasing contracts that cause a utility decrement for one agent.

In more recent work, we have applied the same protocol to negotiations with three agents, again achieving final social welfare values that average roughly 99% of optimal.

INCENTIVES FOR TRUTHFUL VOTING

Any voting scheme introduces the potential for strategic non-truthful voting by the agents, and our scheme is no exception. Imagine that one of the agents always votes truthfully, while the other exaggerates so that its votes are always 'strong'. One might expect that this would bias negotiation outcomes to favor the exaggerator and this is in fact the case (see Table 2).

As we can see, even though exaggerating has substantial negative impact on social welfare, agents are individually incented to exaggerate, thus re-creating the prisoner's dilemma we encountered earlier. The underlying problem is simple: exaggerating agents are able to induce the mediator to accept all the proposals that are advantageous to them (if they are weakly rejected by the other agent), while preventing the other agent from doing the same. What we need, therefore, is an enhancement to the negotiation protocol that incents truthful voting, preserving equity and maximizing social welfare.

Table 2. Truth-telling vs. exaggerating agents with a simple annealing mediator

	Agent 2 exaggerates	Agent 2 tells truth
Agent 1 exaggerates	[.92] .81/.81	[.93] .93/.66
Agent 1 tells truth	[.93] .66/.93	[.99] .84/.84

How can this be done? We found that simply placing a limit on the number of strong votes each agent can use does not work. If the limit is too low, we effectively lose the benefit of vote weight information and get the lower social welfare values that result. If the strong vote limit is high enough to avoid this, then all an exaggerator has to do is save all of its strong votes until the end of the negotiation, at which point it can drag the mediator towards making a series of proposals that are inequitably favorable to it.

Another possibility is to enforce overall parity in the number of "overrides" each agent gets. A override occurs when a contract supported by one agent (the "winner") is accepted by the mediator over the objections of the other agent. Overrides are what drags a negotiation towards contracts favorable to the winner, so it makes sense to make the total number of overrides equal for each agent. But this is not enough, because exaggerators always win disproportionately more than the truth-teller.

Table 3. Truth-telling vs. exaggerating agents with parity-enforcing mediator

	Agent 2 exaggerates	Agent 2 tells truth
Agent 1 exaggerates	[.91] .79/.79	[.92] .78/.81
Agent 1 tells truth	[.92] .81/.78	[.98] .84/.84

The solution, we found, came from enforcing parity between the number of overrides given to each agent *throughout* the negotiation, so neither agent can get more than a given advantage. This way, at least rough equity is maintained no matter when (or whether) either agent chooses to exaggerate. The results of this approach are shown in Table 3; the override disparity was limited to 3.

When we have truthful agents, we find that this approach achieves social welfare just slightly below that achieved by a simple annealing mediator, while offering a significantly ($p < 0.01$) higher payoff for truth-tellers than exaggerators. We found, moreover, that the same pattern of results holds for a range of exaggeration strategies, including exaggerating all the time, exaggerating at random, or exaggerating just near the end of the negotiation. Truth-telling is thus both the individually dominant and socially most beneficial strategy.

Why does this work? Why, in particular, does a truth-teller fare better than an exaggerator with this kind of mediator? One can think of this procedure as giving agents 'tokens' that they can use to 'purchase' advantageous overrides, with the constraint that both agents spend tokens at a roughly equal rate. Recall that in this case a truthful agent, offering a mix of strong and weak votes, is paired with an exaggerator for whom at least some weak accepts and rejects are presented as strong ones. The truthful agent can therefore only get an override via annealing (see Table 3), and this is much more likely when its vote was a strong accept rather than a weak one. In other words, the truthful agent spends its tokens almost exclusively on contracts that truly offer it a strong utility increase. The exaggerator, on the other hand, will spend tokens to elicit a override even when the utility increment it derives is relatively small. At the end of the day, the truthful agent has spent its tokens more wisely and to better effect.

CONTRIBUTIONS

We have shown that negotiation involving complex contracts (i.e., those with many multiple inter-dependent issues) has properties that are substantially different from the simple (independent issue) case that has been studied to date in the negotiation literature, and requires as a result different protocols. This chapter presents, as far as we are aware, the first negotiation protocol designed specifically for complex contracts. While some previous work has studied multi-issue negotiation (e.g., Faratin et al., 2000; Oliveira, Fonseca, & Garção, 1999; Kowalczyk & Bui, 2001; Ströbel, 2000; Barbuceanu & Lo, 2000; Jonker & Treur, 2001; Fatima, Wooldridge, & Jennings, 2002), the issue utilities in these efforts are treated as independent, so the utility functions for each agent are linear, with single optima. As we have seen, however, the introduction of multiple optima changes the game drastically. Multi-attribute auctions (Bichler, Kaukal, & Segev, 1999) represent another scheme for dealing with multiple issues, wherein one party (the buyer) publishes its utility function, and the other parties (the sellers) make bids that attempt to maximize the utility received by the buyer. If none of the bids are satisfactory, the buyer modifies its published utility function and tries again. This introduces a search process,

and the problem with this approach is that it does not provide any guidance for how the parties involved should control their search through the vast space of possibilities.

The essence of our approach can be summarized simply: conceding early and often (as opposed to little and late, as is typical for independent issue negotiations) is a key to negotiating good complex contracts. Conceding is not individually rational in the face of agents that may choose not to concede, but this problem can be resolved by introducing a mediator that stochastically ignores agent preferences. In both cases, the exchange of tokens when one agent overrides another can be used to incent the truthful voting that enables win-win outcomes.

NEXT STEPS

There are many other promising avenues for future work in this area. The high social welfare achieved by our approach partially reflects the fact that the utility functions for each agent, based as they are solely on binary dependencies, are relatively easy to optimize. Higher-order dependencies, common in many real-world contexts, are known to generate more challenging utility landscapes (Kauffman, 1993). We hypothesize that it may be necessary to adapt non-linear optimization techniques such as genetic algorithms into the negotiation context in order to address this challenge. Another possibility involves agents providing limited information about their utility functions to the mediator or to each other in order to facilitate more intelligent search through very large contract spaces. Agents can, for example, tell the mediator which issues are heavily dependent upon each other, allowing the mediator to focus its attention within tightly coupled issue 'clumps', leaving other less influential issues until later. We hypothesize that agents may be incented to tell the truth about their dependency structures in order to ensure that negotiations can complete in an acceptable amount of time. Finally, we would like to derive formal incentive compatibility proofs (i.e., concerning when agents are incented to vote truthfully) for our protocols. New proof techniques will probably be necessary because previous results in this area have made strong assumptions concerning the shape of the agent utility functions, assumptions that do not hold with complex contracts.

While the negotiation protocol described herein was developed in the context of negotiation among software agents, we argue that it is potentially well-suited for negotiation among humans as well. The fundamental issues raised by the negotiation of complex contracts—multi-optima utility functions and intractably large contract spaces—remain the same irregardless of the nature of the negotiators. The annealing approach to non-linear optimization has been used successfully for a very wide range of problem domains, as it requires only that the utility functions optimized over are ultra-metric—that is, they have the property that the highest optima also tend to be the widest ones. This has shown to be a robust property of networks of interdependent parameters such as those involved in complex contracts (Kauffman, 1993). The annealing protocol requires only that negotiators be able to compare the utility of two contracts. The good results achieved in our simulations therefore suggest that the same protocol may prove helpful for human negotiators. Testing this claim will be an important avenue for future work.

ACKNOWLEDGMENTS

This work was supported by funding from the DARPA Control of Agent-Based Systems (CoABS) program, and the NSF Computation and Social Systems program.

REFERENCES

Axelrod, R. (1984). *The evolution of cooperation.* New York: Basic Books.

Bar-Yam, Y. (1997). *Dynamics of complex systems.* Reading, MA: Addison-Wesley.

Barbuceanu, M., & Lo, W.-K. (2000). A multi-attribute utility theoretic negotiation architecture for electronic commerce. *Proceedings of Autonomous Agents 2000,* Barcelona Spain (pp. 239-246).

Bichler, M., Kaukal, M., & Segev, A. (1999). Multi-attribute auctions for electronic procurement. *Proceedings of the 1st IBM IAC Workshop on Internet-Based Negotiation Technologies,* Yorktown Heights, NY.

Ehtamo, H., Ketteunen, E., & Hamalainen, R. (2001). Searching for joint gains in multi-party negotiations. *European Journal of Operational Research, 1*(30), 54-69.

Faratin, P., Sierra, C., & Jennings, N. R. (2000). Using similarity criteria to make negotiation trade-offs. *Proceedings of the 4th International IEEE Computing Society Conference on Multi-Agent Systems* (pp. 119-126).

Fatima, S. S., Wooldridge, M., & Jennings, N. R. (2002). Multi-issue negotiation under time constraints. *Proceedings of the 2002 International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, Bologna, Italy (pp. 143-150).

Fisher, R., Ury, W., & Patton, B. (Eds.). (1991). *Getting to yes: Negotiating agreement without giving in* (2nd ed.). Boston: Houghton Mifflin.

Jonker, C. M., & Treur, J. (2001). An agent architecture for multi-attribute negotiation. *Proceedings of IJCAI-01* (pp. 1195-1201).

Kalos, M. H., & Whitlock, P.A. (1986). *Monte Carlo methods*. New York: John Wiley & Sons.

Kauffman, S. A. (1993). *The origins of order: Self-organization and selection in evolution.* Oxford, UK: Oxford University Press.

Klein, M., Faratin, P., & Bar-Yam, Y. (2002). Using an annealing mediator to solve the prisoners' dilemma in the negotiation of complex contracts. *Proceedings of the 4th Agent-Mediated Electronic Commerce Workshop (AMEC-IV),* Bologna, Italy (pp. 194-202).

Kowalczyk, R., & Bui, V. (2001). On constraint-based reasoning in e-negotiation agents. In F. Dignum & U. Cortes (Eds.), *Agent-mediated electronic commerce III, current issues in agent-based electronic commerce* (LNCS, pp. 31-46). Berlin: Springer-Verlag.

Oliveira, E., Fonseca, J. M., & Garção, A. S. (1999). Multi-criteria negotiation in multi-agent systems. *Proceedings of the CEEMAS'99 International Conference,* St. Petersburg, Russia.

Raiffa, H. (1982). *The art and science of negotiation.* Cambridge, MA: Belknap Press of Harvard University Press.

Rosenschein, J.S., & Zlotkin, G. (1994). *Rules of encounter: Designing conventions for automated negotiation among computers.* Cambridge, MA: MIT Press.

Ströbel, M. (2000). Framework for electronic negotiations based on adjusted winner mediation. *Proceedings of the DEXA 2000 Workshop on E-Negotiations,* London (pp. 1020).

Chapter XLIX
Agents for Multi-Issue Negotiation

John Debenham
University of Technology, Australia

ABSTRACT

This chapter describes a generic multi-issue negotiating agent that is designed for a dynamic information-rich environment. The agent strives to make informed decisions by observing signals in the marketplace and by observing general information sources including news feeds. The agent assumes that the integrity of some of its information decays with time, and that a negotiation may break down under certain conditions. The agent makes no assumptions about the internals of its opponent—it focuses only on the signals that it receives. Two agents are described. The first agent conducts multi-issue bilateral bargaining. It constructs two probability distributions over the set of all deals: the probability that its opponent will accept a deal, and the probability that a deal should be accepted by the agent. The second agent bids in multi-issue auctions—as for the bargaining agent, this agent constructs probability distributions using entropy-based inference.

INTRODUCTION

This work is based on the assumption that when an intelligent agent buys a hat, a car, a house, or a company, she does so because she feels comfortable with the general terms of the deal. This "feeling of comfort" is achieved as a result of information acquisition and validation. Negotiation is as much of an information acquisition and exchange process as it is an offer exchange process—one feeds off the other.

The generic multi-issue negotiation agent Π draws on ideas from information theory. Game theory (GT) tells us what to do, and what outcome to expect, in many well-known negotiation situations, but these strategies and expectations are derived from assumptions about the internals of the opponent. Game theoretic analyses of bargaining are founded on the notion of agents as utility optimizers in the presence of complete and incomplete information about their opponents (Muthoo, 1999).

Two probability distributions form the foundation of both the offer evaluation and the offer making processes. They are both over the set of all deals and are based on all information available to the agent. The first distribution is the probability that a deal is acceptable to Ω. The second distribution is the probability that a deal will prove to be acceptable to Π—this distribution generalizes the notion of utility.

Π may not have a von Neumann-Morgerstern utility function. Π makes no assumptions about the internals of Ω in particular whether it has a utility function. Π does make assumptions about: the way in which the integrity of information will decay, preferences that its opponent may have for some deals over others, and conditions that may lead to breakdown. It also assumes that unknown probabilities can be inferred using *maximum entropy probabilistic logic* (McKay, 2003) that is based on random worlds (Halpern, 2003). The maximum entropy probability distribution is "the least biased estimate possible on the given information; i.e., it is maximally noncommittal with regard to missing information" (Jaynes, 1957). In the absence of knowledge about Ω's decision-making apparatus, Π assumes that the "maximally noncommittal" model is the correct model on which to base its reasoning.

A *preference relation* is an assumption that Π makes about Ω's preferences for some deals over others—for example, that she prefers to pay a lower price to a higher price. A *single-issue preference relation* assumes that she prefers deals on the basis of one issue alone, independent of the values of the other issues. A preference relation may be assumed prior to the negotiation, or during it based on the offers made. For example, the opponent may display a preference for items of a certain color; Faratin, Sierra, and Jennings (2003) describe a basis for ordering colors. The preference relations illustrated here are single-issue orderings, but the agent's reasoning operates equally well with any preference relation as long as it may be expressed in Horn clause logic.

THE MULTI-ISSUE NEGOTIATION AGENT Π

The integrity of information decays in time. Little appears to be known about how the integrity of information, such as news-feeds, decays. One source of information is the signals received by observing the behavior of the opponent agents both prior to a negotiation and during it. For example, if an opponent bid $8 in an auction for an identical good two days ago, then my belief that she will bid $8 now could be 0.8. When the probability of a decaying belief approaches 0.5, the belief is discarded.

Agent Architecture

Incoming messages from all sources are time-stamped and placed in an "In Box" X as they arrive. Π has a knowledge base K and a belief set B. Each of these two sets contains statements in L. K contains statements that are generally true, such as $\forall x(Accept(x) \leftrightarrow \neg Reject(x))$—that is, an agent does one thing or the other. The belief set $B = \{\beta_i\}$ contains statements that are each qualified with a *given sentence probability* $B\{\beta_i\}$, which represents an agent's belief in the truth of the statement. These sentence probabilities may decay in time.

A *deal* is a pair of commitments $\delta_{\Pi:\Omega}(\pi, \omega)$ between an agent Π and an opponent agent Ω, where π is Π's commitment and ω is Ω's commitment. $D = \{\delta_i\}_{i=1}^{D}$ is the deal set—the set of all possible deals. If the discussion is from the point of view of a particular agent, then the subscript "Π:" may be omitted, and if that agent has just one opponent, the "Ω" may be omitted as well. These commitments may involve multiple issues and not simply a single issue such as

trading price. The set of *terms* T is the set of all possible commitments that could occur in deals in the deal set. An agent may have a real-valued *utility* function $\mathbf{U}: T \rightarrow \Re$ that induces a total ordering on T. For any deal $\delta = (\pi, \omega)$, the expression $\mathbf{U}(\omega) - \mathbf{U}(\pi)$ is called the *surplus* of δ and is denoted by $\mathbf{L}(\delta)$ where $\mathbf{L}: T \times T \rightarrow \Re$. For example, the values of the function **U** may be expressed in units of money. It may not be possible to specify the utility function either precisely or with certainty.[1]

Π has a knowledge base K and a belief set B. Each of these two sets contains statements in a first-order language L. K contains statements that are generally true. The belief set $B = \{\beta_i\}$ contains statements β_i that are each qualified with a *given sentence probability* $B\{\beta_i\}$ that represents the agent's belief in the truth of the statement. The integrity of the statements in B may decay in time. The distinction between the knowledge base K and the belief set B is simply that K contains unqualified statements and B contains statements that are qualified with sentence probabilities.

Π's actions are determined by its "strategy." A *strategy* is a function $\mathbf{S}: K \times B \rightarrow A$ where A is the set of actions. The idea is that at certain distinct times the function **S** is applied to K and B and the agent does something. In between the discrete times at which **S** is activated, information may arrive. Incoming information from all sources is time-stamped and placed in an "In Box" X as it arrives. Then, momentarily before the **S** function is activated, a "revision function" R is activated: $R: (X \times K \times B) \rightarrow (K \times B)$. R clears the "In Box," and updates K and B to ensure consistency. It is not described here.

Maximum Entropy Inference

Π uses maximum entropy inference *ME*. Let G be the set of all positive ground literals that can be constructed using the predicate and function symbols in L.[2] A *possible world* is a valuation function $\mathbf{v}: G \rightarrow \{\top, \bot\}$. That is, a possible world assigns either true ($\top$) or false ($\bot$) to each ground literal in G. **V** denotes the set of all possible worlds, and $\mathbf{V}_K$ denotes the set of possible worlds that are consistent with the agent's knowledge base K (Halpern, 2003).

A *random world* for K is a probability distribution $\mathbf{W}_K = \{p_i\}$ over $\mathbf{V}_K = \{\mathbf{v}_i\}$, where $\mathbf{W}_K$ expresses an agent's degree of belief that each of the possible worlds is the actual world. The *derived sentence probability* of any sentence σ in L, *with respect to* a random world $\mathbf{W}_K$, is:

$$P_{\mathbf{W}_K}(\sigma) \sum_n \{ p_n : \sigma \text{ is } \top \text{ in } \mathbf{v}_n \} \tag{1}$$

That is, we only admit those possible worlds in which σ is true. A random world $\mathbf{W}_K$ is *consistent* with the agent's beliefs B if: $(\forall \beta \in B)(B(\beta) = P_{\mathbf{W}_K}(\beta))$. That is, for each belief its derived sentence probability, as calculated using equation 1, is equal to its given sentence probability.

The *entropy* of a discrete random variable X with probability mass function $\{p_i\}$ is defined in the usual way (McKay, 2003): $H(X) = -\sum_n p_n \log p_n$ where: $p_n \geq 0$ and $\sum_n p_n = 1$. Let $\mathbf{W}_{K,B\}}$ be the "maximum entropy probability distribution over $\mathbf{V}_K$ that is consistent with B." Given an agent with K and B, its *derived sentence probability* for any sentence σ in L is:

$$P(\sigma) = P_{\mathbf{W}_{\{K,B\}}}(\sigma) \tag{2}$$

Using equation 2, the derived sentence probability for any belief β_i is equal to its given sentence probability. So the term *sentence probability* is used from here on without ambiguity. Π uses *maximum entropy inference,* which attaches the derived sentence probability to any given sentence σ.

Maximizing Entropy with Linear Constraints

If X is a discrete random variable taking a finite number of possible values $\{x_i\}$ with probabilities $\{p_i\}$, then the *entropy* is the average uncertainty removed by discovering the true value of X, and is given by $H = -\sum_n p_n \log p_n$. The direct optimization of H subject to a number θ of linear constraints of the form $\sum_n p_n g_k(x_n) = \overline{g}_k$ for given constants $\overline{g}_k$, where $k = 1, \cdots, \theta$, is a difficult problem. Fortunately this problem has the same unique solution as the *maximum likelihood problem* for the Gibbs distribution (Pietra, Pietra, & Lafferty, 1997). The solution to both problems is given by:

$$p_n = \frac{exp(-\sum_{k=1}^{\theta} \lambda_k g_k(x_n))}{\sum_m exp(-\sum_{k=1}^{\theta} \lambda_k g_k(x_m))}, \quad n = 1, 2, \cdots \quad (3)$$

where the constants $\{\lambda_i\}$ may be calculated using equation 3 together with the three sets of constraints: $p_n \geq 0$, $\sum_n p_n = 1$ and $\sum_n p_n g_k(x_n) = \overline{g}_k$.

REPRESENTATION DEPENDENCE

ME is criticized (Halpern, 2003) because the way in which the knowledge is formulated in K and B determines the values derived. This property is promoted here as a strength of the method because the correct formulation of the knowledge base, using the rich expressive power of first-order probabilistic logic, encapsulates features of the application at a fine level of detail.

Price is a common issue in bargaining, auction, and market applications. Two ways of representing price in logic are: to establish a logical constant for each possible price, and to work instead with price intervals. Admitting the possibility of an interval containing just one value, the second generalizes the first. To represent price using price intervals, we have to specify the "width" of each interval. Suppose in an application an item will be sold in excess of $100. Suppose the predicate *TopBid*(Ω, δ) means "δ is the highest price that agent Ω is prepared to bid." This predicate will satisfy: $\forall xy((TopBid(\Omega, x) \wedge TopBid(\Omega, y)) \rightarrow (x = y))$. A crude representation of the set of possible bids is as two logical constants in L: [100, 200) and [200, ∞). There are two positive ground literals in G: *TopBid*(Ω, [100, 200)) and *TopBid*(Ω, [200, ∞)), and there are three possible worlds: $\{(\bot, \bot), (\top, \bot), (\bot, \top)\}$. In the absence of any further information, the maximum entropy distribution is uniform, and for example, the probability that Ω's highest bid ≥ $200 is $\frac{1}{3}$. Now if the set of possible bids had been represented as *three* logical constants, [100, 150], [150, 200), and [200, ∞], then the same probability is $\frac{1}{4}$. Which is correct: $\frac{1}{3}$ or $\frac{1}{4}$? That depends on Π's beliefs about Ω. In both of these examples, by using *ME* and by specifying no further knowledge about *TopBid*(.), we have implicitly asserted that the probability of each possible world being the true world is the same. In the first example all three are $\frac{1}{3}$, and in the second all four are $\frac{1}{4}$. This is what happens when the "maximally noncommittal" distribution is chosen. Conversely, if believe that $\forall x, y(P(TopBid(\Omega, x)) = P(TopBid(\Omega, y)))$, then it is not necessary to include this in K—it is implicitly present and we should appreciate that it is so.

Following from the previous paragraph with just two logical constants, suppose the predicate *MayBid*(Ω, δ) means "Ω is prepared to make a bid of δ." Assuming the Ω will prefer to pay less than more, this predicate will satisfy: $\kappa_1 : \forall x, y((MayBid(\Omega, x) \wedge (x \geq y)) \rightarrow MayBid(\Omega, x))$, where x and y are intervals and the meaning of "≥" is obvious. With just κ_1 in K, there are three possible worlds:

$\{(\perp,\perp),(\top,\perp),(\top,\top)\}$. The maximum entropy distribution is uniform, and $P(MayBid(\Omega,[100,200))) = \frac{2}{3}$ and $P(MayBid(\Omega,[200,\infty])) = \frac{1}{3}$. With no additional information, $P(TopBid(\Omega, x))$ will be uniform and $P(MayBid(\Omega, x))$ will be linear decreasing in x.

The conclusion to be drawn from the previous two paragraphs is that when an issue is represented using intervals, there is no "right" or "wrong" choice of intervals. However, choosing the intervals so that the expected probability distribution of at least one key predicate is uniform over those intervals may simplify K and B.

In *GT* each agent models its opponents by speculating on their type. Beliefs concerning an opponent's type may be represented as a probability distribution over its expected utility. In *ME* the relative truth of possible worlds are determined by statements in first-order probabilistic logic that represent beliefs concerning the opponents' behavior. So *ME* models its opponents in a fundamentally different way to *GT* and uses a richer language to do so.

AGENTS FOR BARGAINING

The generic multi-issue negotiation agent Π engages in bilateral bargaining with an opponent Ω. It strives to make informed decisions in an information-rich environment that includes information drawn from the Internet by bots. Its design was provoked by the observation that agents are not always utility optimizers. Π attempts to fuse the negotiation with the information generated both by and because of it. It reacts to information derived from its opponent and from the environment, and proactively seeks missing information that may be of value.

One source of Π's information is the signals received from Ω. These include offers to Π, and the acceptance or rejection of Π's offers. If Ω rejected Π's offer of $8 two days ago, then what is Π's belief now in the proposition that Ω will accept another offer of $8 now? Perhaps it is around 0.1. A linear model is used to model the integrity decay of these beliefs, and when the probability of a decaying belief approaches 0.5,[3] the belief is discarded. This choice of a linear model is independent of the bargaining method. The model of decay could be exponential, quadratic, or whatever.

Under some circumstances bilateral bargaining has questionable value as a trading mechanism. Bilateral bargaining is known to be inherently inefficient (Myerson & Satterthwaire, 1983). Bulow and Klemperer (1996) show that a seller is better off with an auction that attracts $n + 1$ buyers than bargaining with n individuals, *no matter what* the bargaining protocol is. Neeman and Vulkan (2000) show that the weaker bargaining types will fare better in exchanges leading to a gradual migration. These results hold for agents who aim to optimize their utility and do limit the work described here.

Interaction Protocol

The agents communicate using sentences in a first-order language L. This includes the exchange, acceptance, and rejection of offers. L contains the following predicates: *Offer*(δ), *Accept*(δ), *Reject*(δ), and *Quit*(.), where *Offer*(δ) means "the sender is offering you a deal δ," *Accept*(δ) means "the sender accepts your deal δ," *Reject*(δ) means "the sender rejects your deal δ," and *Quit*(.) means "the sender quits—the negotiation ends."

Two negotiation protocols are described: first, negotiation *without decay* in which all offers stand for the entire negotiation; and second, negotiation *with decay* in which offers stand only if accepted by return—Π represents Ω's offers as beliefs with sentence probabilities that decay in time.

Π and Ω each exchange offers alternately at successive discrete times (Kraus, 2001). They enter into a commitment if one of them accepts a standing offer. The protocol has three stages:

1. simultaneous, initial, binding offers from both agents;
2. a sequence of alternating offers; and
3. an agent quits and walks away from the negotiation.

The negotiation ceases *either* in the second round if one of the agents accepts a standing offer *or* in the final round if one agent quits and the negotiation breaks down.

In the first stage the agents simultaneously send *Offer*(.) messages to each other. These initial offers are taken as limits on the range of values that are considered possible. This is crucial to the maximum entropy method when there are domains that would otherwise be unbounded. The exchange of initial offers "stakes out the turf" on which the subsequent negotiation will take place. In the second stage an *Offer*(.) message is interpreted as an implicit rejection *Reject*(.) of the opponent's offer on the table.

Π uses three things to make offers: an estimate of the likelihood that Ω will accept any offer; an estimate of the likelihood that Π will, in hindsight, feel comfortable accepting any particular offer; and an estimate of when Ω may quit and leave the negotiation.

ESTIMATING $P(\Omega Acc(.))$

Π does two different things: first, it reacts to offers received from Ω; second, it sends offers to Ω. This section describes the estimation of $P(\Omega Acc(\delta))$ where the predicate $\Omega Acc(\delta)$ means "the deal δ is acceptable to Ω."

When a negotiation commences Π may have no information about Ω or about prior deals. If so then the initial offers may only be based on past experience or circumstantial information.[4] So the opening offers are simply taken as given.

In the four sub-sections following, Π is attempting sell something to Ω. In the previous work, Π's terms τ are to supply a particular good, and Ω's terms ω are money—in those examples the amount of money ω is the subject of the negotiation. In the following work, Π's terms are to supply a particular good together with some negotiated warranty period, and Ω's terms are money—in those examples the amount of money p and the period of the warranty period w are the subject of the negotiation.

One Issue: Without Decay

The unary predicate $\Omega Acc(x)$ means "the amount of money $\$x$ is acceptable to Ω." Π is interested in whether the unary predicate $\Omega Acc(x)$ is true for various values of $\$x$. Π assumes the following preference relation on the $\Omega Acc(\cdot)$ predicate:

$$\kappa_1 : \forall x, y((x > y) \rightarrow (\Omega Acc(x) \rightarrow \Omega Acc(y)))$$

Suppose that Π's opening offer is $\overline{\omega}$, and Ω's opening offer is $\underline{\omega}$ where $\underline{\omega} < \overline{\omega}$. Then K now contains two further sentences: $\kappa_2 : \neg\Omega Acc(\overline{\omega})$ and $\kappa_3 : \Omega Acc(\underline{\omega})$. There are now $\overline{\omega} - \underline{\omega}$ possible worlds, and the maximum entropy distribution is uniform.

Suppose that Π knows its true valuation for the good u_Π, and that Π has decided to make an "expected-utility-optimizing" offer: $x = \frac{\overline{\omega} + u_\Pi}{2}$. This offer is calculated on the basis of the preference ordering κ_1 and the two signals that Π has received from Ω. The response is in terms of only Π's valuation u_Π and the signal

$Reject(\overline{\omega})$—it is independent of the signal $Offer(\underline{\omega})$, which implies that $\underline{\omega}$ is acceptable.

In the standard game theoretic analysis of bargaining (Muthoo, 1999), Π assumes that Ω has a utility u_Ω, that it lies in some interval $[\underline{u},\overline{u}]$, and that the expected value of u_Ω is uniformly distributed on that interval. On the basis of these assumptions, Π then derives the expected-utility-optimizing offer: $\frac{\overline{u}+u_\Pi}{2}$. These two offers differ by $\overline{u}$ in the game-theoretic result and $\overline{\omega}$ in the maximum entropy result. The game theoretic approach relies on estimates for $\underline{u}$ and $\overline{u}$:

$$E([\underline{u},\overline{u}] \mid Reject(\overline{\omega}) \wedge Accept(\underline{\omega}))$$

If Ω has a utility, and it may not, then if Ω is rational: $\underline{u} \le \underline{\omega} \le \overline{u}$. The inherent inefficiency of bilateral bargaining (Myerson & Satterthwaite, 1983) shows for an economically rational Ω that u_Ω, and so consequently $\overline{u}$, may be greater than $\overline{\omega}$. There is no reason to suspect that $\overline{u}$ and $\overline{\omega}$ will be equal.

One Issue: With Decay

As in the previous example, suppose that the opening offers at time t_0 are taken as given and are $\underline{\omega}$ and $\overline{\omega}$. Then K contains κ_1, κ_2, and κ_3. Suppose L contains n consecutive integer constants in the interval $[\underline{\omega},\overline{\omega}]$, where $n = \overline{\omega} - \underline{\omega} + 1$, that represent various amounts of money. κ_1 induces a total ordering on the sentence probabilities for $\Omega Acc(x)$ on the interval $[\underline{\omega},\overline{\omega}]$, where the probabilities are ≈ 0 at $\overline{\omega}$ and ≈ 1 at $\underline{a}$.

Suppose that at time t_1, Π makes an offer ω_{na} which is rejected by Ω, who has replied at time t_2 with an offer of ω_{op} where $\underline{\omega} \le \omega_{op} \le \omega_{na} \le \overline{\omega}$. At time t_3, B contains β_1 : $\Omega Acc(\omega_{na})$ and β_2 : $\Omega Acc(\omega_{op})$. Suppose that there is some level of integrity decay on these two beliefs: $0 < B(\beta_1) < 0.5 < B(\beta_2) < 1$. Then $\mathbf{V}_K$ contains $n+1$ possible worlds ranging from "all false" to "all true," each containing n literals. So a random world for K will consist of $n + 1$ probabilities $\{p_i\}$, where, say, p_1 is the probability of "all true," and p_{n+1} is the probability of "all false". $P_{\{K,B\}}$ will be the distribution that maximizes $-\sum_n p_n \log p_n$ subject to the constraints:

$$p_n \ge 0, \sum_n p_n = 1, \sum_{n=1}^{\overline{\omega}-\omega_{na}+1} p_n = B(\beta_1)$$

and

$$\sum_{n=1}^{\overline{\omega}-\omega_{op}+1} p_n = B(\beta_2).$$

The optimization of entropy H, subject to linear constraints, is described above. $P_{\{K,B\}}$ is:

$$p_n = \begin{cases} \frac{B(\beta_1)}{\overline{\omega}-\omega_{na}+1} \text{ if } 1 \le n \le \overline{\omega}-\omega_{na}+1 \\ \frac{B(\beta_2)-B(\beta_1)}{\omega_{na}-\omega_{op}} \text{ if } \overline{\omega}-\omega_{na}+1 < n < \overline{\omega}-\omega_{op}+2 \\ \frac{1-B(\beta_2)}{\omega_{op}-\underline{\omega}+1} \text{ if } \overline{\omega}-\omega_{op}+2 \le n \le \overline{\omega}-\underline{\omega}+2 \end{cases}$$

Using equation 2, for $\omega_{op} \le x \le \omega_{na}$:

$$P(\Omega Acc(x)) = B(\beta_1) + \frac{\omega_{na}-x}{\omega_{na}-\omega_{op}}(B(\beta_2)-B(\beta_1)) \quad (4)$$

These probability estimates are used by the negotiation strategy **S** to calculate Π's next offer.

The values for $P(\Omega Acc(x))$ in the region $\omega_{op} \le x \le \omega_{na}$ are derived from only two pieces of information that are the two signals $Reject(\omega_{na})$ and $Offer(\omega_{op})$, each qualified with the time at which they arrived and the decay rate on their integrity. The assumptions in the analysis given above are: the choice of values for $\underline{\omega}$ and $\overline{\omega}$—which do not appear in equation 4 in any case—and the choice of the "maximally noncommittal" distribution.

If the agents continue to exchange offers, then new beliefs will be acquired and the integ-

rity of old beliefs will decay. If the next pair of offers lies within the interval $[\omega_{op},\omega_{na}]$ and if the integrity of β_1 and β_2 decays, then the sentence probabilities of β_1 and β_2 will be inconsistent with those of the two new beliefs due to the total ordering of sentence probabilities on $[\underline{\omega},\overline{\omega}]$ induced by κ_1. This inconsistency is resolved by the revision function *R* that here discards inconsistent older beliefs β_1 and β_2 in favor of more recent beliefs. If the agents continue in this way, then the sentence probabilities for the $\Omega Acc(\cdot)$ predicate are given simply by equation 4 using the most recent values for ω_{na} and ω_{op}.

The analysis given above requires that values be specified for the opening offers $\underline{\omega}$ and $\overline{\omega}$. The only part of the probability distribution that depends on the values chosen for $\underline{\omega}$ and $\overline{\omega}$ are the two "tails" of the distribution. So the choice of values for these two opening offers is unlikely to affect the estimates. The two tails are necessary to "soak up" the otherwise unallocated probability.

Two Issues: Without Decay

The above approach to single-issue bargaining generalizes without modification to multi-issue bargaining; it is illustrated with two issues only for ease of presentation. The problem considered is the sale of an item with 0,...,4 years of warranty. The terms being negotiated specify an amount of money *p* and the number of years warranty *w*. The predicate $\Omega Acc(w, p)$ now means "Ω will accept the offer to purchase the good with *w* years warranty for $*p*."

Π assumes the following two preference orderings, and *K* contains:

$$\kappa_{11} : \forall x, y, z((x > y) \rightarrow (\Omega Acc(y, z) \rightarrow \Omega Acc(x, z)))$$
$$\kappa_{12} : \forall x, y, z((x > y) \rightarrow (\Omega Acc(z, x) \rightarrow \Omega Acc(z, y)))$$

As previously, these sentences conveniently reduce the number of possible worlds. The number of possible worlds will be finite as long as K contains two statements of the form: $\neg\Omega Acc(4, a)$ and $\Omega Acc(0, b)$ for some a and b. Suppose that Π's initial offer was "4 years warranty for $21" and Ω's initial offer was "no warranty for $10." *K* now contains:

$$\kappa_{13} : \neg\Omega Acc(4,21) \quad \kappa_{14} : \Omega Acc(0,10)$$

These two statements, together with the restriction to integers only, limit the possible values of *w* and *p* in $\Omega Acc(w, p)$ to a 5×10 matrix.

Suppose that Π knows its utility function for the good with 0,…,4 years warranty and that its values are: $11.00, $11.50, $12.00, $13.00, and $14.50 respectively. Suppose that Π uses the strategy $\mathbf{S}^{(n)}$ which is described below—the details of that strategy are not important now. If Π uses that strategy with $n = 2$, then Π offers $Offer(2, \$16)$ which suppose Ω rejects and counters with $Offer(1, \$11)$. Then with $n = 2$ again, Π offers $Offer(2, \$14)$ which suppose Ω rejects and counters with $Offer(3, \$13)$. $P(\Omega Acc(w, p))$ now is:

	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$
$p=20$	0.0000	0.0000	0.0000	0.0455	0.0909
$p=19$	0.0000	0.0000	0.0000	0.0909	0.1818
$p=18$	0.0000	0.0000	0.0000	0.1364	0.2727
$p=17$	0.0000	0.0000	0.0000	0.1818	0.3636
$p=16$	0.0000	0.0000	0.0000	0.2273	0.4545
$p=15$	0.0000	0.0000	0.0000	0.2727	0.5454
$p=14$	0.0000	0.0000	0.0000	0.3182	0.6364
$p=13$	0.0455	0.0909	0.1364	1.0000	1.0000
$p=12$	0.0909	0.1818	0.2727	1.0000	1.0000
$p=11$	0.1364	1.0000	1.0000	1.0000	1.0000

and the expected-utility-optimizing offer is: $Offer(4, \$18)$. If Π makes that offer, then the expected surplus is $0.95. The matrix above contains the "maximally non-committal" values for $P(\Omega Acc(w, p))$; those values are recalculated each time a signal arrives. The example

demonstrates how the Π is able to conduct multi-issue bargaining in a focused way without making assumptions about Ω's internals, in particular whether Ω is aware of a utility function (Osborne & Rubinstein, 1990).

Two Issues: With Decay

Following from the previous section, suppose that K contains κ_{11}, κ_{12}, κ_{13}, and κ_{14}. The two preference orderings κ_{11} and κ_{12} induce a partial ordering on the sentence probabilities in the $P(\Omega Acc(w, p))$ array from the top-left where the probabilities are ≈ 0, to the bottom-right where the probabilities are ≈ 1. There are 51 possible worlds that are consistent with K.

Suppose that b contains: β_{11} : $\Omega Acc(2,16)$, β_{11} : $\Omega Acc(2,16)$, β_{12} : $\Omega Acc(2,14)$, β_{13} : $\Omega Acc(1,11)$, and β_{14} : $\Omega Acc(3,13)$ and with a 10% decay in integrity for each time step: $P(\beta_{11}) = 0.4$, $P(\beta_{12}) = 0.2$, $P(\beta_{13}) = 0.7$, and $P(\beta_{14}) = 0.9$. Belief β_{11} is inconsistent with $K \cup = \{\beta_{12}\}$, as together they violate the sentence probability ordering induced by κ_{11} and κ_{12}. Resolving this issue is a job for the belief revision function R which discards the older, and weaker, belief β_{11}.

Equation 3 is used to calculate the distribution $\mathbf{W}_{\{\mathbf{K}, \mathbf{B}\}}$ which has just five different probabilities in it. The resulting values for the three λ's are: λ_{12}=2.8063, λ_{13}=2.0573, and λ_{14}=2.5763. $P(\Omega Acc(w, p))$ now is:

	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$
$p=20$	0.0134	0.0269	0.0286	0.0570	0.0591
$p=19$	0.0269	0.0537	0.0571	0.1139	0.1183
$p=18$	0.0403	0.0806	0.0857	0.1709	0.1774
$p=17$	0.0537	0.1074	0.1143	0.2279	0.2365
$p=16$	0.0671	0.1343	0.1429	0.2849	0.2957
$p=15$	0.0806	0.1611	0.1714	0.3418	0.3548
$p=14$	0.0940	0.1880	0.2000	0.3988	0.4139
$p=13$	0.3162	0.6324	0.6728	0.9000	0.9173
$p=12$	0.3331	0.6662	0.7088	0.9381	0.9576
$p=11$	0.3500	0.7000	0.7447	0.9762	0.9978

In this array, the derived sentence probabilities for the three sentences in B are shown in bold type; they are exactly their given values.

ESTIMATING $P(\Pi Acc(.))$

The proposition $\Pi Acc(\delta)$ means: "δ is acceptable to Π." This section describes how Π attaches a conditional probability to the proposition $P(\Pi Acc(\delta) \mid I_t)$ in light of information I_t. The meaning of "acceptable to Π" is described below. This is intended to put Π in the position "looking back on it, I made the right decision at the time"—this is a vague notion but makes sense to the author. The idea is for Π to accept a deal δ when $P(\Pi Acc(\delta) \mid I_t) \geq \alpha$ for some threshold value α that is one of Π's mental states.

$P(\Pi Acc(\delta) \mid I_t)$ is derived from conditional probabilities attached to four other propositions:

$$P(Suited(\omega) \mid I_t),$$

$$P(Good(OP) \mid I_t),$$

$$P(Fair(\delta) \mid I_t \cup \{Suited(\omega), Good(OP)\}),$$

and

$$P(Me(\delta) \mid I_t \cup \{Suited(\omega), Good(OP)\}),$$

meaning respectively: "terms ω are perfectly suited to my needs," "Ω will be a good agent for me to be doing business with," "δ is generally considered to be a good deal for Π," and "on strictly subjective grounds, δ is acceptable to Π." The last two of these four probabilities factor out both the suitability of ω and the appropriateness of the opponent Ω. The difference between the third and fourth is that the third captures the concept of "a good market

deal" and the fourth a strictly subjective "what ω is worth to *NA*." The "*Me*(.)" proposition is related to the concept of a private valuation in game theory.

To determine $P(Suited(\omega) \mid I_t)$: if there are sufficiently strong preference relations to establish extrema for this distribution, then they may be assigned extreme values ≈0.0 or 1.0. Π is repeatedly asked to provide probability estimates for the offer ω that yields the greatest reduction in entropy for the resulting distribution (McKay, 2003). This continues until Π considers the distribution to be "satisfactory." This is tedious, but the "preference acquisition bottleneck" appears to be an inherently costly business (Castro-Schez, Jennings, Luo, & Shadbolt, 2004).

To determine $P(Good(OP) \mid I_t)$ involves an assessment of the reliability of Ω. For some retailers (sellers), information—of varying reliability—may be extracted from sites that rate them. For individuals, this may be done either through assessing their reputation established during prior trades (Ramchurn, Jennings, Sierra, & Godo, 2003) or by the inclusion of some third-party escrow service that is then rated for "reliability" instead.

$P(Fair(\delta) \mid I_t \cup \{Suited(\omega), Good(OP)\})$ is determined by market data. As for dealing with *Suited,* if the preference relations establish extrema for this distribution, then extreme values may be assigned. Independently of this, real market data, qualified with given sentence probabilities, is fed into the distribution. The revision function *R* identifies and removes inconsistencies, and missing values are estimated using the maximum entropy distribution.

Determining $P(Me(\delta) \mid I_t \cup \{Suited(\omega), Good(OP)\})$ is a subjective matter. It is specified using the same device as used for *Fair,* except that the data is fed in by hand "until the distribution appears satisfactory." To start this process, first identify those δ that "Π would never accept"—they are given a probability of ≈0.0; and second those δ that "Π would be delighted to accept"—they are given a probability of ≈1.0. The *Me* proposition links the information-theory approach with "private valuations" in game-theory.

There is no causal relationship between the four probability distributions as they have been defined, with the possible exception of the third and fourth. To link the probabilities associated with the five propositions, the probabilities are treated as epistemic probabilities and the nodes form a simple Bayesian net. The weights on the four arcs of the Bayesian net are a subjective representation of what "acceptable" means to Π. The resulting net divides the problem of estimating $P(\Pi Acc(.))$ into four simpler subproblems.

The conditionals on the Bayesian network are subjective—they are easy to specify because 12 of them are zero—that is, for the cases in which Π believes that either *Me* or *Suited* is "false." For example, if the conditionals (set by Π) are:

$$P(\Pi Acc \mid Me, Suited, Good, Fair) = 1.0$$

$$P(\Pi Acc \mid Me, Suited, \neg Good, Fair) = 0.1$$

$$P(\Pi Acc \mid Me, Suited, Good, \neg Fair) = 0.4$$

$$P(\Pi Acc \mid Me, Suited, \neg Good, \neg Fair) = 0.05$$

then, with probabilities of 0.9 on each of the four evidence nodes, the probability $P(\Pi Acc) = 0.75$. It then remains to manage the acquisition of information I_t from the available sources to, if necessary, increase $P(\Pi Acc(\delta) \mid I_t)$ so that δ is acceptable. The conditional probabilities on the net represent an agent's priorities for a deal, and so they are specified for each class of deal.

The $\Pi Acc(.)$ predicate generalizes the notion of utility. Suppose that Π knows its utility

function **U**. If the conditionals on the Bayesian net are as in the previous paragraph and if either $P(Me(.))$ or $P(Suited(.))$ are zero, then $P(\Pi Acc(\cdot))$ will be zero. If the conditional probabilities on the Bayesian net are 1.0 when *Me* is true and are 0.0 otherwise, then $P(\Pi Acc)=P(Me)$. Then define:

$$P(Me(\tau,\omega)) = \frac{1}{2}\times(1+\frac{\mathbf{U}(\omega)-\mathbf{U}(\tau)}{\mathbf{U}(\overline{\omega})-\mathbf{U}(\tau)})$$

for $\mathbf{U}(\omega) > \mathbf{U}(\tau)$ and zero otherwise, where $\overline{\omega} = \max_{\omega} \mathbf{U}(\omega)$.[5] A bargaining threshold $\alpha > 0.5$ will then accept offers for which the surplus is positive. In this way $\Pi Acc(\cdot)$ represents utility-based bargaining with a private valuation.

$\Pi Acc(\cdot)$ also is intended to be able to represent apparently irrational bargaining situations (e.g., "I've just *got* to have that hat"), as well as tricky multi-issue problems such as those typical in e-procurement. It enables an agent to balance the degree of suitability of the terms offered with the reliability of the opponent and with the fairness of the deal.

NEGOTIATION STRATEGIES

We have estimated the probability distribution $P(\Omega Acc(\cdot))$ that Ω will accept an offer, and the probability distribution $P(\Pi Acc(\cdot))$ that Π should be prepared to accept an offer. These two probability distributions represent the opposing interests of the two agents Π and Ω. $P(\Omega Acc(\cdot))$ will change every time an offer is made, rejected, or accepted. $P(\Pi Acc(\cdot))$ will change as the background information changes. This section discusses Π's strategy **S**.

Bargaining can be a game of bluff and counter-bluff in which an agent may even not intend to close the deal if one should be reached. A basic conundrum in any offer-exchange bargaining is: it is impossible to force your opponent to reveal information about their position without revealing information about your own position. Further, by revealing information about your own position, you may change your opponent's position—and so on.[6] This infinite regress, of speculation and counter-speculation, is avoided here by ignoring the internals of the opponent and by focusing on what is known for certain—that is, *what* information is contained in the signals received and *when* those signals arrived.

A fundamental principle of competitive bargaining is "never reveal your best price," and another is "never reveal your deadline—if you have one" (Sandholm & Vulkan, 1999). It is not possible to be prescriptive about what an agent *should* reveal. All that can be achieved is to provide strategies that an agent may choose to employ. The following are examples of such strategies.

Without Breakdown

An agent's strategy **S** is a function of the information I_t that it has at time t. That information will be represented in the agent's K and B, and will have been used to calculate $P(\Omega Acc(\cdot))$ and $P(\Pi Acc(\cdot))$. Simple strategies choose an offer only on the basis of $P(\Omega Acc(\cdot))$, $P(\Pi Acc(\cdot))$, and α. The greedy strategy $\mathbf{S}^{+}$ chooses:

$$\arg\max_{\delta}\{P(\Pi Acc(\delta)) \mid P(\Omega Acc(\delta)) \gg 0\};$$

it is appropriate for an agent that believes Ω is desperate to trade. The *expected-acceptability-to-NA-optimizing strategy* $\mathbf{S}^{*}$ chooses:

$$\arg\max_{\delta}\{P(\Omega Acc(\delta))\times P(\Pi Acc(\delta)) \mid P(\Pi Acc(\delta)) \geq \alpha\}$$

it is appropriate for a confident agent that is not desperate to trade. The strategy $\mathbf{S}^{-}$ chooses:

$$\arg\max_{\delta}\{P(\Omega Acc(\delta)) \mid P(\Pi Acc(\delta)) \geq \alpha\};$$

it optimizes the likelihood of trade—it is a good strategy for an agent that is keen to trade without compromising its own standards of acceptability.

An approach to issue-tradeoffs is described in Faratin, Sierra, and Jennings (2003). The bargaining strategy described there attempts to make an acceptable offer by "walking round" the iso-curve of Π's previous offer (that has, say, an acceptability of $\alpha_{na} \geq \alpha$) towards Ω's subsequent counter-offer. In terms of the machinery described here, an analogue is to use the strategy $\mathbf{S}^{-}$:

$$\arg\max_{\delta}\{ P(\Omega Acc(\delta)) \mid P(\Pi Acc(\delta) \mid I_t)\alpha_{na} \}$$

for $\alpha = \alpha_{na}$. This is reasonable for an agent that is attempting to be accommodating without compromising its own interests. Presumably such an agent will have a policy for reducing the value α_{na} if her deals fail to be accepted. The complexity of the strategy in Faratin, Sierra, and Jennings (2003) is linear with the number of issues. The strategy described here does not have that property, but it benefits from using $P(\Omega Acc)$, which contains footprints of the prior offer sequence in that distribution; more recent offers have stronger weights.

With Breakdown

A negotiation may break down because one agent is not prepared to continue for some reason. $\mathbf{p}_B$ is the probability that the opponent will quit the negotiation in the next round. There are three ways in which Π models the risk of *breakdown*. First, $\mathbf{p}_B$ is a constant determined exogenously to the negotiation, in which case at any stage in a continuing negotiation, the expected number of rounds until breakdown occurs is $\frac{1}{P_B}$. Second, $\mathbf{p}_B$ is a monotonic increasing function of time—this attempts to model an impatient opponent. Third, $\mathbf{p}_B$ is a monotonic increasing function of $(1 - \Omega Acc(\delta))$—this attempts to model an opponent who will react to unattractive offers.

At any stage in a negotiation, Π may be prepared to gamble on the expectation that Ω will remain in the game for the next n rounds. This would occur if there is a constant probability of breakdown $P_B = \frac{1}{n}$. Let I_t denote the information stored in Π's K and B at time t. $\mathbf{S}$ is Π's strategy. If Π offered to trade with Ω at $\mathbf{S}(I_1)$, then Ω may accept this offer, but may have also been prepared to settle for terms more favorable than this to Π. If Π offered to trade at $\mathbf{S}(I_1 \cup \{Accept(\mathbf{S}(I_1))\})$, then Ω will either accept this offer or reject it. In the former case trade occurs at more favorable terms than $\mathbf{S}(I_1)$, and in the latter case a useful piece of information has been acquired: $Reject\ Reject(\mathbf{S}(I_1))$ which is added to I_1 before calculating the next offer. This process can be applied twice to generate the offer $\mathbf{S}(I_1 \cup \{\neg Accept(\mathbf{S}(I_1 \cup \{\neg Accept(\mathbf{S}(I_1))\}))\})$, or any number of times, optimistically working backwards on the assumption that Ω will remain in the game for n rounds. The strategy is $\mathbf{S}^{(n)}$, where $\mathbf{S}^{(1)} = \mathbf{S}^{*}$—the expected-acceptability-to-Π-optimizing strategy. $\mathbf{S}^{(n)}$ is the strategy of working back from $\mathbf{S}^{(1)}$ $(n-1))$ times. At each stage $\mathbf{S}^{(n)}$ will benefit also from the information in the intervening counter-offers presented by Ω. The strategy $\mathbf{S}^{(n)}$ is reasonable for a risk-taking, expected-acceptability-optimizing agent.

Define the value of making an offer $Offer(\delta)$ to be: $\Upsilon(Offer(\delta)) = P(\Pi Acc(\delta))$ if δ is accepted, and zero otherwise. The *expected value* of making an offer is then:

$$E(\Upsilon(Offer(\mathbf{S}(I_t)))) =$$
$$P(\Omega Acc(\mathbf{S}(I_t))) \times P(\Pi Acc(\mathbf{S}(I_t))) +$$

$$(1 - P(\Omega Acc(\mathbf{S}(I_t)))) \times (1 - P_B) \times E(\Upsilon(Offer(\mathbf{S}(I_{t+1}))))$$

where $I_{t+1} = I_t \cup \{\neg Accept(\mathbf{S}(I_t))\}$. This is of little help in finding the "best" $\mathbf{S}$, but two approximations

are interesting. Either replace the $\mathbf{S}$ in the final term by a simple strategy such as $\mathbf{S}^-$. Or assume that $E(\Upsilon(Offer(\mathbf{S}(I_{t+1})))) = \theta \times E(\Upsilon(Offer(\mathbf{S}(I_t))))$—for some $\theta < 1$—then:

$$E(\Upsilon(Offer(\mathbf{S}(I_t)))) = \frac{P(\Omega Acc(\mathbf{S}(I_t))) \times P(\Pi Acc(\mathbf{S}(I_t)))}{1-(1-P(\Omega Acc(\mathbf{S}(I_t)))) \times (1-P_B) \times \theta}$$

In either case the expression can be optimized numerically, even if P_B is a function of $P(\Omega Acc(\mathbf{S}(I_t)))$.

The preceding considers the possibility of Ω quitting. Π may choose to quit and cause the negotiation to break down if the negotiation appears to be leading nowhere. One measure of convergence is to monitor the sequence: $\max_\delta (P(\Omega Acc(\delta)) \mid P(\Pi Acc(\delta)) \geq \alpha)$—that is, the greatest likelihood of acceptable trade. If this sequence is not increasing in time to a "reasonable" value, then Π may choose to quit.

AGENTS FOR AUCTIONS AND BIDDING

The form of negotiation considered is between bidding agents and an auctioneer Υ in an information-rich environment. The agent described here is called the *Bidding Agent*, or P; it engages in auctions with a set of S opponents $\{\Omega_1,\cdots,\Omega_S\}$. General information is extracted from the World Wide Web using special purpose 'bots' that import and continually confirm information that is then represented in prespecified predicates. P receives information by observing its opponents $\{\Omega_i\}$ and from these bots.

Game theory, dating back to the work of John von Neumann and Oscar Morgenstern, provides the basis for the analysis of auctions and bidding. There is a wealth of material in this analysis (Klemperer, 2000) originating with the work of William Vickrey. Fundamental to this analysis is the central role of the utility function, and the notion of rational behavior by which an agent aims to optimize its utility, when it is able to do so, and to optimize its expected utility otherwise. Analyses that are so founded on game theory are collectively referred to as *game theoretic,* or *GT*.

The application of *GT* to the design of auction mechanisms has been both fruitful and impressive—rational behavior provides a theoretical framework in which mechanism performance may be analyzed. A notable example being the supremely elegant *Generalized Vickrey* mechanism (Varian, 1995). *GT* also leads to prescriptive results concerning agent behavior, such as the behavior of agents in the presence of hard deadlines (Sandholm & Vulkan, 1999). The general value of *GT* as a foundation for a prescriptive theory of agent behavior is limited both by the extent to which an agent knows its own utility function, and by its certainty in the probability distributions of the utility functions (or, *types*) of its opponents.

In some negotiations—such as when an agent buys a hat, a car, a house, or a company—she may not know her utility with certainty. Nor may she be aiming to optimize anything—she may simply want to buy it. Further, she may be even less certain of her opponents' types or whether her opponents are even aware of the concept of utility. In such negotiations, an agent may be more driven towards establishing a feeling of personal "comfort" through a process of information acquisition, than by a desire to optimize an uncertain personal utility function.[7]

A negotiation agent Π attempts to fuse the negotiation with the information that is generated both by and because of it. Π decides what to do—such as whether to bid in an auction—on the basis of information that may be qualified by expressions of degrees of belief. Π uses this information to calculate and continually re-

calculate probability distributions for that which it does not know. One such distribution, over the set of all possible deals, expresses Π's belief in the acceptability of a deal. Other distributions attempt to predict the behavior of its opponents—such as what they might bid in an auction. These distributions are calculated from Π's knowledge and beliefs using maximum entropy inference. Π makes no assumptions about the internals of its opponents, including whether they have or are even aware of the concept of utility functions. Π is purely concerned with its opponents' behavior—what they do—and not with assumptions about their motivations. The agents communicate using the following predicate: $Bid(.)$, where $Bid(\delta)$ means "the sender bids a deal δ." The set of actions A is limited to sending $Bid(.)$ messages to the auctioneer ϒ.

An Application

An exemplar application used follows. It concerns the purchase of a particular second-hand motor vehicle, with some period of warranty, for cash. So the two issues in this negotiation are: the period of the warranty, and the cash consideration. The meaning of the predicate *MayBid*(Ω, δ) is unchanged, but δ now consists of a pair of issues and the deal set has no natural ordering. Suppose that P wishes to apply *ME* to estimate values for: $P(MayBid(\Omega,\delta))$ for various δ. Suppose that the warranty period is simply 0,···,4 years, and that the cash amount for this car will certainly be at least $5,000 with no warranty, and is unlikely to be more than $7,000 with a four-year warranty. In what follows all price units are in thousands of dollars. Suppose then that the deal set in this application consists of 55 individual deals in the form of pairs of warranty periods and price intervals: { (*w*, [5.0, 5.2)), (*w*, [5.2, 5.4)), (*w* , [5.4, 5.6)), (*w*, [5.6, 5.8), (*w*, [5.8, 6.0)), (*w* , [6.0, 6.2)), (*w*, [6.2, 6.4)), (*w*, [6.4, 6.6)), (*w* , [6.6, 6.8)), (*w*, [6.8, 7.0)), (*w*, [7.0, ∞)) }, where $w = 0,\cdots,4$. Suppose that P has received intelligence that agent Ω is prepared to bid 6.0 with no warranty, and to bid 6.9 with a one-year warranty, and P believes this with probability 0.8. Then this leads to two beliefs: $\beta_1 : TopBid(0,[6.0,6.2)); B(\beta_1) = 0.8$, $\beta_2 : TopBid(1,[6.8,7.0)); B(\beta_2) = 0.8$. Following the discussion above, before "switching on" *ME*, P should consider whether it believes that $P(MayBid(\Omega,\delta))$ is uniform over δ. If it does, then it includes both β_1 and β_2 in B, and calculates $\mathbf{W}_{\{K,B\}}$ that yields estimates for $P(MayBid(\Omega,\delta))$ for all δ. If it does not, then it should include further knowledge in K and B. For example, P may believe that Ω is more likely to bid for a greater warranty period the higher her bid price. If so, then this is a multi-issue constraint that is represented in B and is qualified with some sentence probability.

FROM UTILITY TO ACCEPTABILITY

One aim of this discussion is to lay the foundations for a normative theory of auctions and bidding that does not rely on knowledge of an agent's utility, and does not require an agent to make assumptions about her opponents' utilities or types, including whether they are aware of their utility. Such a theory must provide some mechanism that determines the *acceptability* of a deal—that is, the probability that the deal is acceptable to an agent. Agent Π is attempting to buy or bid for a second-hand motor vehicle with a specific period of warranty. Here, Π is bidding in a multi-issue auction for a vehicle, where the two issues are price and warranty period.

The proposition $(Accept(\delta) \mid I_t)$ means: "Π will be comfortable accepting the deal δ given that Π knows information I_t at time t." In an

auction for terms ω, Π's strategy **S** may bid one or more π for which $P(Accept((\pi,\omega))\,|\,I_t) \geq \alpha$ for some threshold constant α. This section describes how Π estimates $P(Accept(\delta)\,|\,I_t)$. The meaning of $Accept(\delta)$ is described below; it is intended to put Π in the position: "Looking back on it, I made the right decision at the time." This is a vague notion but makes good sense to the author.

With the motor vehicle application in mind, $P(Accept(\delta)\,|\,I_t)$ is derived from conditional probabilities attached to four other propositions: $Suited(\omega)$, $Good(\Omega)$, $Fair(\delta)$, and $Me(\delta)$, meaning respectively: "terms ω are perfectly suited to Π's needs," "Ω will be a good agent for Π to be doing business with," "δ is generally considered to be a fair deal at least," and "on strictly subjective grounds, the deal δ is acceptable to Π." These four probabilities are: $P(Suited(\omega)\,|\,I_t)$, $P(Good(\Omega)\,|\,I_t)$, $P(Fair(\delta)|\ I_t \cup \{Suited(\omega),\ Good(\Omega)\})$, and $P(Me(\delta)|\ I_t \cup \{Suited(\omega),\ Good(\Omega)\})$. The last two of these four probabilities factor out both the suitability of ω and the appropriateness of the opponent Ω. The third captures the concept of "a fair market deal" and the fourth a strictly subjective "what ω is worth to Π." The "*Me*(.)" proposition is closely related to the concept of a private valuation in game theory. This derivation of $P(Accept(\delta)\,|\,I_t)$ from these four probabilities may not be suitable for assessing other types of deals.

To determine $P(Suited(\omega)\,|\,I_t)$, if there are sufficiently strong preference relations to establish extrema for this distribution, then they may be assigned extreme values ≈ 0.0 or 1.0. Π is then repeatedly asked to provide probability estimates for the offer ω that yields the greatest reduction in entropy for the resulting distribution (McKay, 2003). This continues until Π considers the distribution to be "satisfactory." This is tedious, but the "preference acquisition bottleneck" appears to be an inherently costly business (Castro-Schez et al., 2004).

To determine $P(Good(\Omega)\,|\,I_t)$ involves an assessment of the reliability of the opponent Ω. For some retailers (sellers), information—of varying reliability—may be extracted from sites that rate them. For individuals, this may be done either through assessing their reputation established during prior trades (Ramchurn et al., 2003) or through the use of some intermediate escrow service that is rated for "reliability" instead.

$$P(Fair(\delta)\,|\,I_t \cup \{Suited(\omega), Good(\Omega)\})$$

is determined by reference to market data. Suppose that recently a similar vehicle with a three-year warranty sold for $6,500, and another less similar was sold for $5,500 with a one-year warranty. These are fed into I_t and are represented as two beliefs in B: β_3 : $Fair(2,[6.4,6.6))$; $B(\beta_3) = 0.9$, β_4 : $Fair(2,[5.4,5.6))$; $B(\beta_4) = 0.8$. In an open-cry auction, one source of market data is the bids made by other agents. The sentence probabilities that are attached to this data may be derived from knowing the identity and the reputation of the bidding agent. In this way the acceptability value is continually adjusted as information becomes available. In addition to β_3 and β_4, there are three chunks of knowledge in K. First, κ_2 : $Fair(4,4999)$ that determines a base value for which $P(Fair) = 1$, and two other chunks that represent Π's preferences concerning price and warranty:

$$\kappa_3 : \forall x,y,z((x > y) \rightarrow (Fair(z,x) \rightarrow Fair(z,y)))$$
$$\kappa_4 : \forall x,y,z((x > y) \rightarrow (Fair(y,z) \rightarrow Fair(x,z)))$$

The deal set is a 5 × 11 matrix with highest interval [7.0, ∞). The three statements in K mean that there are 56 possible worlds. The two beliefs are consistent with each other and with K. A complete matrix for $P(Fair(\delta)\,|\,I_t)$ is derived by solving two simultaneous equations

of degree two using equation 3. As new evidence becomes available, it is represented in B, and the inference process is re-activated. If new evidence renders B inconsistent, then this inconsistency will be detected by the failure of the process to yield values for the probabilities in [0,1]. If B becomes inconsistent, then the revision function R identifies and removes inconsistencies from B prior to re-calculating the probability distribution. The values were calculated using a program written in Java:

	$w=0$	$w=1$	$w=2$	$w=3$	$w=4$
$p=[7.0,\infty)$	0.0924	0.1849	0.2049	0.2250	0.2263
$p=[6.8,7.0)$	0.1849	0.3697	0.4099	0.4500	0.4526
$p=[6.6,6.8)$	0.2773	0.5546	0.6148	0.6750	0.6789
$p=[6.4,6.6)$	0.3697	0.7394	0.8197	0.9000	0.9053
$p=[6.2,6.4)$	0.3758	0.7516	0.8331	0.9147	0.9213
$p=[6.0,6.2)$	0.3818	0.7637	0.8466	0.9295	0.9374
$p=[5.8,6.0)$	0.3879	0.7758	0.8600	0.9442	0.9534
$p=[5.6,5.8)$	0.3939	0.7879	0.8734	0.9590	0.9695
$p=[5.4,5.6)$	0.4000	0.8000	0.8869	0.9737	0.9855
$p=[5.2,5.4)$	0.4013	0.8026	0.8908	0.9790	0.9921
$p=[5.0,5.2)$	0.4026	0.8053	0.8947	0.9842	0.9987

The two evidence values are shown above in bold face.

Determining $P(Me)(\delta) \mid I_t \cup \{Suited(\omega), Good(\Omega)\}$ is a subjective matter. It is specified using the same device as used for *Fair,* except that the data is fed in by hand "until the distribution appears satisfactory." To start this process, first identify those δ that "Π would be never accept"—they are given a probability of ≈ 0.0; and second those δ that "Π would be delighted to accept"—they are given a probability of ≈ 1.0. The *Me* proposition links the *ME* approach with "private valuations" in *GT*.

AUCTIONS

The *ME* analysis of auctions focuses on what agents actually do rather than their reasons for doing what they do. The four common auction mechanisms are considered for an auctioneer Υ, a single item and multi-issue bids each consisting of a set of deals. In the Dutch auction the auctioneer calls out successive sets of deals until one bidding agent shouts "mine." In the first- and second-price, sealed-bid mechanisms, bidding agents submit any number of multi-issue bids. The "Australian" mechanism is a variant of the common English mechanism in which agents alternately bid successive sets of deals until no further bids are received—as each set of deals is received, the auctioneer identifies the current winning bid. So, unlike in the multi-issue English mechanism, in the Australian mechanism the auctioneer is not required to publicize fully her winner determination criterion in advance, and the bidders are not required to submit successive bids that are increasing with respect to that criterion. In the two sealed-bid mechanisms and the Australian mechanism, the auctioneer determines the winner—and the runner up in the second-price mechanism—using a preference ordering on the set of all possible deals that may be made known to the bidding agents. The bids in these auctions may contain a large number of deals which is rather impractical.

Consider what happens from the auctioneer's point of view. Υ's expectation of what might happen will rely on both an understanding of the motivations and strategies of the agents taking part, and the rules of the auction mechanism. These two matters will effect Υ's choice of deal set, but otherwise the analysis is the same for the four common auction mechanisms. Suppose that there are S agents $\{\Omega_i\}_{i=1}^{S}$ bidding in the auction, and the value set $D = \{\delta_i\}_{i=1}^{D}$ contains D elements. Suppose that Υ has a total preference ordering $_{\Upsilon}$ on the deal set D, and that D is labeled such that if $i > j$ then $\delta_i {}_{\Upsilon}\delta_j$. Let the predicate $TopBid(\Omega,\delta)$ now mean "deal δ is the highest bid that Ω will make with respect to the order $_{\Upsilon}$." There are $S \times D$ ground literals in

terms of this predicate. This predicate will satisfy:

$$\kappa_7 : \forall ixy((TopBid(\Omega_i, x) \wedge TopBid(\Omega_i, y)) \rightarrow (x = y)).$$

Suppose that the deal set D has been chosen so that Υ expects each of the $(D + 1)^S$ possible worlds that are consistent with κ_7 to be equally probable for $TopBid(.)$ for each Ω_i for $i = 1,...,S$. This is the *symmetric* case when the expected performance of each of the S bidding agents is indistinguishable. The maximum entropy distribution is uniform and $\forall ijP(TopBid(\Omega_i, \delta_j)) = \frac{1}{D+1}$. Let the predicate $WinningBid(\delta)$ mean "deal δ is the highest bid that the $\{\Omega_i\}_{i=1}^S$ will make with respect to the order $\succ_\Upsilon$." Then:

$$\kappa_8 : \forall i(WinningBid(\delta_i) \leftrightarrow (\neg \exists jkTopBid(\Omega_j, \delta_k) \wedge (k > i)) \wedge (\exists nTopBid(\Omega_n, \delta_i)).$$

There are now $(S \times D) + D$ ground literals in terms of these two predicates, but still only $(D + 1)^S$ possible worlds. So:

$$P(WinningBid(\delta_i)) = (1 - \frac{D-i}{D+1})^S \times (1 - (\frac{i}{i+1})^S) \quad (5)$$

For example, if $S = 2$ and $D = 3$, then the probability of the highest of the three possible deals being bid by at least one of the two agents is $\frac{7}{16}$. If the total ordering $\succ_\Upsilon$ is established by a utility function, then this result enables the estimation of the expected utility.[8] The analysis completed so far may be applied to any sealed-bid auction or to any open-cry auction prior to any bids being placed. Once the bidding starts in an open-cry auction, information about what agents are or are not prepared to bid is available. This information may alter a bidding agent's assessment of the acceptability of a deal by feeding into the $Fair(.)$ predicate. It also alters the assessments of the probabilities of what the various opponents will bid, and of any deal being the winning bid. Bids made in an Australian auction provide lower limits, and bids not made in a Dutch auction provide upper limits to what the opponents will bid.[9] As these limits change, the assessment of these probabilities are revised using equation 3. A formula for $P(WinningBid(\delta_i))$ in terms of these limits is rather messy.[10] The value derived for $P(WinningBid(\delta_i))$ relies on κ_7 and κ_8 in K, together with expressions of the observed limits and the assumed expectation that each possible world is equally probable for $extitTopBid(.)$.

Now consider the four auctions from a bidding agent's point of view. Two strategies **S** for bidding agents are described for illustration only: first, a *keen agent* who prefers to trade on any acceptable deal to missing out—they are not primarily trying to optimize anything—although in the Australian auction they may choose to bid strategically and may attempt to reach the most acceptable deal possible; and second, a *discerning agent* who attempts to optimize expected acceptability and is prepared to miss out on a deal as a result.

First, consider keen agents. In a first-price, sealed-bid auction, these agents will bid the entire set $\{\delta \mid P(Accept(\delta) \mid I_t) \geq \alpha\}$. In an Australian, open-cry auction, these agents may attempt to submit bids that are just "superior" to the bids already submitted by other agents. The meaning of "superior" is determined by $\succ_\Upsilon$ and may be private information. If a bidding agent does not know $\succ_\Upsilon$, then it will have to guess and assume it. Suppose that Δ is the set of bids submitted so far by the opponents in an Australian auction. First define the set of bids that are just superior to:

$$\Delta : \Delta^+ = \{\delta \in D \mid \delta \notin \Delta, \exists \delta_1 \in \Delta, \delta \succ r\delta_1, \forall \delta_2((\delta \succ r\delta_2 \succ r\delta_1) \rightarrow ((\delta_2 = \delta) \vee (\delta_2 = \delta_1)))\}.$$

Now bid $\{\text{argmax}_\delta\{P(Accept(\delta) \mid I_t) \geq \alpha) \wedge (\delta \in \Delta^+)\}\}$. To avoid bidding against itself in a Vickrey auction, an agent will bid a set of deals that forms a *shell* Σ with respect to $_\Upsilon$ [i.e., $\forall \delta_i \delta_j \in \Sigma(\neg(\delta_i \Upsilon \delta_j))$]. An agent will only bid in a Vickrey auction if $_\Upsilon$ is known, because that ordering will determine the "highest" non-winning bid. This uncertainty makes the Vickrey auction less attractive to keen agents than the other three forms. If keen agents do *not* feed bidding information into their acceptability mechanism in the open-cry cases, then the expected revenue will be greatest in the first-price, sealed-bid, followed by the Dutch and then by the Australian—it is not clear how the Vickrey auction fares due to the uncertainty in it. Feeding bidding information into the acceptability mechanisms of keen agents may have an inflationary effect on expected revenue in an Australian auction, and bidding non-information may have a deflationary effect in the Dutch auction. The extent to which these effects may change the expected-revenue ordering will be strategy specific.

Second, consider discerning agents. A similar analysis to the result in equation 5 may be used by a discerning agent to optimize expected acceptability in the symmetric case. This analysis follows the general pattern of the standard *GT* analysis for utility optimizing agents (e.g., Wolfstetter, 1999); it is not developed here. For a discerning agent, the Vickrey mechanism has a dominant strategy to bid at, and the Australian mechanism right up to, the acceptability margin. For the Dutch and first-price mechanisms, the acceptability of the deals bid will be shaded-down from the margin. In both the Dutch and the Australian mechanisms, the margin of acceptability may move as bidding information becomes available.

TAKE IT OR LEAVE IT

The take-it-or-leave-it mechanism is a degenerate auction in that an agent makes a single bid that stands until it is withdrawn. An opponent agent may then choose to accept a standing bid. Further, some popular auctions, such as eBay, offer vendors the facility of a "Buy Now" option. To use this option the vendor must determine a take-it-or-leave-it price. The case of one buyer and one seller is considered here. Introduce the predicate $WillTrade(\Omega, \delta)$ meaning "that agent Ω will accept a proposed deal δ." Then an "acceptability optimizing" Π, with information I_t, will offer Ω the deal: $\text{argmax}_\delta (P(WillTrade(\Omega,\delta)) \times P(Accept(\delta) \mid I_t))$. For multi-issue δ, the distribution for $P(WillTrade(\Omega,\delta))$ is evaluated using equation 2 and the distribution for $P(Accept(\delta) \mid I_t)$.

The single-issue case is analyzed to illustrate the "*ME* method" and because it gives a different value to *GT*. Suppose that a seller has a good that she values at *r,* and she wishes to determine a take-it-or-leave-it price. First assume that the single buyer Ω will prefer to pay less than more:

$$\kappa_9 : \forall xy((\delta_x > \delta_y) \rightarrow (WillTrade(\Omega,\delta_x) \rightarrow WillTrade(\Omega,\delta_y))).$$

Second, choose the intervals $D = \{\delta_i\}_{i=1}^{D}$ such that $\mathbf{W}_{\{\{\kappa_9\},B\}}$ is uniform, where D is ordered naturally. Then $\mathbf{V}_K$ contains $D+1$ possible worlds for the predicate $WillTrade(\Omega, \delta)$ for $\delta \in D$, and $P(WillTrade(\Omega,\delta_i)) = \frac{i}{D+1}$. Suppose that the seller knows the additional information that Ω will pay δ_y and will not pay δ_n. Then K now contains two further sentences: $\kappa_{10} : \neg WillTrade(\Omega,\delta_n)$ and $\kappa_{11} : WillTrade(\Omega,\delta_y)$. There are now $n-y$ possible worlds, the maximum entropy distribution is uniform, and using equation 2:

$P(WillTrade(\Omega,\delta_i)) = \frac{n-i}{n-y}, \quad y \le i \le n$. In general the seller's expected surplus in offering the deal δ to agent Ω is: $P(WillTrade(\Omega,\delta)) \times (\mathbf{U}(\delta) - r)$, where $\mathbf{U}(\delta)$ is the utility. In the continuous case, the "expected utility-optimizing price" is $\frac{\mathbf{U}(\delta_n)+r}{2}$. This price is in terms of only the seller's valuation r and the knowledge $\neg WillTrade(\Omega,\delta_n)$; it is independent of the knowledge $WillTrade(\Omega,\delta_y)$. Both *ME* and *GT* assume κ_9. In the *GT* analysis (Wolfstetter, 1999), the expected utility optimizing price is $\frac{\bar{u}+r}{2}$, where $\bar{u}$ is the upper bound of an assumed uniform distribution for Ω's utility. The *GT* analysis relies on that assumption. The *ME* analysis relies on the observation $WillTrade(\Omega,\delta_y)$ and shows that the price is $\frac{\mathbf{U}(\delta_n)+r}{2}$. It is no surprise that these expressions have a similar structure. However they do not have the same value. Ω may be aware of her utility u_Ω for the good. The inherent inefficiency of bilateral bargaining (Myerson & Satterthwaire, 1983) shows for an economically rational Ω that u_Ω, and so consequently $\bar{u}$ may be greater than $\mathbf{U}(\delta_n)$. Further, δ_n may be a "high" offer and $\bar{u}$ may be less than $\mathbf{U}(\delta_n)$. It is unlikely that they will be equal.

CONCLUSION

The negotiating agent achieves its goal of reaching informed decisions while making no assumptions about the internals of its opponent.

As a bargaining agent, Π has five ways of leading a negotiation towards a positive outcome:

1. by making more attractive offers to Ω;
2. by reducing its threshold α;
3. by acquiring information to hopefully increase the acceptability of offers received;
4. by encouraging Ω to submit more attractive offers; and
5. by encouraging Ω to accept Π's offers.

The first two of these have been described. The third has been implemented but is not described here. The remaining two are the realm of argumentation-based negotiation which is the next step in this project. The integrated way in which Π manages both the negotiation and the information acquisition should provide a sound basis for an argumentation-based negotiator.

As an auction agent, Π bids because she feels comfortable as a result of knowledge acquisition, rather that being motivated by expected utility optimization. Information is derived generally from the World Wide Web, from market data, and from observing the behavior of other agents in the market. The agents described do not make assumptions about the internals of their opponents. In competitive negotiation, an agent's motivations should be kept secret from its opponents. So speculation about an opponent's motivations necessarily leads to an endless counter-speculation spiral of questionable value. These agents require a method of uncertain reasoning that can operate on the basis of a knowledge base that contains first-order statements qualified with sentence probabilities. Maximum entropy inference is eminently suited to this requirement, and has the additional bonus of operating with logical constants and variables that represent individual deals. So the deals may be multi-issue. Four simple multi-issue auction mechanisms have been analyzed for two classes of agent: keen agents that are primarily motivated to trade, and discerning agents that are primarily motivated by the optimization of their expected acceptability. The acceptability mechanism generalizes game-theoretic utility in that acceptability is expressed in terms of probabilities that are dynamically revised during a negotiation in response to both changes in the background information and the opponents' actions.

Much has not been described here including: the data and text mining software, the use of the Bayesian net to prompt a search for information that may lead to Π raising—or perhaps lowering—its acceptability threshold, and the way in which the incoming information is structured to enable its orderly acquisition (Debenham, 2004). The following issues are presently being investigated. The random world computations are performed each time the knowledge K or beliefs B alter—there is scope for using approximate updating techniques interspersed with the exact calculations. The offer-accepting machinery operates independently from the offer-making machinery—but not vice versa; this may mean that better deals could have been struck under some circumstances. Sierra and Debenham (2005) describe a foundation for trust based on information theory and maximum entropy.

REFERENCES

Bulow, J., & Klemperer, P. (1996). Auctions versus negotiations. *American Economic Review, 86*(1), 180-194.

Castro-Schez, J., Jennings, N., Luo, X., & Shadbolt, N. (2004). Acquiring domain knowledge for negotiating agents: A case study. *International Journal of Human-Computer Studies, 61*(1), 3-31.

Debenham, J. (2004). Bargaining with information. In N. Jennings, C. Sierra, L. Sonenberg, & M. Tambe (Eds.), *Proceedings of the 3rd International Conference on Autonomous Agents and Multi Agent Systems (AAMAS-2004)* (pp. 664-671).

Faratin, P., Sierra, C., & Jennings, N. (2003). Using similarity criteria to make issue tradeoffs in automated negotiation. *Journal of Artificial Intelligence, 142*(2), 205-237.

Halpern, J. (2003). *Reasoning about uncertainty.* Cambridge, MA: MIT Press.

Jaynes, E. (1957). Information theory and statistical mechanics: Part I. *Physical Review, 106,* 620-630.

Klemperer, P. (2000). *The economic theory of auctions: Volumes I and II.* Oxford, UK: Edward Elgar.

Kraus, S. (2001). *Strategic negotiation in multiagent environments.* Cambridge, MA: MIT Press.

MacKay, D. (2003). *Information theory, inference and learning algorithms.* Cambridge, UK: Cambridge University Press.

Muthoo, A. (1999). *Bargaining theory with applications.* Cambridge, UK: Cambridge University Press.

Myerson, R., & Satterthwaite, M. (1983). Efficient mechanisms for bilateral trading. *Journal of Economic Theory, 29,* 1-21.

Neeman, Z., & Vulkan, N. (2000). *Markets versus negotiations.* Technical Report, Center for Rationality and Interactive Decision Theory, Hebrew University, Israel.

Osborne, M. J., & Rubinstein, A. (1990). *Bargaining and markets.* San Diego: Academic Press.

Pietra, S.D., Pietra, V.D., & Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19*(2), 380-393.

Ramchurn, S., Jennings, N., Sierra, C., & Godo, L. (2003). A computational trust model for multi-agent interactions based on confidence and reputation. *Proceedings of the 5th International Workshop on Deception, Fraud and Trust in Agent Societies* (pp. 69-75).

Sandholm, T., & Vulkan, N. (1999). Bargaining with deadlines. *Proceedings of the National Conference on Artificial Intelligence (AAAI)* (pp. 44-51).

Sierra, C., & Debenham, J. (2005). An information-based model for trust. *Proceedings of the 4th International Conference on Autonomous Agents and Multi Agent Systems (AAMAS-2005)* (pp. 497-504).

Varian, H. (1995). Mechanism design for computerized agents. *Proceedings of the Usenix Workshop on Electronic Commerce* (pp. 13-21).

Wolfstetter, E. (1999). *Topics in microeconomics*. Cambridge, UK: Cambridge University Press.

ENDNOTES

1 The often-quoted oxymoron "I paid too much for it, but it's worth it" attributed to Samuel Goldwyn, movie producer, illustrates that intelligent agents may negotiate with uncertain utility.

2 Constants are 0-ary functions.

3 A sentence probability of 0.5 represents "maybe, maybe not."

4 In rather dire circumstances, King Richard III of England is reported to have initiated a negotiation with remarkably high stakes: "A horse! a horse! my kingdom for a horse!" [William Shakespeare]. Fortunately for Richard, a person named Catesby was nearby and advised Richard to retract this rash offer "Withdraw, my lord," and so Richard's intention to honor his commitments was not put to the test.

5 The introduction of $\overline{\omega}$ may be avoided by defining $P(Me(\tau,\omega))\ \frac{1}{1+\exp(-\beta\times(\mathbf{U}(\omega)-\mathbf{U}(\tau))}$ for $\mathbf{U}(\omega)\geq\mathbf{U}(\tau)$ and zero otherwise, where β is some constant. This is the sigmoid transfer function used in some neural networks. This function is near-linear for $\mathbf{U}(\omega)\approx\mathbf{U}(\tau)$, and is concave or "risk averse" outside that region. The transition between these two behaviors is determined by the choice of β.

6 This is reminiscent of Werner Heisenberg's indeterminacy relation, or *unbestimmtheitsrelationen*: "You can't measure one feature of an object without changing another"—with apologies.

7 After becoming CEO of Citicorp in 1967, Walter Wriston said: "Banking is not about money; it is about information." It is tempting to echo this as: "Negotiation is not about utility…."

8 In the continuous *GT* analysis, if X_i is a random variable representing the amount bid by Ω_i, and if the distributions for the X_i are uniform on [0.1], then the expected value of the winning bid is given by the expected value of the Sth order statistic $E(X_{(S)})=\frac{S}{S+1}$.

9 This is the *asymmetric* case.

10 In the continuous *GT* analysis, given a sample of S non-identical, independent random variables $\{X_i\}_{i=1}^{S}$ where X_i is uniform on $[C_i,1]$. For each sample, $p_i=P(X_i\geq X)=\frac{1-X}{1-C_i}$ if $C_i\leq X\leq 1$ and zero otherwise. So the probability that *none* of the X_i exceed $Y\geq\max\{C_i\}$ is $P(Y)=\prod_{j=1}^{S}(1-p_i)=\prod_{j=1}^{S}\frac{Y-C_j}{1-C_j}$, which is the probability distribution function for the largest Y. Then $E(Y)=\int_{Y=\max\{C_i\}}^{Y=1}Y\times f(Y)\times dY$ where $f(Y)=(\prod_{j=1}^{S}\frac{Y-C_j}{1-C_j})\times(\sum_{i=1}^{S}\frac{1}{Y-C_i})$. For example, for $S=2$, $C_1=c$, $C_2=d$, $0\leq c\leq d\leq 1$ then $E(Y)=\frac{4-(3\times d)-d^3+(3\times c\times(d^2-1))}{6\times(1-c)\times(1-d)}$, and if $c=d=0$ then $E(Y)=\frac{2}{3}$ as we expect.

Chapter L
An Introduction of Evolutionary Computation in Auctions

Asunción Mochón
Universidad Nacional de Educación a Distancia, Spain

Yago Sáez
University Carlos III of Madrid, Spain

David Quintana
University Carlos III of Madrid, Spain

Pedro Isasi
University Carlos III of Madrid, Spain

ABSTRACT

The increasing use of auctions as a selling mechanism has led to a growing interest in the subject. Thus both auction theory and experimental examinations of these theories are being developed. A recent method used for carrying out research on auctions has been the design of computational simulations. The aim of this chapter is to give a background about auction theory and to present how evolutionary computation techniques can be applied to auctions. Besides, a complete review to the related literature is also made. Finally, an explained example shows how a genetic algorithm can help automatically find bidders' optimal strategies for a specific dynamic multi-unit auction—the Ausubel auction—with private values, drop-out information, and with several rationing rules implemented. The method provides the bidding strategy (defined as the action to be taken under different auction conditions) that maximizes the bidder's payoff. The algorithm is tested under several experimental environments that differ in the elasticity of their demand curves, number of bidders, and quantity of lots auctioned. The results suggest that the approach leads to strategies that outperform sincere bidding when rationing is needed.

INTRODUCTION

According to the historical archives, the first time auctions were implemented was 500 B.C. Herodotus reports that in Babylon, men had to buy their future wives by bidding in auctions. Nowadays auctions are widely used as selling mechanisms in different markets around the world. The items auctioned range from the wide variety being offered on Internet marketplaces like eBay, the auctions of art, antiques, financial assets, agricultural goods, fish, and so forth. Perhaps at the present time the most important agents using auctions are governments. The authorities usually carry out auctions both for selling and buying. Governments sell by means of auction commodities like emission permits, electromagnetic bands for communication, or the rights for use of natural resources or public companies. On the other hand, when governments are the buyer that is in procurement auctions, bidders compete for the right to sell their products or services and the lowest bid wins the contract. As a result of these activities, auction theory and experimental examinations of these theories are of growing interest.

Experiments done on auctions suggest that frequently bidders make systematic bidding errors, so the final outcome does not fit the theoretical results (Kagel, 1995). In order to understand these divergences, a recent method used for analyzing strategies on auctions is the use of systems of artificial adaptive agents (AAAs). Moreover, these learning models with adaptive agents can be helpful not only where theory and experimental results disagree, but for those situations where the environment is so complicated that a theoretical result has not been modeled yet. In this context, genetic algorithms (GAs) are a good learning method for optimization in complex problem domains. In this way, the use of machine learning systems can help find equilibrium strategies or in the evaluation, from different points of view, of an auction itself with respect to other possible auctions. The analysis of these systems gives us new approaches to understanding the economic and social behavior of auctions.

The remainder of the chapter is organized as follows. The next section gives the reader a minimum background about auction theory. The third section presents how evolutionary computation techniques can be applied to auctions and gives a review of related literature. Then, in the section Analysis of the Ausubel Auctions by Means of Evolutionary Computation, a detailed example of one research project that uses genetic algorithms to develop bidder strategies in Ausubel auctions is given; at this point the experimental framework is shown, along with a description of the bidding strategy encoding with the results obtained. Finally, conclusions are drawn.

THE STANDARD AUCTION TYPES

Although auctions have been used from time immemorial, they did not enter into the economic literature until relatively recently. Most likely the starting point of the auction theory is the work done by Vickrey (1961) on the equivalence in the expected revenue of different auction forms, where he described, for the first time, the sealed-bid second-price auction. Since this research a lot of work has been done on this topic, especially at the end of the 1970s. It is very difficult to list the numerous papers on auctions; however, some references are: Klemperer (2000a, 2000b), who collects together in two volumes most of the critical papers in the economic literature of auctions up to the year 2000; Krishna (2002), who gives an account of developments in the field in the 40

years since Vickrey's pioneering paper; Klemperer (2003), who surveys the basic theory of how auctions work and describes the world-record-setting 3G mobile-phone license auctions; and Milgrom (2004), who gives an up-to-date treatment of both traditional theories of optimal auctions and newer theories of multi-unit auctions and package auctions.

There are four basic types of auctions when a unique item is to be bought or sold: the open ascending price or English auction, the open descending price or Dutch auction, the sealed-bid first-price auction, and the sealed-bid second-price or Vickrey auction. In the English auction the price is successively raised until one bidder remains and that bidder wins the item at the final price. In the Dutch auction the auctioneer starts at a very high price and then lowers it continuously. The first bidder who calls out that he accepts the current price wins the object. In the first and second price auction, each bidder independently submits a single bid, and the item is sold to the bidder with the highest bid. In the first price, the winning bidder pays his bid, but in the second price, the winning bidder pays the second highest bidder's bid.

Optimal Auctions and Efficiency Auctions

The performance of auctions can be evaluated on two different grounds depending on the context. From the seller's point of view, he or she will be interested in revenues, the expected selling price. In this sense, an auction is defined as optimal when it reports the highest expected revenue to the seller among all the different kinds of auctions. On the other hand, if we consider society as a whole, efficiency may be more important. An auction is said to be efficient when it allocates the object to the bidder who values it the most *ex post.*

As we have mentioned, much of the auction literature, theory and experimental, analyzes different auction formats in order to compare the revenues for both buyers and sellers or the final allocation of the items. As we will see in the section, Evolutionary Computation Techniques and Auctions, most of the research done with computational experiments focuses on the bidders' strategy or the market design in order to evaluate these components.

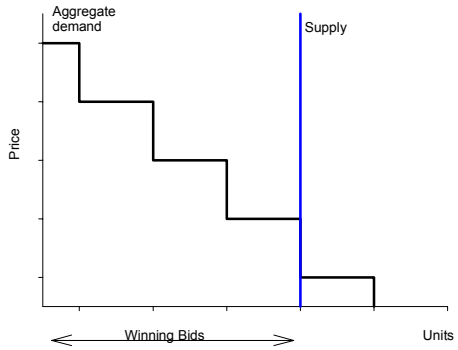
Multi-Unit Auctions

When we have multiple objects to be sold, the seller has many options to consider. First, he must decide whether to sell the objects separately in multiple auctions (sequential or simultaneous auctions) or jointly in a single auction. Moreover, he must determine whether the objects to be sold are identical or heterogeneous. When the items are not homogeneous, the seller must face the question about how to package the goods offered for sale. An alternative is to design an auction that let the bidders choose the packages for themselves (package auctions or combinatorial auctions).

In multiple-object environments where individual bidders may demand more than one homogeneous item in a single auction, the seller must choose among a wide variety of auction formats.

The most common multi-unit auctions with sealed-bid formats are the discriminatory auction, the uniform-price auction, and the Vickrey auction. In these auctions, bidders are asked to submit a bid vector indicating how much they are willing to pay for each additional unit. Then an aggregate demand function is obtained by horizontally adding the individual demand functions. As the supply function is just a vertical line, the winning bids would be those on the left side of the intersection of both functions. The allocation rule establishes that the awarded units are equal to the units of winning bids submitted (see Figure 1). The differentiating features in these auctions are the pricing rules.

Figure 1. Allocation rule for sealed bid multi-unit auction

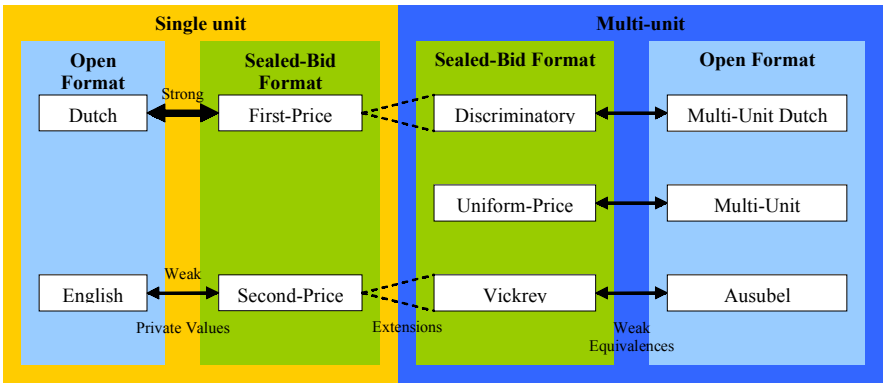


In the discriminatory auction each bidder pays an amount equal to the sum of his winning bids. In a uniform price auction, all the items are sold at the market-clearing price so that demand equals supply. Finally, in the Vickrey auction, a bidder who wins a unit pays for the highest losing bids of the other bidders, not including his own.

Likewise, in the open format we can emphasize the Dutch (or open descending price) auction, the English (or open ascending-price) auction, and the Ausubel auction. In the multi-unit Dutch auctions, the auctioneer calls out a price that is gradually lowered until a bidder indicates that he is willing to buy an item at the standing price. This bidder is awarded an item and the auction proceeds until all the units are sold. In a multi-unit English auction, the auctioneer starts calling out a low price and bidders ask for the units they are willing to pay. As the price goes up, bidders reduce their demands until the demand equals the supply at a price where all the units are sold and the auction is over. The Ausubel auction is an alternative ascending-price auction, and items are allocated according to the following procedure. As the price goes up, the demanded quantities go down, until the number of units demanded by the other bidders is less than the supply. At that price, a bidder "clinched" or earned an item. The bidders will have to pay for the awarded units at the standing prices when they clinched the items. A deeper explanation of this format is offered further on.

Just as in the single-unit environment, there is some equivalence between these auction formats. Under private value models, the following weak equivalences hold: the discriminatory auction and the multi-unit Dutch auction, the uniform-price auction and the multi-unit English auction, and finally, the Vickrey auction and the Ausubel auction. Figure 2 includes the equivalences of the basic auction formats for both single units and multi-unit.

Figure 2. Equivalence of auction formats



Double Auction

All the auctions that we have commented on so far have a common pattern: they are one-sided auctions. This means that there is one single seller and many buyers, or vice versa one buyer and many sellers (procurement auctions). Nevertheless, it is also possible to have many buyers and many sellers in what is referred to as a two-sided auction.

A double auction is a two-sided auction where buyers and sellers are treated symmetrically with buyers submitting bids and sellers submitting offers. In the simple double auction model with a single good to trade in the market, the buyer and seller each submit a sealed bid: the buyer submits the price he is willing to pay (p^b) and the seller submits the minimum payment for which he will deliver the good (p^s). During each period every seller is randomly matched with buyers, and if $p^b \geq p^s$, one unit of the good is traded.

A particular specification of this model is the continuous double auction. In this auction sellers announce decreasing offer prices, while simultaneously and asynchronously the buyers present increasing bid prices, with the sellers being free to accept any buyer's bid at any item and the buyers being free to accept any seller's offer at any time, in the absence of an auctioneer.

EVOLUTIONARY COMPUTATION TECHNIQUES AND AUCTIONS

Research in negotiation models has been pursued in different fields: game theory, social sciences, and artificial intelligence. In recent years there has been an increasing interest in the theory of auctions and in experimental works in order to prove these theories. Both experimenters and theorists have to deal with different results and conclusions, and it is generally perceived that the "misbehavior" of subjects in their bidding strategies is the main reason for those differences encountered (Cox, Roberson, & Smith, 1982; Cox, Smith, & Walker, 1983, 1985; Kagel & Levin, 1985, 1986; Kagel, Harstad, & Levin, 1987; Kagel & Dyer, 1988; Andreoni & Miller, 1995). Thus, to have a better understanding of the different behaviors of bidders and auction outcomes, it is necessary to add computational experimental frameworks that complement auction theory. The experimental frameworks are developed to simulate auction environments with real bidder behaviors. Such computational experiments allow us to study the complex, dynamic, and stochastic processes inherent in auction learning models (Andreoni & Miller, 1995). This hard task usually requires the application of different types of artificial intelligence techniques. The most popular approaches study the interaction of AAAs, developing their behavior with adaptive learning algorithms like GAs, genetic programming (GP), evolutionary strategies (ESs), or evolutionary programming (EP).

The main problem in designing the experimental frameworks is how to encode the possible behaviors and bidders' strategies. This problem is solved in different manners according to the aim of the research. There is a great deal of work done using evolutionary computation (EC) techniques to simulate the learning behavior of rational agents. Some authors use agents that are already implemented, like agents with Zero Intelligent Plus (ZIP) learning heuristics (Cliff, 1998; Priest & Tol, 1998; Priest, 1999). Alternatively, other authors prefer to develop agents with their own particular protocol for negotiation decisions, one-to-one negotiations (Faratin, Sierra, & Jennings, 1998), bilateral negotiations (Matos, Sierra, & Jennings, 1998), or multiple auctions (Anthony & Jennings, 2003).

Evolving Bidding Strategies

The most popular approaches to bidding optimization are conducted with GAs. These algorithms are usually applied to search for the optimal strategy driving the behavior of the AAA. Figure 3 illustrates this procedure. In this way, a population of agents or bidders interacts with an auction simulator implemented with an EC algorithm like a GA. The GA makes bidders' strategy evolve, searching for the best bid in the virtual environment.

In this field the agents can be implemented in different ways. Usually each agent is represented by one individual of the GA population. Elsewhere, the agents are autonomous software that interacts and have their own behavior implemented.

Encoding the Strategies

The first problem found when applying GAs to evolve bidders' behavior is the fact that the bidding strategies must be defined and encoded. This difficult task can be achieved in different ways, but the most widespread approach is to represent the agents' behavior with a binary string. The length of the string can be used to determine the number of possible strategies for one agent. Moreover, there are other features that heavily affect the behavior of the bidders such as: auction format, affiliation, risk-aversion, budget constraints, asymmetries, remaining time of the auction, bargain tactic, and so on. As the complexity and number of strategies grow, the binary encoding is more difficult to apply, the real or integer encoding being the most recommendable. For example, a possible encoding to represent strategies described above is using one population of individuals where each individual is represented with a vector of floating point values. Each individual represents one bidding agent, and its genes are the parameters of the different tactics. The genes' value is the relative weight for each tactic, as Anthony and Jennings (2003) had. Another simple and efficient encoding is using strings to represent parameters which define a determined bidding function (Andreoni & Miller, 1995).

An important constraint of auction theory is that, generally, it assumes a benchmark model where the hypothesis of the revenue equivalence principle holds: private values, risk-neutral, budget constraints, and symmetric valuation, or at least most of them. Nevertheless, the EC techniques are powerful tools to simulate auction environments where these assumptions can be relaxed.

Related Work

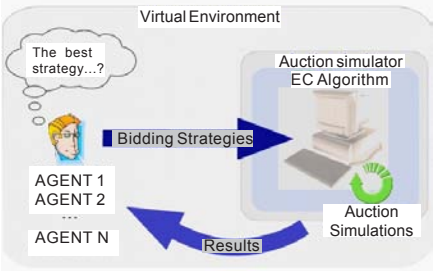
There is work done in this field with techniques of EC as mentioned before. A summary of some of them and their results are presented below:

- Cliff (2003) makes a proposal for automated online auctions. The author suggests that so far, the most popular auction market mechanisms were designed by humans for humans. Nevertheless, it is possible that in the future, trader agents might replace human traders in online marketplaces (Das, Hanson, Kephart, & Tesauro, 2001). However, the number of possible auction mechanisms is so large that the search for the most appropriate one is not an easy task. The approach suggested by Cliff starts with a representation that allows a wide range of possible market mechanisms, including the continuous double auction market (CDA), the one-sided auctions (similar to the English) and the Dutch auctions, as well as an infinite range of hybrids for private value

auctions. Cliff models the market by means of a set of Zero-Intelligence-Plus traders. These agents (whose basic behavior was described in Cliff, 2002) generate quotes according to a set of parameters using the Widrow-Hoff rule. These markets were allowed to evolve using a GA, in order to minimize the deviation of transaction prices from the theoretical equilibrium prices. The results suggest that CDA mechanism, the most common in financial markets, might not be the best one in every situation. Evolved hybrid market mechanisms could be more efficient when shocks in supply and demand occur, while CDA can be more appropriate when the nature of supply-and-demand curves cannot be predicted.

- Andreoni and Miller (1995) created a GA model to capture the bidding patterns evident among human subjects in experimental auctions. These authors compared the different results for first- and second-price auction formats with affiliated private values, independent private values, and common value auctions. Experiments were done for first- and second-price auctions with four and eight bidder groups. For each auction they ran 20 trials until 1,000 generations. The algorithm searches over 2-parameter linear functions in each auction. In the results it is remarkable how the subjects in second-price auctions do better than those in first-price auctions, although bidders tend to over-bid in all auctions. Moreover, buyers' profits are lower in first-price independent value auctions than in second-price auctions. This result contradicts the revenue equivalence theorem and shows how bidders make systematic bidding errors. With these findings they concluded that adaptive learning can provide useful insights into a bidder's behavior, and can reconcile existing theoretical and experimental results.
- Dawid (1999) modeled a two-population GA to study the learning behavior of buyers and sellers in a sealed-bid double auction with private information. To apply the genetic operators, he encodes each individual of the population with a string, and a fitness value is calculated measuring the success of the individual by the average payoff earned in the last period (a group of "*m*" iterations). The algorithm is launched during 2,500 iterations, and the fitness is updated every 20 generations (one period). After doing a theoretical study describing the behavior of GAs in economic systems, they confirmed their findings with simulations, and illustrated how agents can coordinate on efficient bidding strategies in a sealed-bid double auction market without previous knowledge of the structure of the model.
- Anthony and Jennings (2002, 2003) suggested an agent-based approach to bidding strategy evolution for specific environments using GAs. Under their framework, there are different auctions—English, Dutch, and Vickrey—and the agent should show a behavior that ensures that the desired item is delivered according to the user's preferences. The agents are expected to obtain one instance of the desired item by a deadline paying a price lower or equal to the consumer's private valuation. The behavior of the agents is subject to a set of polynomial functions (Anthony, Hall, & Dang, 2001) that represent bidding constraints. These include the remaining time left, the remaining auctions left, the user's desire for a bargain, and the user's level of desperateness. The authors denote these functions as tactics. The combination of these tactics is re-

Figure 3. General scheme for EC and auctions



ferred to as a strategy. The strategy requires the allocation by the agent of the appropriate weight to each tactic. These weights plus the set of parameters required to define the above-mentioned constraint functions conform the array of floating point values that define the individuals in the population to be evolved. The fitness of the individuals is assessed according to three criteria. The first one is to be used if the delivery of the item is extremely important. The other two are appropriate when the agent is looking for a bargain and when the previous objectives are equally important. The authors compare the evolved strategy to a basic strategy that recommends bidding in the auction that has the closest end time where the current bid is lower than its private valuation. The main result is that is it feasible for evolving agents to perform successfully under both specific and diverse auction frameworks.

- Byde (2003) also explores the possibility of using GAs in the auction mechanism design field. As the revenue equivalence theorem depends on a number of assumptions that may not hold, the author suggests a robust method that might be less dependent on the simplifications required by mathematical analyses. The work examines a space of sealed-bid auction mechanisms where the payments are determined by linear combinations of first- and second-price auctions. In this context, the behavior of bidders depends on a number of parameters such as risk attitudes or correlation between preferences. The strategy followed by an agent is defined by a piecewise linear function from input signal to bid. All the agents follow optimal strategies that are determined by a GA. The author finds types of environment in which both pure first- and second-price auctions are inferior to hybrid approaches.

ANALYSIS OF THE AUSUBEL AUCTIONS BY MEANS OF EVOLUTIONARY COMPUTATION

For this chapter an example of recent research that uses EC techniques to find bidders' optimal strategies for the Ausubel dynamic multi-unit auction is presented. In this research a GA provides the bidding strategy (defined as the action to be taken under different auction conditions) that maximizes the bidders' payoff. The bidders' payoff is defined as the difference of the values of the units clinched by the bidders minus the standing price at which they clinched them. The algorithm is tested under several experimental environments, number of bidders, and quantity of lots auctioned.

Ausubel (2004) demonstrated that in this auction format with pure private values, sincere bidding (SB) by every bidder is an *ex post* perfect equilibrium yielding to an efficient outcome. Notwithstanding, SB is not the unique equilibrium. In the experiment carried out by Kagel, Kinross, and Levin (2004) with independent private values, behavior in the Ausubel

auction with dropout information comes significantly closer to SB. As the price goes up and demanded quantities go down, it is possible that, for a certain increase of price, the supply is not covered at the final price. In these circumstances the following rationing rule is introduced. We turn to the previous price and allocate all the demanded units to the bidders. Then the excess demand is calculated, and units are removed from the bidders with the highest demanded quantity not clinched until the total demand equals supply. It is not possible to remove units from bidders that were already clinched in previous rounds.

For a better understanding of the auction format, Appendix A includes an example of the Ausubel auction format with the SB strategy and the GA strategy.

Designing the GA for the Bidding Strategy

The aim of the GA is to find a set of bidding rules that maximize the payoff of a bidder. To this end there are several experimental environments defined for which each bidder has a different set of values and a specific bidding strategy.

Defining the Experimental Environments

The experiment employs an independent-private-value framework in which bidders having weakly diminishing marginal values, are risk neutral, and have no budget constraints. All bidders except one have a fixed strategy, which is SB. Bidders will reduce their demanded quantities at the exact moment when the actual price is equal to their real values. On the other hand, the bids of the bidder who follows the GA strategy will depend on several actions that we will define as deviations from the SB strategy. This behavior is explored for several environments that differ in the following variables: the number of lots auctioned $m = 10, 15, 20, 25$; the number of bidders $n = 4, 6, 8$; and the elasticity of the bidders' demand curves. The elasticity of bidders' demand curve depends on their valuations of the items which are generated as we describe below.

Every bidder has a set of values organized from higher to lower that specify the marginal value from the consumption of each additional unit. In our model we force each bidder to define his values for at least as many items as the total supply. Bidders' values are drawn independently and identically distributed from a uniform distribution with support [0,200], with new random draws for each additional unit. These values are generated with two different algorithms, in the same way as Mochón, Quintana, Sáez, and Isasi (2005).

Bidding Strategy

In each auction all bidders are programmed to bid sincerely. There is only one bidder that has to beat his computer rivals by following the GA bidding strategy. The definition of the GA strategy that we suggest requires the identification of actions linked to specific auction conditions. Each action is defined in terms of deviations (over and underbidding) from the SB strategy. The demanded quantity according to the SB strategy of bidder i in the round l is represented by q^{l}_{SBi}. To this end we consider the following four possible actions to be taken: bid half of the SB quantity, bid the SB quantity, bid 50% more of the SB quantity, or bid twice as much as the SB quantity (see Table 1). The bidder that evolves with the GA strategy will bid one of these four actions according to the auction condition of each round. All these strategies have an upper bound that is the lowest of either the number of units being auctioned or, alterna-

Table 1. Each bidder considers four possible actions for each auction condition.

Bidding Strategy	Quantity Demanded	Binary Code
bid half of the SB	$q^{l}_{SBi}/2$	00
bid the SB	q^{l}_{SBi}	01
bid 50% more of the SB	q^{l}_{SBi} * 1.5	10
bid twice as much as the	q^{l}_{SBi} *2	11

tively, the units demanded in the previous round (as demand is required to be non-increasing). The lower bound is the number of units that the participant has already clinched.

Bidders face 61 possible auction conditions. The first one represents the initial market where no relevant information is available. To define the other 60 possible auction conditions, we make up a combination of the potential values of three different indicators. In each auction the value of these indicators changes from one round to another. As our auction format has drop-out information, the bidder that follows the GA strategy constantly calculates these indicators and selects the appropriate action or bid.

The first indicator is the trend of price-demand elasticity from the competitors ($e_{p,d}$), classified into five categories: ≥ 1.5; <1.5; =1; <1; and <0.5. The second one is the percentage of active bidders with respect to the number of bidders at the beginning of the auction (AB%),

Table 3. Payoff average difference for SB and GA bidding strategies

	m=10	*m=15*	*m=20*	*m=25*
Elastic				
n=4	0.16	1.40	18.98	7.36
n=6	1.60	0.32	0.08	0.00
n=8	0.24	1.32	0.68	0.64
Inelastic				
n=4	0.60	0.84	0.64	1.08
n=6	0.40	0.40	0.44	0.80
n=8	1.36	0.12	0.08	1.60

Table 2. Bidders face 61 possible auction conditions made up from the combination of three indicators plus the initial condition.

$\varepsilon_{p,d}$		AB(%)		OM	
≥ 1.5	+	< 100%	+	≥ 2	+Initial= 61
< 1.5		< 50%		< 2	
= 1		< 25%		< 1.25	
< 1				< 1	
>0.5					

classified into three categories: <100%; < 50%; and < 25%. And the third indicator is the operating margin of the strategic bidder (OM), defined as the number of objects that have not been clinched with respect to the number of objects demanded by the bidder. It has been classified it into four categories: ≥ 2; <2; <1.25; and <1. For a better understanding of these indicators, see Mochón et al. (2005). These indicators are represented in Table 2.

Genetic Algorithm

The encoding of the bidding strategies to the individuals is a direct process. As we mentioned before, all strategies consist of a set of 61 integers valued from 0 to 3, depending on the action to be taken which depends on the auction conditions. Since the actions can be encoded in 2 bits, each strategy could be represented as an

Table 4. Effect of the GA strategy respect to the SB one for all participants with the SRR

	Payoff GA-Payoff SB bidder 1	Payoff GA-Payoff SB $i \neq 1$	Seller`s revenue GA-Seller`s revenue SB
Elastic demand curves			
Example 1	73	-27	-59
Example 2	20	12	-40
Example 3	6	-10	-1
Inelastic demand curves			
Example 4	8	-6	-15
Example 5	11	14	-32
Example 6	9	14	-25

array of 122 bits. This includes the SB strategy. Since we define the action of following the SB strategy as 1 (01), the strategy of a SB bidder would be represented as a string of 122 bits (where the value of every odd bit is 0 and 1 otherwise).

The assessment of a bidding strategy is made by running an auction twice. In the first run, all bidders fulfill SB. In the second one, we let one bidder (bidder 1) evolve according to the GA and the others follow SB. Once we have the allocations and payments for each bidder, the payoff for the strategic bidder is calculated for both runs. The GA strategies will be successful as higher payoff for bidder 1 are in the second run rather than in the first one, no matter what the others bidders' payoff or the sellers' revenue.

The search for the best strategy for each environment was performed using populations of 30 individuals that evolved for 1,000 generations. The GA used elitism, a roulette wheel selection mechanism, single-point crossover, and a gene mutation operator with likelihood 1%.

Evaluating the Bidding Strategy

For each environment explained before, we ran the algorithm 25 times, which means 30 individuals that evolve for 1,000 generations. To evaluate the GA strategy, a comparison of the GA bidder's payoff with what he would have obtained with the SB strategy is made. Table 3 includes the payoff average difference for bidder 1 with the GA strategy and the SB.

The results show that the GA outperforms the SB strategy in all the experiments except one, as the bidders' payoff is always higher with the GA strategy than with the SB one. This reveals that, at least in some cases, bidders' payoff can be improved by using our GA strategy rather than the SB one.

The key point is to find out what is the bidding strategy of the GA that outperforms the SB. To this end we have analyzed in detail the process of several auctions (with elastic and inelastic demand curves) for n=4 and m=15 in which the GA beats the SB strategy. An example is included in Appendix A.

When we compare the GA strategy of bidder 1 for the experiments analyzed, it can be observed that the participant is overbidding at the beginning of the auction. He continues overbidding until he determines the best moment to underbid and push the auction to rationing. When the demand of the other competitors plus what he has already clinched is equal to M, the excess demand of the auction is equal to bidder i's bid minus what he has already clinched. Therefore, by underbidding in the next round, he has a high probability of pushing the auction to a rationing situation.

The strategic bidder will also calculate the number of items that he will get if the rationing rule is put into effect in that round. When what he has clinched plus what he will probably get with the rationing rule is connected with his true valuation, then it would be the moment to underbid. At that point the best strategy for bidder 1, according to the GA, is to make his possible minimum bid in the next round.

With the specific rationing rule assumed, we have analyzed the effect of the GA strategy over all the participants involved (see Table 4). In the six examples that concern us, bidder 1 is always better off by following the GA bidding strategy rather than the SB. He always wins at least the same number of objects and at a lower average price. On the other hand, the seller is always worse off as he sells all the items, but at a lower price. Hence his revenues are lower. The final outcome for the other participants depends on each specific auction. Sometimes they also have higher payoff as the average selling price of the items is lower. Neverthe-

less, their payoffs can also be reduced from the fact that sometimes bidder 1 takes advantage of the rationing rule.

CONCLUSION

In this chapter a review of the auction literature has been made. The starting point is the most representative work in the field. Moreover, an introduction of how the EC and the implementation of AAAs can be a helpful tool to study auctions has been made. Using learning mechanisms based on Artificial Intelligence, such as the GA, can be a key point for improving the understanding of bidders' strategies and the final outcome of an auction, both in allocations and payments. Although there is work done up to now, there are many possibilities for implementing EC techniques with different auction formats in order to test the results with the theory models.

As an example of the application of EC to auctions, a dynamic ascending multi-unit auction with implementation of Vickrey pricing has been chosen, which is referred to as the Ausubel auction. The authors have developed a GA that can be successfully employed to evolve bidding strategies for this auction format. By relating the GA strategy to a specific bidder, the authors compare the payoff to what would have been obtained with SB with two different demand curves: elastic and inelastic. The experiments were conducted separately for different numbers of bidders (n=4, 6, 8) and objects auctioned (m=10, 15, 20, 25).

The evaluation of the GA strategy revealed that the algorithm outperforms the SB strategy, as the bidder payoff is always equal to or higher than with the SB strategy. The optimal bidding strategy that the GA proposes for bidder 1 is to overbid just until the GA finds the optimal round to push the auction to rationing (according to the probability to force rationing and its final allocation of items with respect to its real valuations). At that point the bidder underbids by making its lowest possible bid. With this performance the bidder maximizes his payoff. In all the auctions analyzed, bidder 1 is always better off, the seller is always worse off, and the outcome of the other participants will depend on each specific auction.

These results reveal that the implementation of GAs and the selection of a rationing rule can be a key point in the final outcome of an Ausubel auction, both in allocations and payments. Some authors have already studied the importance of the rationing rule in establishing the existence of equilibrium in many games, including auctions (see Jackson, Simon, Swinkels, & Zame, 2002).

After reviewing the literature that combines EC and auctions, and the example explained, it is possible to conclude that the EC techniques are a powerful tool in simulating auction environments without restrictions, which means a significant constraint to theoretical models. With this advantage it can be possible to understand the divergences found in the experimental and theoretical analysis of auctions.

REFERENCES

Andreoni, J., & Miller, J. H. (1995). Auctions with artificial adaptive agents. *Games and Economic Behavior, 10*(1) 1039-1051.

Anthony, P., Hall, W., Dang, V. D., & Jennings, N. R. (2001). Autonomous agents for participating in multiple online auctions, in: Proceedings of the International Joint Conference on Artificial Intelligence *Proceedings of the Workshop on E-Business and Intelligent Web* (pp. 54-61).

Anthony, P., & Jennings, N. R. (2002). Evolving bidding strategies for multiple auctions. *Proceedings of the 15th European Conference on AI* (ECAI-2002) (pp. 178-182), Lyon, France.

Anthony, P., & Jennings, N. R. (2003). Developing a bidding agent for multiple heterogeneous auctions. *ACM Transactions on Inter. Technology, 3*(3) 185-217.

Ausubel, L. M. (2004). An efficient ascending-bid auction for multiple objects. *American Economic Review, 94,* 1452-1475.

Byde, A. (2003). Applying evolutionary game theory to auction mechanism design. *Proceedings of the ACM Conference on Electronic Commerce 2003* (pp. 192-193).

Cliff, D. (1998). Evolving parameter sets for adaptive trading agents in continuous double-auction markets. *Proceedings of the Agents'98 Workshop on Artificial Societies and Computational Markets,* Minneapolis, MN (pp. 38-47).

Cliff, D. (2003). Explorations in evolutionary design of online auction market mechanisms. *Electronic Commerce Research and Applications, 2,* 162-175.

Cliff, D. (2002). *Minimal-intelligence agents for bargaining behaviors in market environments.* Hewlett-Packard Laboratories Technical Report HPL-2002-321.

Cox, J., Roberson, B., & Smith, V. (1982). Theory and behavior of single object auctions. In V. L. Smith (Ed.), *Research in experimental economics* (Vol. 2). Greenwich, CT: JAI Press.

Cox, J., Smith, V., & Walker, J. (1983). Tests of a heterogeneous bidder's theory of first price auctions. *Economics Letters, 12*(3-4), 207-212.

Cox, J., Smith, V., & Walker, J. (1985). Experimental development of sealed-bid auction theory; calibrating controls for risk aversion. *American Economic Review, 75*(May), 160-165.

Das, R., Hanson, J., Kephart, J., & Tesauro, G. (2001). Agent-human interactions in the continuous double auction. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-01),* (pp. 1169-1187).

Dawid, H. (1999). On the convergence of genetic learning in a double auction market. *Journal of Economic Dynamics and Control, 23*(9-10), 1545-1567.

Faratin, P., Sierra, C., & Jennings, N. R. (1998). Negotiation decision functions for autonomous agents. *International Journal of Robotics and Autonomous Systems, 24*(3-4) 159-182.

Jackson, M. O., Simon, L. K., Swinkels, J. M., & Zame, W. R. (2002). Communication and equilibrium in discontinuous games of incomplete information. *Econometrica, 70,* 1711-1740.

Kagel, J. H. (1995). Auctions: A survey of experimental research. In J. H. Kagel & A. E. Roth (Eds.), *The handbook of experimental economics* (pp. 501-585). Princeton, NJ: Princeton University Press.

Kagel, J. H., & Dyer, D. (1988). Learning in common value auctions. In R. Tietz, W. Albers, & R. Selten (Eds.), *Experimental games and markets* (pp. 184-197). Berlin: Springer-Verlag.

Kagel, J. H., Harstad, R. M., & Levin, D. (1987). Information impact and allocation rules in auctions with affiliated private values: A laboratory study. *Econometrica, 55*(6), 1275-1304.

Kagel, J. H., Kinross, S., & Levin, D. (2004). *Implementing efficient multi-object auctions*

institutions: An experimental study of the performance of boundedly rational agents. OSU Working Paper 11/04 (pp. 1-24).

Kagel, J. H., & Levin, D. (1985). Individual bidder behavior in first-price private value auctions. *Economics Letters, 19,* 125-128.

Kagel, J. H., & Levin, D. (1986). The winner's curse and public information in common value auctions. *American Economic Review, 76*(5), 894-920.

Klemperer, P. (2000a). *The economic theory of auctions* (Vol. 1). Cheltenham, UK: Edward Elgar.

Klemperer, P. (2000b). *The economic theory of auctions* (Vol. 2). Cheltenham, UK: Edward Elgar.

Klemperer, P. (2003). *Auctions: Theory and practice.* Princeton, NJ: Princeton University Press.

Krishna, V. (2002). *Auction theory.* San Diego: Academic Press.

Matos, N., Sierra, C., & Jennings, N. (1998). Determining successful negotiation strategies: An evolutionary approach. *Proceedings of the 3rd International IEEE Conference on Multi Agent Systems (ICMAS)* (p. 182), Washington, DC.

Mitchell, M. (1996). *An introduction to genetic algorithms.* Cambridge, MA: MIT Press.

Mochón, A., Quintana, D., Sáez, Y., & Isasi, P. (2005). Analysis of Ausubel auctions by means of evolutionary computation. *Proceedings of the IEEE Congress on Evolutionary Computation 2005* (Vol. 3, pp. 2645-2652).

Preist, C. (1999). Commodity trading using an agent-based iterated double auction. *Proceedings of the 3rd Annual Conference on Autonomous Agents* (pp. 131-138).

Preist, C., & van Tol, M. (1998). Adaptive agents in a persistent shout double auction. *Proceedings of the 1st International Conference on the Internet, Computing and Economics* (pp. 11-18). Charleston, SC: ACM Press.

Vickrey, W. (1961). Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance, XVI,* 8-37.

APPENDIX A: AN EXAMPLE OF AN AUCTION PROCESS WHERE THE GA BEATS THE SB STRATEGY

Bidders' values for m=[1..15]
Bidder 1. $v_{1,k}$: 182 91 81 41 35 20 10 5 3 2 2 2 1 1 1;
Bidder 2. $v_{2,k}$: 100 50 25 22 15 12 6 3 2 1 1 1 1 1 1 ;
Bidder 3. $v_{3,k}$: 100 50 33 17 16 8 4 4 2 1 1 1 1 1 1;
Bidder 4. $v_{4,k}$: 106 76 44 27 14 7 4 2 1 1 1 1 1 1 1;

Table A.1. Auction process and outcome with SB for all bidders

Table A.1a. Auction process with SB for all bidders

p^l	Q^l	q^l_1	C^l_1	q^l_2	C^l_2	q^l_3	C^l_3	q^l_4	C^l_4
10	22	6	0	6	0	5	0	5	0
12	21	6	0	5	0	5	0	5	0
14	20	6	1	5	0	5	0	4	0
15	19	6	2	4	0	5	1	4	0
16	18	6	3	4	1	4	1	4	1
17	17	6	4	4	2	3	1	4	2
20	16	5	4	4	3	3	2	4	3
22	15	5	5	3	3	3	3	4	4
Seller's revenue= 269									

	Price paid for the units clinched					Final allocation	Final payment	Payoff
	1st unit	2nd unit	3rd unit	4th unit	5th unit			
Bidder 1	14	15	16	17	*22*	*5*	*84*	346
Bidder 2	16	17	20			*3*	*53*	122
Bidder 3	15	20	22			*3*	*57*	126
Bidder 4	16	17	20	22		*4*	*75*	178

Table A.2. Auction process and outcome with SB for i≠ 1 and bidder 1 GA

Table A.2.a. Auction process with SB for i≠ 1 and bidder 1 GA

p^l	Q^l	q^l_1	C^l_1	q^l_2	C^l_2	q^l_3	C^l_3	q^l_4	C^l_4
10	28	12	0	6	0	5	0	5	0
12	27	12	0	5	0	5	0	5	0
14	26	12	1	5	0	5	0	4	0
15	25	12	2	4	0	5	0	4	0
16	24	12	3	4	0	4	0	4	0
17	14	3	6	4	3	3	3	4	3
Seller's revenue= 249									

	Price paid for the units clinched						Final allocation	Final payment	Payoff
	1st unit	2nd unit	3rd unit	4th unit	5th unit	6th unit			
Bidder 1	14	15	16	17	17	17	*6*	*96*	354
Bidder 2	17	17	17				*3*	*51*	124
Bidder 3	17	17	17				*3*	*51*	132
Bidder 4	17	17	17				*3*	*51*	175

p^l: price at round l.
Q^l: total demand at round l.
q^l_i: bidder i demand at round l.
C^l_i: items clinched by bidder i at round l.

Chapter LI
Virtual Organization Support through Electronic Institutions and Normative Multi-Agent Systems

Henrique Lopes Cardoso
University of Porto, Portugal

Ana Paula Rocha
University of Porto, Portugal

Eugénio Oliveira
University of Porto, Portugal

ABSTRACT

The multi-agent system (MAS) paradigm has become a prominent approach in distributed artificial intelligence. Many real-world applications of MAS require ensuring cooperative outcomes in scenarios populated with self-interested agents. Following this concern, a strong research emphasis has been given recently to normative MAS. A major application area of MAS technology is e-business automation, including the establishment and operation of business relationships and the formation of virtual organizations (VOs). One of the key factors influencing the adoption of agent-based approaches in real-world business scenarios is trust. The concept of an electronic institution (EI) has been proposed as a means to provide a regulated and trustable environment, by enforcing norms of behavior and by providing specific services for smooth inter-operability. This chapter exposes our work towards the development of an agent-based EI providing a virtual normative environment that assists and regulates the creation and operation of VOs through contract-related services. It includes a presentation of the EI framework, knowledge representation structures for norms in contracts, and a description of two main institutional services, namely negotiation mediation and contract monitoring.

INTRODUCTION

The multi-agent system (MAS) paradigm has become a prominent approach in distributed artificial intelligence. These systems start with the individual—the agent—and evolve to populated environments where the most important feature is the interaction between agents.

The concept of intelligent agents is today dominant in artificial intelligence (Russell & Norvig, 2003). Agents are described as entities having a set of intrinsic capabilities, such as *autonomy* (control of own decision making), *reactivity* (response to changes in the environment), and *pro-activeness* (goal-directed behavior) (Wooldridge & Jennings, 1995). Moreover, the so-called strong notion of agency considers agents as entities having mental states, including *beliefs, desires,* and *intentions* (the BDI architecture; Rao & Georgeff, 1995). Another important capability—*social ability*—emphasizes the fact that agents do not act in an isolated environment, and will inevitably have to interact with other agents.

Thus, while agent theory has its inspirations in psychology and cognitive science, MAS research is influenced by organizational and social sciences, distributed computing, and economics. Furthermore, many MAS researchers look for inspiration in nature, both from collective cooperative behavior in groups of animals (e.g., ant colonies) and from our social interactions as (not necessarily cooperative) human beings.

Typical MAS applications include inherently distributed or complex domains. While some problems require system architectures including cooperative agents developed so as to accomplish an overall goal, in other cases agents may represent independent self-interested entities, with no presupposed cooperation besides mere interaction efforts. The former types of problems may be addressed through a centralized design, producing a top-down specification of a MAS environment with an overall purpose. The latter types are usually conceived as open environments, where heterogeneous agents arising from different sources interact either cooperatively or competitively. In this setting, agents may form organizations that dynamically emerge in a bottom-up fashion from the individuals, which together agree, usually through a negotiation process, to cooperatively perform some task not doable individually.

As one might expect, although decentralized and dynamic systems are much more appealing, they must be handled with hybrid approaches, since a minimum set of requirements is necessary to allow for heterogeneous and independently developed agents to successfully interact. One way of achieving such a common milieu is by defining communication standards, such as those proposed by FIPA (2002). However, an important issue arises when attempting to apply agents in real-world settings: how to ensure cooperative outcomes in scenarios populated with self-interested agents. A possible answer to this problem is to regulate the environment, enforcing appropriate types of agent behavior. This should provide a level of trust necessary for the development of real-world applications of open MAS.

Following these concerns, a strong research emphasis has been given recently to normative MAS (Boella, van der Torre, & Verhagen, 2005). A normative system is a set of interacting agents whose behavior can usefully be regarded as governed by norms (Jones & Sergot, 1993). Agents are subject to these norms, which influence their decision making. Therefore, besides their goals, agents must take into account the norms that apply to them. However, considering autonomy as a central property of agents, norms are often used as a means to regulate the environment by providing incentives for cooperative behavior through norma-

tive constraints, while allowing agents to choose whether to obey or to violate them (Castelfranchi, Dignum, Jonker, & Treur, 2000).

A major application area of MAS technology is e-business automation, comprising not only the use of information gathering and filtering agents, but also the establishment and operation of business relationships. The business-to-business case is particularly important, and includes not only sell and purchase operations, but also the formation of enterprise consortiums, which have been addressed through the notion of virtual enterprises or virtual organizations. This application domain clearly matches the aforementioned MAS research concepts (Fischer, Muller, Heimig, & Scheer, 1996): agents can represent the individual interests of each business entity, and negotiate in order to establish cooperative business agreements. Agent organizations can thus emerge establishing a set of commitments among them.

One of the key factors towards the adoption of agent-based approaches in real-world business scenarios is trust. When attempting to automate the creation and operation of business relationships, the behavior of agents must be made predictable, by creating a regulated environment that enforces agents' commitments. The concept of an *electronic institution* (EI) (Dignum & Dignum, 2001; Lopes Cardoso, Malucelli, Rocha, & Oliveira, 2005) has been proposed as a means to provide such a regulated and trustable environment, by enforcing norms and providing specific services.

In this chapter we expose our work towards the development of an agent-based EI providing a virtual normative environment that assists and regulates the creation and operation of virtual organizations (VOs). These are composed of agents representing different real-world entities, which adhere to a set of norms regulating the established cooperation commitments. We will start by introducing the framework of our EI and by exposing its normative environment. We will then address the issue of organizational emergence among agents within the institutional environment, and the formal means to explicitly state agents' commitments. A presentation of the main institutional services will follow—namely negotiation mediation and contract monitoring, focusing on the enforcement of norms. The chapter will conclude discussing the real-world application of the described approach and its relation to relevant contract law concepts.

ELECTRONIC INSTITUTION FRAMEWORK

The application of open MAS to real-world problems raises trust concerns. It is essential to provide guarantees that (software) agents will comply with coordination efforts. Moreover, when considering scenarios where agents agree on cooperative joint activities, it is necessary to provide mechanisms that force them to comply. The intrinsic nature of agents within MAS research does not provide cooperative assumptions, as agents are defined as autonomous and self-interested entities. This case is exacerbated when dealing with heterogeneous, independently developed, and privately owned agents.

The concept of social structure is a central issue in the social sciences: it is assumed to exist in order to impose a sense of order and predictability within a group of individuals. Social structure typically involves a framework of norms attached to roles played by members of a society. A role represents the way someone is expected to behave in particular situations, by having an associated set of normative expectations.

From a social perspective, different types of norms can be identified, with different abstrac-

tion levels. On one hand we have values, conventions, and abstract norms that are implicitly adhered to and may not have an explicitly defined social response in case of deviation. On the other side of the spectrum, we may have formal (legal) norms and prescriptions that include explicitly defined punishments to deal with rule violations.

Norms can also play an important role in open artificial agent systems, where they improve coordination and cooperation (Conte, Falcone, & Sartor, 1999) and allow for the development of trust and reputation mechanisms. As in real-world societies, norms provide us a way to achieve social order (Castelfranchi, 2000) by controlling the environment and making it more stable and predictable.

Having norms is not sufficient by itself, since agents will not voluntarily submit themselves to associated penalties in case of deviation. Therefore, appropriate mechanisms are needed to enforce norm compliance. This is where an electronic institution comes about: it provides the necessary level of trust by ensuring an enforceable normative environment. Other approaches towards the concept of EI (Esteva, 2003) have considered restrictive settings in which agents are not allowed to violate norms. In our perspective, to enforce norms is not the same as preventing their violation. This approach allows us to maintain the autonomous nature of agents, while influencing their decision making by ensuring that certain consequences will hold in case of non-compliance.

Institutional Reality

In order to provide a trustable social (agent) environment, the EI must have means to register what is going on. In general, two main types of events compose this "institutional reality" (Lopes Cardoso, & Oliveira, 2005, inspired by Searle, 1995), which is based on the creation of "institutional facts."

Most events will depend on agents' activity. In the case of software agents, we must take into account their observable actions: illocutions (messages exchanged between agents which must be monitored by appropriate institutional services). On the other hand, there are events that do not depend on the performance of any action. The most obvious example is the passage of time, which is important to verify the fulfillment or violation of norms (more on this later on). Therefore, institutional facts may come about from these two sources. Taking into account the moment when an institutional fact comes about, we represent it using the following structure:

ifact(InstitutionalFact, Timestamp)

When applying the EI framework to real-world scenarios like e-contracting, it is essential that external events (those that take place in the real world) are reflected inside the EI's virtual environment. This issue is addressed with institutional roles, as explained below.

Norm Specification

Formal models of norms typically rely on deontic logic (von Wright, 1951), a branch of modal logic. Also known as the logic of normative concepts, it embraces the notions of obligation, permission, and prohibition. Although traditionally used to analyze normative reasoning in law, applications of deontic logic in computer science exist (Wieringa & Meyer, 1993), not limited to the domain of legal knowledge representation and legal expert systems; other applications include, for example, authorization mechanisms and electronic contracting.

Extensions to the original work on deontic logic have been made so as to allow its practical

use. These include approaches to handling norm violations and associated sanctions, which may consist of contrary-to-duty obligations (Jones & Carmo, 2001): obligations that come into force in sub-ideal situations (i.e., when other obligations are violated). Other important variations consider the use of conditional and temporal aspects: obligations are often made conditional on the occurrence of another event and have an associated deadline (Dignum, Meyer, Dignum, & Weigand, 2002; Dignum, Broersen, Dignum, & Meyer, 2005).

Taking a simplified approach, we can consider norms as describing the expected behavior of agents when certain circumstances arise. In order to achieve a computational representation of norms, we can thus distinguish two main parts—the situation in which the norm applies, and its prescription:

Situation $\rightarrow$ *Prescription*

The *Situation* describes when the norm is in place. The *Prescription* specifies what should be accomplished in order for the norm to be fulfilled. With this representation, norms lend themselves to a rule-based implementation. Rule-based systems (Giarratano & Riley, 1998) allow us to encode simple reasoning procedures through rules. Each rule has two main components: the antecedent part (where the rule's conditions are specified, also known as the *left-hand side* of the rule) and the consequent part (where the rule's conclusions are stated, also known as the *right-hand side* of the rule). The conditional element of norms is satisfied according to institutional facts. Moreover, the prescriptive part indicates what else should be accomplished.

The most obvious way of prescribing certain behavior through a norm is by indicating obligations. Thus, an agent may be obliged to bring about a certain state of affairs. For practical purposes, obligations typically include a deadline by which they are to be satisfied:

obligation(Bearer, InstitutionalFact, Deadline).

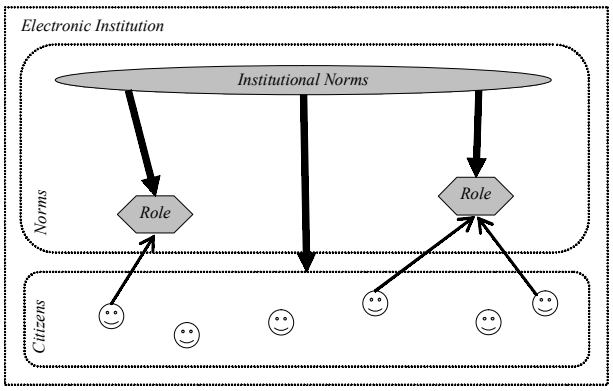
The *InstitutionalFact* that the *Bearer* must bring about by a certain *Deadline* will be part of institutional reality when the obligation is fulfilled. Notice that, instead of dictating the exact action an agent must perform, we prescribe the institutional fact that he must bring about. This enables an agent to delegate or outsource tasks conducting to the accomplishment of such state of affairs, while still being responsible before the institution for the (un)fulfillment of the obligation.

Other approaches towards norm implementation (e.g., Vázquez-Salceda, Aldewereld, & Dignum, 2004; García-Camino & Rodríguez-Aguilar, 2005) have also considered conditional permissions and prohibitions, subject to temporal references for both their activation and ceasing. With such an approach, it is straightforward to represent permissions and prohibitions with the same structure as the obligation above. Yet, we may adopt the deontic logic formalizations of permission and prohibition with respect to that of obligation: an agent is permitted do to something if it is not obliged to not do it; an agent is forbidden to do something if it is obliged not to do it.

Norms and Roles

The prescription of institutional norms comprises the top-down specification of the MAS environment (see Figure 1). Agents become subject to a set of norms when they assume a social dimension to which those norms refer. Within the EI framework, norms may apply to all citizens, to citizens enacting specific roles, or to specific groups of agents. The next section

Figure 1. Institutional norms as a top-down specification of the MAS environment



addresses the case where a group of agents may adopt a set of norms to regulate their joint activity.

Institutional norms represent the commitments of agents towards the EI. By assuming roles, agents become committed to their associated norms. A model of normative specifications based on roles can be found in Pacheco and Carmo (2003). The idea of ascribing norms to roles allows us to specify and anticipate the expected performance of the parts of a complex system (e.g., an agent-based organization) before specific agents start taking part in the system. When agents take on roles, they inherit their associated norms.

Within an EI, we can define a set of general roles (e.g., supplier or customer) that agents may adhere to, and prescribe the expected behavior of agents enacting those roles. When an agent enters the institutional environment (by registering in the EI) and announces itself as performing some institutionally defined role(s), it implicitly becomes committed to the institutional norms attached to the role(s) it plays.

Another important part that roles play in our EI setting concerns the creation of institutional reality. As mentioned before, agents' actions concern the exchange of illocutions. Two issues are worth mentioning here. First, since agents are autonomous: they are free to utter any illocutions they like. Second, part of the institutional reality concerns what happens in the "real world." This poses a problem of action certification. If, for instance, an institutional fact reports a payment of a certain amount to have taken place, we cannot rely on the bearer of such a payment stating that it is fulfilled. Instead, we would trust an independent financial third party, providing a certified institutional service.

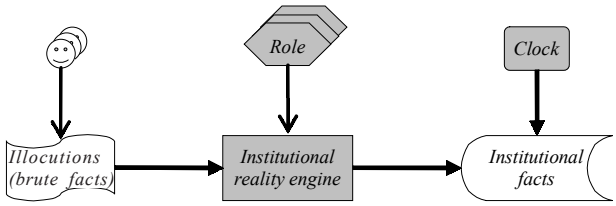
Therefore, we must define a set of institutional roles with which we assign powers concerning the creation of institutional reality. Following Searle (1995), inside the EI we distinguish between brute facts, which concern agents' illocutions, and institutional facts (see Figure 2). Rules implementing authoritative relations between roles and institutional reality are defined, making a connection between what is said and what is taken for granted. According to Searle (1995), these are "constitutive rules." An agent performing a given institutional role is said to be empowered to achieve the effects expressed in its role-related constitutive rules.

Until now, we have described an institutional environment within which agents interact by taking roles. We have also presented a means to specify norms that regulate agent behavior and how institutional reality comes about by connecting agents' illocutions to the roles they play. In the next section we describe how to specify, with contracts, commitments resulting from agents' interactions.

VIRTUAL ORGANIZATION EMERGENCE

Forming temporary coalitions of agents' efforts is an important strategy for live entities to deal with hard, complex tasks. The creation of vir-

Figure 2. Creation of institutional reality



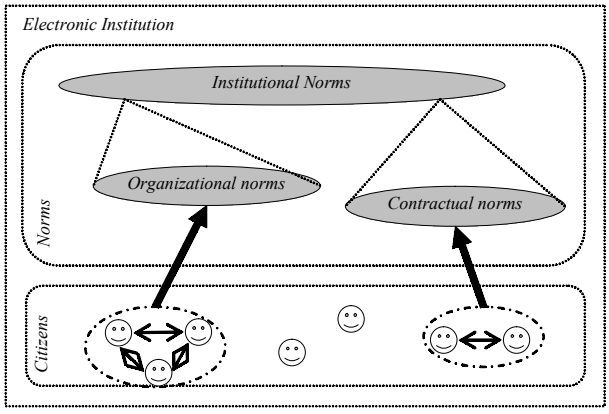
tual organizations is, likewise, a major trend in cooperative business, where different companies try to align their businesses to perform integrated and more complex activities. B2B players are becoming more focused on their core businesses, and rely on outsourcing and dynamic consortiums. The autonomous and self-interested nature of agents fits this scenario. Agents may represent different business units or enterprises, which come together to address new market opportunities by combining skills, resources, risks, and finances no single partner can alone fulfill (Dignum & Dignum, 2002).

Agents may thus form organizations that dynamically emerge from the individuals, in a bottom-up fashion, establishing a set of commitments among partners. Agents agree to align their activities in a cooperative setting, which is regulated by specific norms, usually agreed upon as a result of a negotiation process. Differently from institutional norms, these are voluntarily written and adhered to by agents, instead of being pre-established in the institutional environment. Moreover, these norms apply only to the subgroup of agents participating in the agreement. Agents commit to cooperative agreements because it is in their interest to do so. Figure 3 illustrates bottom-up norm emergence.

Any cooperation activity requires trust between the involved partners. In open MAS, agents' performance records may not be assessable. If such information is available, reputation ranks can be used by agents when choosing appropriate partners. Nevertheless, a VO may potentially include agents that have never worked together in the past. In this case, trust must be based on third parties, which verify whether agents comply with their promises. This makes it necessary to formulate agents' commitments explicitly through e-contracts, which are then monitored and enforced by the EI.

In the following specifications we use the Prolog (Clocksin & Mellish, 1981) notation conventions for variables and relations, although our rule-based approach is founded on general first-order logic (and thus does not assume the Prolog clause syntax).

Figure 3. Bottom-up norm emergence from the individual agents



Contract Specification

In Dignum et al. (2002), a logical formalism for describing interaction in an agent society, including social norms and contracts, is presented. Focusing on contract representation, it gives emphasis to conditional obligations with deadlines. Other approaches to contract representation through norms include Sallé (2002), considering normative statements that can comprise obligations, permissions, and prohibitions. Sanctions are seen as obligations or prohibitions activated by the violation of another obligation. Also, Kollingbaum and Norman (2002) propose the inclusion of obligations, permissions, and sanctions in a contract specification language. Each of these constructs has a similar structure, including activation and expiration conditions (which together define a "window of opportunity" within which an agent must or can act).

The use of sanctions as contrary-to-duty obligations is what makes normative multiagent systems effective. Sanctions and the desire to avoid them are the motive for agents to comply with norms. Agents reason about the effects of their actions both in terms of positive and negative consequences: they consider both the norm and the sanction to decide whether to fulfill the norm or not (Sallé, 2002).

An approach to contract representation based on event calculus can be found in Knottenbelt and Clark (2005). This representation incorporates how a contract is to be fulfilled (that is, which events initiate or terminate obligations), making it a heavier structure.

Following the use of norms as explained before, we concentrate on specifying contracts using norms that prescribe obligations, where these concern the achievement of institutional facts by certain deadlines. However, since these norms are not institutional, but relative to a certain context—that of the agreement established between agents—we extend the norm representation to include its scope:

[Context] Situation → Prescription

This allows us to have, within the same normative environment, norms applicable to different contexts.

We also introduce a distinction between one-shot contracts and longer-termed cooperation agreements. The aim here is to embrace not only contracts formalizing sell and purchase operations, but also more complex settings, such as a VO that involves specific interactions during a certain timeframe (Lopes Cardoso & Oliveira, 2004). The VO is regulated by a set of norms that regulate its continuous nature. Inside this ongoing relationship, contracts implementing the agreed-upon cooperation can be established, comprising a set of instantiated norms.

Simple Purchase Contracts

General contracts for purchase operations between a customer and a vendor can be specified by an institutional fact stating that such a contract (identified by *IdPC*) is in place:

ifact(purchase_contract(IdPC, Customer, Vendor, Item, Quantity, Price), PCTime)

Additionally, a set of behavior norms representing contract clauses prescribe the expected behavior of contracting partners. The following templates represent this information:

[purchase_contract:IdPC] obligation(Vendor, delivery(Vendor, Item, Quantity, Customer), PCTime+10)
[purchase_contract:IdPC] fulfilled(Vendor, delivery(Vendor, Item, Quantity, Customer), TD)

→ obligation(Customer, payment(Customer, Price, Vendor), TD+30)

According to the clauses above, a vendor's obligation exists to deliver the contract's item to the customer within 10 time units; when this is done, the customer is obliged to pay to the vendor the agreed amount. Within a norm, literals either in the situation or in the prescription part are dependent on the context of the norm. This means that prescribed obligations and their fulfillment occur within a certain context. The same applies to institutional facts, which are often related to a given context (as for instance a payment in the context of a certain purchase).

The contract is typically complemented with sanctioning norms applicable to sub-ideal (contrary-to-duty) situations. These are activated when other norms are violated. Their structure looks like the following:

[Context] violated(Bearer, IFact, Deadline) → obligation(Bearer, NewIFact, NewDeadline)

where *NewIFact* consists of a new state of affairs supposedly harder than the previously prescribed and violated one. Exemplifying, two additional clauses could be added to the contract above:

*[purchase_contract:IdPC] violated(Vendor, delivery(Item, Quantity, Customer), Deadline) → obligation(Vendor, delivery(Item, Quantity, Customer), Deadline+5) ∧ obligation(Vendor, payment(10%*Price, Customer), Deadline+5)*
*[purchase_contract:IdPC] violated(Customer, payment(Price, Vendor), Deadline) → obligation(Customer, payment(Price*110%, Vendor), Deadline+15)*

The first of these clauses indicates that if the vendor violates its obligation to deliver the item, it becomes obliged to pay the customer 10% of the price, while keeping the original obligation with a new deadline. The second clause states that, should the customer fail to pay the price to the vendor, it becomes obliged to pay a 10% increased price by a new deadline.

Eventually, problems may arise when an agent chooses to violate a sanction. A discussion of possible approaches to such situations is offered later.

Virtual Organization Cooperation Agreements

The creation of contracts regulating VO agreements is becoming common in the B2B world, where parties create consortiums that exist during a period of time. A distinguishing feature between these and the former purchase contracts is that they do not terminate after a predetermined normative path (that is, a sequence of fixed obligations). Instead, VO cooperation agreements have an ongoing nature: they may include repetitive yet unscheduled interactions.

Let us illustrate the normative specification of a VO cooperation agreement, considering a case where the intended cooperation consists of the exchange of resources between partners. The agreement will aggregate the organization's constitutional information, including the cooperation effort parties commit to and their general business process flow. Institutional facts register the standing cooperation agreement, which resulted from a successful negotiation process. We consider the following templates:

[] ifact(cooperation_agreement(IdCA, Participants, Resources), CATime)

[cooperation_agreement:IdCA]
coop_effort(Participant, Resource, MinQuantity, MaxQuantity, Frequency, UnitPrice)
[cooperation_agreement:IdCA]
business_process(From, Resource, To)

Besides the initial fact stating who the participants are and what resources are considered within this cooperation agreement, cooperation efforts indicate quantity ranges for the supply of resources within a given frequency, together with agreed prices. Business process entries indicate the resources that are supposed to flow between participants. Their effective transfer, however, is dependent on appropriate requests, as we shall see. A minimum of one *coop_effort* fact per participant and resource should be present. Likewise, there should be at least one *business_process* fact for each *coop_effort,* and it must conform to the latter. Both these kinds of facts refer to a certain cooperation agreement (*IdCA*) and are thus contextualized by it.

Although the commitments of each party are implicitly defined by such institutional facts, the possibility for their verification requires that they are made explicit. The following norm addresses this need:

[cooperation_agreement:IdCA]
ifact(request(Requester, Resource, Quantity, Answerer), TR) $\wedge$
business_process(Answerer, Resource, Requester) $\wedge$ *coop_effort(Answerer, Resource, MinQt, MaxQt, Freq, _)* $\wedge$
calculate_performed_effort(Answerer, Resource, Freq, TR, PE) $\wedge$
PE+Quantity<=MaxQt
$\rightarrow$ *obligation(Answerer, acknowledge(Answerer, Resource, Quantity, Requester), TR+10)*

This norm states that if a predicted request (considering the stated business process and cooperation effort) is made in the context of a cooperation agreement, then the envisaged agent is obliged to accept it. An institutional procedure (*calculate_performed_effort*) is invoked for calculating the effort already performed by the agent within the timeframe indicated in the cooperation effort frequency, taking into account the request time. If the agent does not exceed its promised efforts, the obligation comes into effect.

Then, we state that an operational contract (containing operations that implement the cooperation agreement) comes into existence when an agent fulfils its obligation to accept a request:

[cooperation_agreement:IdCA]
fulfilled(Answerer, acknowledge(Answerer, Resource, Quantity, Requester), TA)
$\rightarrow$ *ifact(operational_contract(IdOC, Requester, Answerer, Resource, Quantity), TA)*

Finally, we define how operational contracts are to be handled within a cooperation agreement. For instance, if delivery and payment are to take place:

[cooperation_agreement:IdCA, operational_contract:IdOC]
obligation(Answerer, delivery(_,Resource, Quantity, Requester), OCTime+10)
[cooperation_agreement:IdCA, operational_contract:IdOC]
fulfilled(Answerer, delivery(_, Resource, Quantity, Requester), TD) $\wedge$
coop_effort(Answerer, Resource, _, _, _, UnitPrice)
$\rightarrow$ *obligation(Requester, payment(_, UnitPrice*Quantity, Answerer), TD+30)*

in which *IdOC* remains unbound, as these norms apply to all operational contracts which will be created in the future within agreement *IdCA*.

Facilitating Contract Formation

The creation of contracts may become cumbersome if we assume that they must predict all possibilities concerning their performance. Moreover, when considering the possibility to automatically negotiate, using software agents, a contractual relationship, it is necessary to provide a structure on which negotiation can be based. This is the role of a contract template: it contains domain-independent interaction schemata and variable elements (such as price, quantity, deadlines, and so on) to be filled in with domain-specific data resulting from a negotiation (Kollingbaum & Norman, 2002).

In general, three possibilities exist concerning the creation of a contract (Field & Hoffner, 2005):

- using monolithic contract templates, which are rigid and require no assembling from smaller granularity clauses, only filling in the relevant information from a business agreement;
- building compositional contracts, starting from contract clause templates (the building blocks) that have to be assembled, resulting in a complete contract conforming to certain composition rules; and
- taking a hybrid approach, combining monolithic templates with a number of contract clause templates that can be assembled in different ways.

In any case, complete contracts that spell out in complete detail the exact duties of each party under every possible situation are imaginary; in practice, no real contract ever achieves this level of completeness. As such, incomplete contracts must rely on a legislative system that resolves any issues not explicitly addressed by the parties. In fact, an important concept in contract law theory is the use of "default rules" (Craswell, 2000): rules that define the parties' obligations in the absence of any explicit agreement to the contrary. These exist with the intent of facilitating the formation of contracts, allowing them to be underspecified by defining default clauses or default values.

It is therefore imaginable that institutional norms could embrace default rules. These are particularly useful for handling abnormal contract performance (that is, dealing with violations through default sanctions), since these situations may be (at least in principle) not likely to occur in practice.

INSTITUTIONAL SERVICES

According to our EI rationale, the main goals of an EI include (Lopes Cardoso et al., 2005):

- to support agent interaction as a coordination framework, making the establishment of business agreements more efficient; and
- to provide a level of trust by offering an enforceable normative environment.

The way to achieve these goals is by providing a set of institutional services, covering the phases of the lifecycle of contractual relationships: information discovery, contract negotiation, and execution. These phases have also been identified as *pre-contractual, contractual,* and *post-contractual,* respectively. This nomenclature emphasizes the core of any business relationship: the contract. The EI's normative environment is devoted to monitoring and enforcing contracts that are expressed through sets of norms regulating a cooperative business activity. We consider not only simple contractual relationships between two parties, but also the formation of VOs composed of multiple partners.

Most currently available support to e-business is devoted to the first phase: we can identify typical e-market functions such as yellow-page support, customer aggregation mechanisms, and recommender systems. We concentrate on the subsequent two stages, where we think of services assisting the negotiation of contracts (negotiation mediation, ontology-mapping, contract-building tools through templates) and their execution (contract monitoring and enforcement).

In this section we will focus on negotiation mediation, and on contract monitoring and enforcement services. These are aligned as shown in Figure 4 (adapted from Lopes Cardoso et al., 2005).

Negotiation Mediation

Different negotiation protocols may be devised that enable the establishment of B2B contracts. Most work in this area has focused on bilateral interactions, developing on multi-attribute utility theory for handling multiple issue negotiations. Extensive work has also been done with auctions, addressing in most cases single-issue interactions (based on the price of goods). There is a comparatively smaller amount of research concerning the formation of a VO, where the outcome of a negotiation is the establishment of an agreement between a set of partners. Here we present our approach to a negotiation process devoted to the formation of a VO (Rocha, Lopes Cardoso, & Oliveira, 2005).

Our proposed negotiation protocol has in mind to make electronic commerce activity closer to what happens in traditional markets. Intelligent trading agents, representing the individual organizations (*Organization Agents*) and the VO organizer (*VO Organizer Agent*) engage themselves in a negotiation process by exchanging proposals and counter-proposals, trying to convince opponents to modify their bidding values. Two important features of agent-mediated electronic commerce are dealt with in this negotiation process that leads to VO formation: multi-issue and adaptation.

The Multi-Issue Negotiation Protocol

In commerce transactions, goods under negotiation should be described along several issues. Each one of these issues has a relative importance to its owner, which should be taken into account during negotiation.

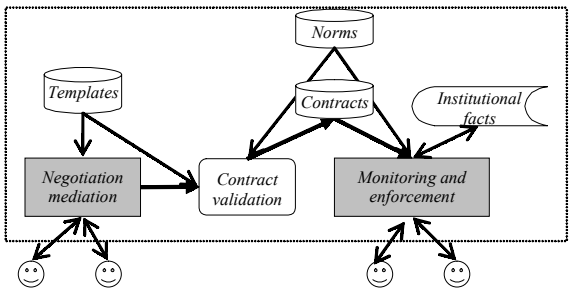
The *VO Organizer Agent* starts the process with an announcement and evaluates received bids. Bid evaluation is done through a multi-issue function that encodes the preferences for both the issues and issues' values, which are private to the agent that plays the role of *VO Organizer*. This multi-issue evaluation function is defined by the following formula:

$$Ev = \frac{1}{Deviation}$$

$$Deviation = \frac{1}{n} * \sum_{i=1}^{n} dif(PrefV_i, V_i)^i$$

where n is number of issues that defines a specific good. In the *Deviation* formula, parcels are presented in increasing order of pref-

Figure 4. Negotiation and contract execution



erence. The issue identified by number 1 (*i=1*) is the one that the *VO Organizer Agent* classifies as the least important, and issue *n* (*i=n*) is the most important. It seems intuitive that the *VO Organizer Agent* chooses a proposal that contains the values nearest to the preferable ones, for those issues the agent evaluates the most. The function $dif(PrefV_i, V_i)$ computes, for an issue *i*, the degree of acceptability of the current value (V_i) proposed by a specific *Organization Agent* when compared to the *VO Organizer Agent*'s preferred value (*PrefVi*).

The bid that presents the highest evaluation value is selected as the winner bid in the current round, since it is the solution that contains issue values closer to the preferred values. The winning bid in this current round is compared with the winning bid in all past rounds, and the best one is selected.

All other bids are compared with the winning bid in order to inform the respective agents about the reason why their bids were not selected. A comparison is made for the values proposed for the most important issues. These values are then classified in a qualitative way: low/high, very-low/very-high, or extremely-low/extremely-high. This classification is used as feedback to the agents that made the proposals.

Improving Offers by Learning

Another important characteristic that should be available in automated EC negotiation is the capability of learning, since it is naturally present in any commerce activity. Similar to what happens with human behavior, agents use past experience to influence future negotiations. Past proposals in a negotiation process can, and should, constrain the value of the next participant's proposal. The learning methodology we have chosen is a non-supervised learning called *reinforcement learning*. Reinforcement learning (RL) systems learn how to behave through trial-and-error interactions with its environment. RL is based on the idea of rewarding actions that produce good results and punishing those that produce bad results. Agents will use their experience to improve performance over time. "The idea that we learn by interacting with our environment is probably the first to occur to us when we think about the nature of learning" (Sutton & Barto, 1998).

We use the Q-learning algorithm (Watkins & Dayan, 1992). This is a reinforcement learning algorithm that maps state-action pairs to values, called Q-values. The *Organization Agent* receives a scalar evaluation from the environment for each action selected in a specific situation. When an agent in current state *s* performs action *a,* receives reward *r*, and arrives at next state *s'*, the correspondent Q-value is updated as follows (Sutton & Barto, 1998):

$$Q(s,a) = Q(s,a) + \alpha \left(r + \gamma \max_b Q(s',b) - Q(s,a) \right)$$

where $0 \leq \alpha \leq 1$ (learning rate) and $0 \leq \gamma \leq 1$ (discount factor). The discount factor is used to decrease the weight of reinforcements received in the future. It causes immediate reinforcements to have more importance than future reinforcements.

An agent is faced with a dilemma in order to choose what action to perform in each specific situation. Should it choose the most promising one? Or should it try to explore new states? If no exploration is made, the agent, although using a greedy strategy, may miss the dynamics of the market. If too much exploration is made, the agent can make a lot of poor choices. It is necessary to ensure that sufficient exploration is made while choosing actions with high known Q-values; the so-called *Boltzmann exploration* (Sutton & Barto, 1998) equation allows us to deal with this issue.

We have adapted the Boltzmann exploration to our application domain, by not exploring

all the possible actions, but reducing the exploration to the set of actions that are constrained by means of the feedback to past bids.

In our application domain, a state is defined by the n-uple:

$$s = (V_1, V_2, \ldots V_n)$$

V_i is the value for issue i. The state will also represent the bid sent to the *VO Organizer Agent*.

We define an action by the n-uple:

$$a = (A_1, A_2, \ldots, A_n), A_i \in \{increase, decrease, maintain\}$$

When an *Organization Agent* receives a feedback message from the *VO Organizer Agent* to its previous bid (this bid represents the state s that results from performing action a_p in state s_p), it tries to formulate a new bid by executing the following steps:

- Calculate a reward value for the previous bid (that also defines the current state s), using the qualitative evaluation included in the received feedback message.
- Update the state/action pair Q-value ($Q(s_p,a_p)$) using the Q-learning formula presented above.
- Derive all new promising actions a' taking into account the specific feedback message.
- Move to the state s^* with greatest reward value, that is, choose the action a^* with greatest probability value, according to the Boltzmann exploration formula (note that actions never tried before are stored in state s^* with Q-value=*default*).
- Perform action a^* by sending the new bid (state) out to the *Organizer Agent*.

This introduced negotiation protocol and adaptive negotiation strategy is detailed in Rocha et al. (2005). It has been tested with simulation experiments in simple scenarios of the textile industry domain.

Contract Monitoring and Norm Enforcement

Having a contract specified through a set of norms, we need appropriate mechanisms to monitor and enforce norm execution. Within the framework of an EI, monitoring and enforcement services can be rendered by the institution itself. Only a trusted third party can enable the necessary level of confidence between the parties involved in a business relation.

According to Vázquez-Salceda et al. (2004), an operational semantics for norms comes down to either: (1) defining constraints on unwanted behavior, or (2) detecting violations and reacting accordingly. In MAS, the absence of control over autonomous agent behavior leads us to the latter practice.

Monitoring the compliance of parties to norms touches some problems of deontic logic, namely the need to consider violations and their handling, as well as the inclusion of deadlines. These issues must be resolved in order to enable the implementation of norms in practical applications. As explained before, our approach consists of prescribing conditional norms with deadlines and defining sanctioning norms, also known as contrary-to-duty obligations (Jones & Carmo, 2001).

Since we essentially rely on obligations, verifying norm compliance consists of detecting the fulfillment and violation of obligations. For this, we define rules that are triggered by corresponding events: the achievement of obligations' institutional facts (for fulfillment) and the reach of deadlines (for violation).

Contextualized institutional facts are used to verify the fulfillment of obligations. For this, we define an obligation *fulfillment rule* appli-

cable to all contexts (that is, to any contractual relationship monitored by the EI):

[Context] ifact(IFact, T) ∧ obligation(Bearer, IFact, Deadline) ∧ T<Deadline
→ fulfilled(Bearer, IFact, T)

This rule indicates that if an institutional fact prescribed by an obligation is achieved before its deadline, then that obligation is fulfilled. As with behavior norms, literals within the rule are dependent on its context. That is, if an obligation within a certain contract is accomplished, the fulfillment of such obligation occurs, obviously, inside that same contract.

This rule is fundamental for enabling the chaining of obligations within a contractual relationship. It establishes a connection between the institutional facts that are added and the pending obligations.

The connection between behavior norms and sanctioning norms is achieved through violation detection mechanisms. These are based on *violation detection rules,* which fire when deadlines have elapsed. For this we consider time events, which are generated as institutional facts by a clock triggering mechanism, corresponding to the time when obligations are due.

[Context] ifact(time, Deadline) ∧ obligation(Bearer, IFact, Deadline) ∧ not(fulfilled(Bearer, IFact, _))
→ violated(Bearer, IFact, Deadline)

This violation detection rule states that in any context, if a deadline referring to an obligation was reached, and such obligation was not fulfilled, then a violation occurred (in the same context).

These two rules comprise the core of our normative environment, enabling the chaining of norms. The implementation of such a framework is achievable using a rule-based engine, like Jess (Friedman-Hill, 2003), which enables a forward-chaining approach based on the occurrence of events.

DISCUSSION

The concept of virtual enterprise or virtual organization has been applied to many forms of cooperative business, such as outsourcing, supply chains, or temporary consortiums. We are mostly concerned with the latter case, considering flexible and dynamic relationships. Their temporary nature requires quick set-up phases, allowing the VO to start operating as soon as possible.

Taking this into account, tools are being developed that allow for automating part of the process of setting up VOs. These include negotiation mediation and contract formation services, as we have described in this chapter. We have also developed a model that addresses the operation phase of a VO, allowing the monitoring of parties' compliance to contractual commitments. Other developments would include regulating the structure of VOs by defining norms that specify when and how partners can leave or enter a given consortium, and when and how the consortium should terminate.

The approach presented in this chapter is inspired by our social interactions as humans. In order to regulate the real-world environment, we have created institutions that deal with specific aspects of our daily lives, giving us some sense of order. The Electronic Institution concept is proposed to regulate the interactions of computational agents, providing trust by enforcing appropriate types of behavior. Furthermore, we address the issue of commitment creation between individual agents, which form a VO that is regulated by norms enforced by the EI.

In the real world, when contractual obligations are assumed by parties, they are not automatically enforced. For this reason, in non-electronic contracts, contrary-to-duty structures are not common. The violation of an obligation entitles the offended party to invoke legal power on a court of law, which may prescribe a secondary obligation to be imposed on the prevaricator (Daskalopulu & Maibaum, 2001). Besides, parties are not willing to stipulate handling procedures for all possible circumstances, deferring them to when and if situations arise. In our framework, the use of default rules (Craswell, 2000) as institutional norms addresses this problem, while enabling the automation of contract monitoring and enforcement.

Another significant intricacy concerns how to approach situations in which sanctions are violated. In such cases, more restrictive measures are needed. Instead of prescribing further sanctions, we can constrain the behavior of agents, for example by preventing their access to institutional services. Other approaches consist of indirect punishment measures through reputation mechanisms.

Relational contract theory (Hviid, 2000) studies continuing relations that are naturally self-enforceable. Instead of a detailed enforceable contract based on a third party, a relational contract is based on repeated interactions and social norms, representing an informal agreement sustained by the value of future relationships. Formal contracts are preferred when establishing relationships between unknown parties. On the other hand, regular partners generally rely on implicit relationships, supported by trust and by the threat of withholding business from anyone who has broken a promise in the past. The information on agents' reputation may also be used, if not as a ruling out factor, at least when deciding the level of detail a contract should have.

The adoption of MAS in domains such as the B2B world is challenged by matters of user trust. It is arguable what kinds of tasks human decision makers will be likely to delegate to software agents. The delegation of business-critical tasks is not likely to be a reality in the near future. Therefore, we envision the use of agent-based infrastructures—such as our Electronic Institution environment—as supporting tools that assist contractual relationships. Besides facilitating contract formation, our framework also enables monitoring contract fulfillment, and may be used to alert real-world entities concerning their upcoming obligations, the consequences of their actions, and their partners' compliance.

REFERENCES

Boella, G., van der Torre, L., & Verhagen, H. (2005). Introduction to normative multiagent systems. *Proceedings of the 1st International Symposium on Normative Multiagent Systems,* Hatfield, UK (pp. 1-7).

Castelfranchi, C. (2000). Engineering social order. In A. Omicini, R. Tolksdorf, & F. Zambonelli (Eds.), *Engineering societies in the agents' world* (pp. 1-18). Berlin: Springer.

Castelfranchi, C., Dignum, F., Jonker, C., & Treur, J. (2000). Deliberative normative agents: Principles and architectures. In N. Jennings & Y. Lesperance (Eds.), *Intelligent agents VI: Agent theories, architectures, and languages* (pp. 364-378). Orlando, FL: Springer.

Clocksin, W. F., & Mellish, C. S. (1981). *Programming in Prolog.* Berlin: Springer-Verlag.

Conte, R., Falcone, R., & Sartor, G. (1999). Introduction: Agents and norms: How to fill the gap? *Artificial Intelligence and Law, 7*(1), 1-15.

Craswell, R. (2000). Contract law: General theories. In B. Bouckaert & G. De Geest (Eds.), *Encyclopedia of law and economics, volume III: The regulation of contracts* (pp. 1-24). Cheltenham: Edward Elgar.

Daskalopulu, A, & Maibaum, T. (2001). Towards electronic contract performance. *Proceedings of Legal Information Systems Applications, the 12th International Conference and Workshop on Database and Expert Systems Applications* (pp. 771-777).

Dignum, F., Broersen, J., Dignum, V., & Meyer, J.-J. (2005). Meeting the deadline: Why, when and how. In M. G. Hinchey, J. L. Rash, W. F. Truszkowski, C. Rouff, & D. Gordon-Spears (Eds.), *Formal approaches to agent-based systems* (pp. 30-40). Berlin: Springer-Verlag *(LNAI 3228)*.

Dignum, V., & Dignum, F. (2001). Modeling agent societies: Coordination frameworks and institutions. In P. Brazdil & A. Jorge (Eds.), *Progress in artificial intelligence: Knowledge extraction, multi-agent systems, logic programming, and constraint solving* (LNAI 2258, pp. 191-204). Berlin: Springer-Verlag.

Dignum, V., & Dignum, F. (2002). Towards an agent-based infrastructure to support virtual organizations. In L.M. Camarinha-Matos (Ed.), *Collaborative business ecosystems and virtual enterprises* (pp. 363-370). Sesimbra, Portugal: Kluwer.

Dignum, V., Meyer, J.-J., Dignum, F., & Weigand, H. (2003). Formal specification of interaction in agent societies. In M. Hinchey, J. Rash, W. Truszkowski, C. Rouff, & D. Gordon-Spears (Eds.), *Formal approaches to agent-based systems* (LNAI 3228, pp. 37-52). Berlin: Springer-Verlag.

Esteva, M. (2003). *Electronic institutions: From specification to development.* PhD thesis, Technical University of Catalonia, Barcelona, Spain.

Field, S., & Hoffner, Y. (2005). Dynamic contract generation for dynamic business relationships. In G. D. Putnik & M. M. Cunha (Eds.), *Virtual enterprise integration: Technological and organizational perspectives* (pp. 207-228). Hershey, PA: Idea Group Publishing.

FIPA (Foundation for Intelligent Physical Agents). (2002). *FIPA-ACL message structure specification*. Retrieved from http://www.fipa.org/specs/fipa00061/

Fischer, K., Muller, J., Heimig, I., & Scheer, A. (1996). Intelligent agents in virtual enterprises. *Proceedings of the 1st International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology,* London (pp. 205-223).

Friedman-Hill, E. (2003). *Jess in action*. Greenwich, CT: Manning Publications Co.

García-Camino, A., & Rodríguez-Aguilar, J.A. (2005). Implementing norms in electronic institutions. *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS2005)*, Utrecht, The Netherlands (pp. 667-673).

Giarratano, J., & Riley, G. (1998). *Expert systems: Principles and programming* (3rd ed.). Boston: PWS.

Hviid, M. (2000). Long-term contracts and relational contracts. In B. Bouckaert & G. De Geest (Eds.), *Encyclopedia of law and economics, volume III: The regulation of contracts* (pp. 46-72). Cheltenham: Edward Elgar.

Jones, A., & Carmo, J. (2002). Deontic logic and contrary-to-duties. In D. Gabbay (Ed.), *Handbook of philosophical logic* (pp. 203-279). Dordrecht: Kluwer.

Jones, A., & Sergot, M.J. (1993). On the characterization of law and computer systems: The normative systems perspective. In J.-J. Meyer & R. J. Wieringa (Eds.), *Deontic logic in computer science: Normative system specification* (pp. 275-307). Chichester, UK: John Wiley & Sons.

Knottenbelt, J., & Clark, K. (2005). Contract-related agents. *Proceedings of the 6th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA VI)*, London.

Kollingbaum, M. J., & Norman, T. J. (2002). Supervised interaction—Creating a web of trust for contracting agents in electronic environments. In C. Castelfranchi & W. Johnson (Eds.), *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1* (pp. 272-279). New York: ACM Press.

Lopes Cardoso, H., Malucelli, A., Rocha, A. P., & Oliveira, E. (2005). Institutional services for dynamic virtual organizations. *Proceedings of the 6th IFIP Working Conference on Virtual Enterprises (PRO-VE'05)*, Valencia, Spain.

Lopes Cardoso, H., & Oliveira, E. (2004). Virtual enterprise normative framework within electronic institutions. In M.-P. Gleizes, A. Omicini, & F. Zambonelli (Eds.), *Engineering societies in the agents world V* (pp. 14-32). Toulouse, France: Springer.

Lopes Cardoso, H., & Oliveira, E. (2005). Towards an institutional environment using norms for contract performance. *Proceedings of the 4th International Central and Eastern European Conference on Multi-Agent Systems,* Budapest, Hungary.

Pacheco, O., & Carmo, J. (2003). A role-based model for the normative specification of organized collective agency and agents interaction. *Journal of Autonomous Agents and Multi-Agent Systems, 6*(2), 145-184.

Rao, A. S., & Georgeff, M. P. (1995). BDI agents: From theory to practice. In V. R. Lesser & L. Gasser (Eds.), *Proceedings of the 1st International Conference on Multiagent Systems* (pp. 312-319). Cambridge, MA: MIT Press.

Rocha, A. P., Lopes Cardoso, H., & Oliveira, E. (2005). Contributions to an electronic institution supporting virtual enterprises' lifecycle. In G. Putnik & M. M. Cunha (Eds.), *Virtual enterprise integration: Technological and organizational perspectives* (pp. 229-246). Hershey, PA: Idea Group Publishing.

Russell, S., & Norvig, P. (2003). *Artificial intelligence: A modern approach* (2nd ed). Englewood Cliffs, NJ: Prentice-Hall.

Sallé, M. (2002). Electronic contract framework for contractual agents. In R. Cohen & B. Spencer (Eds.), *Proceedings of Advances in Artificial Intelligence, the 15th Conference of the Canadian Society for Computational Studies of Intelligence* (pp. 349-353). London: Springer.

Searle, J. R. (1995). *The construction of social reality.* New York: The Free Press.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction.* Cambridge, MA: MIT Press.

Vázquez-Salceda, J., Aldewereld, H., & Dignum, F. (2004). Implementing norms in multiagent systems. In G. Lindemann, J. Denzinger, I.J. Timm, & R. Unland (Eds.), *Multiagent system technologies* (pp. 313-327). Berlin: Springer.

von Wright, G. (1951). Deontic logic. *Mind, 60,* 1-15.

Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine Learning, 8*(3/4), 279-292.

Wieringa, R. J., & Meyer, J.-J. (1993). Applications of deontic logic in computer science: A concise overview. In J.-J. Meyer & R. J. Wieringa (Eds.), *Deontic logic in computer science: Normative system specification* (pp. 17-40). Chichester, UK: John Wiley & Sons.

Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review, 10*(2), 115-152.

KEY TERMS

Agent: A goal-directed entity capable of interacting with the surrounding environment in an autonomous fashion; a software program representing an individual business entity.

Contract: An explicit formulation of agents' commitments.

Cooperation Agreement: A contract with an ongoing nature, regulating a virtual organization relationship that exists for a period of time.

Electronic Institution: A software platform providing a set of contract-related services for agent interaction and a normative environment for enforcing contracts.

Multi-Agent System: A system where multiple agents interact.

Norm: A prescription of behavior.

Normative Multi-Agent System: A set of interacting agents whose behavior can be regarded as governed by norms.

Role: A set of normative expectations and empowerments.

Virtual Organization: A consortium of organizations (enterprises) that align their businesses to perform integrated and more complex activities.

Section VIII

Marketing

Chapter LII
Co-Evolving Better Strategies in Oligopolistic Price Wars

Robert Marks
Australian Graduate School of Management, Australia

David Midgley
INSEAD, France

Lee Cooper
UCLA, USA

ABSTRACT

Using empirical market data from brand rivalry in a retail ground-coffee market, we model each idiosyncratic brand's pricing behavior using the restriction that marketing strategies depend only on profit-relevant state variables, and use the genetic algorithm to search for co-evolved equilibria, where each profit-maximizing brand manager is a stimulus-response automaton, responding to past prices in the asymmetric oligopolistic market. This chapter is part of a growing study of repeated interactions and oligopolistic behavior using the GA.

INTRODUCTION

We use simulated evolution to explore oligopolistic behavior in a (retail) market with up to four strategic sellers, comparing our simulation results with historical data derived from a retail market for ground, vacuum-sealed coffee beans. We find that our boundedly rational sellers perform well (as measured by their average weekly profits) compared to their historical counterparts, despite their limited memory and constrained marketing actions.

Significant features of our work are: first, our agents are heterogeneous: they respond idiosyncratically to others' actions, they have distinct costs, face distinct demand curves, and so earn distinct profits. For this reason, we cannot ignore the identities of the separate players, which would be convenient, were the players identical. Second, we use the genetic

Table 1. The nine brands: Average price and market share

Brand	Price	Market Share
Folgers	$2.33	21%
Maxwell House	$2.22	20%
Chock Full O' Nuts	$2.02	11%
Maxwell House Master Blend	$2.72	10%
Chase & Sanbourne	$2.34	4%
Hills Bros.	$2.13	4%
Yuban	$3.11	1%
All Other Branded	$1.96	3%
All Other Private Labels	$1.95	27%

algorithm (GA) to model the players' learning. To avoid "social learning" (Vriend 2000), when players drawn from a single population pass information to their "offspring" through the genotype (an extra-market mechanism), we use distinct populations for the four strategic sellers, which precludes extra-market communication and learning. Third, we use stochastic sampling (commonly know as Monte Carlo sampling; see Judd, 1998) to generate a distribution of marketing behaviors across the sellers: given the stochastic nature of the GA, and the complexity of the genotypes and phenotypes, we use distinct random seeds to generate 50 distinct outcomes.

Computer scientists have developed machine learning, such as the GA (Holland, 1976, 1992; Mitchell, 1996; Goldberg, 1989) and classifier systems (Holland, 1976, 1992) as means of optimizing—of finding the argmax of functions not amenable to calculus-based methods of solution. Social scientists have used and developed these tools (Marks, 1989, 2002; Arifovic, 1993), but less as optimizers and more as generators of "adaptive plans" or "structures that perform well" in complex systems (Holland, 1975, 1992), by modeling adaptive economic agents (Holland & Miller, 1992) that interact. This chapter demonstrates use of the GA in this spirit.

Table 2. Asymmetries of the four strategic brands

	Own-Price Elasticity of Market Share	*AVC* ($/lb.)
Folgers	–4.4	$1.39
Maxwell House	–3.9	$1.32
Chock Full O' Nuts	–4.7	$1.19
Hills Bros.	–0.5	$1.18

OLIGOPOLY THEORY

Rivalry among retail brand managers in a market for vacuum-sealed ground coffee beans can be seen to possess characteristics that clearly reflect the oligopolistic nature of the repeated interaction: the brands are seen as imperfect substitutes by the buyers; the sales of any one brand, if stimulated by heightened marketing actions, will negatively impact on the sales of other brands, and there is no single going market price for coffee. We model Bertrand asymmetric competition among firms, competing with price (and other marketing actions) rather than quantity.

We have access to 78 weeks of supermarket-scanner market data for a city in the U.S. Midwest by supermarket chain. The marketing actions (price, coupons, aisle display, advertising) remain unchanged for seven days, from midnight Saturday for all brands—a property that lends itself to simulation modeling on a digital computer.

One of us (Cooper) has developed a market model, Casper, which calculates, given all of the nine brands' marketing actions, the volume of sales of each brand, the brands' revenues, and profits (Cooper & Nakanishi, 1988).[1] The brands differ not only in the demand response of the market (each of their price elasticities of demand is distinct), but also in their costs. The brands are truly hereogeneous, as seen in Tables 1 and 2.

Casper provides the equivalent of the one-shot payoffs for each of the brands, modeled as playing a repeated game.[2]

Although each brand manager must choose the set of next week's market actions without knowing the other brands' actions next week, this and preceding weeks' actions are observable by all brands. So the brands can choose to remember the actions of their rivals for one, two, or more weeks. Their depth of memory is a measure of their bounded rationality: an unboundedly rational player would choose to forget nothing, and to use all remembered information including its weekly profits in deciding what marketing actions to undertake next week.

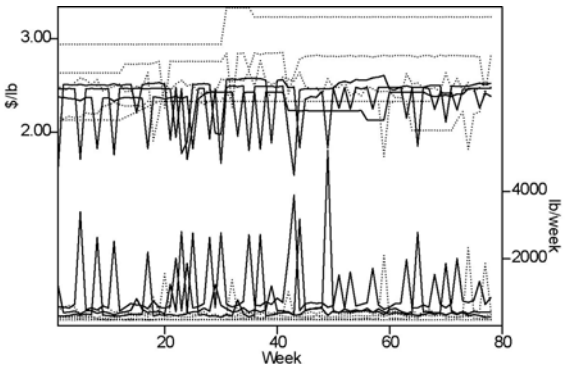
But the brand managers do not have unfettered freedom to choose their marketing actions, since the policies of the supermarket chain constrain them in two ways. Some actions (including a price well below the "shelf price") result in much higher sales and higher profits (the lower margins are more than offset by higher volumes of sales). The chain constrains use of these so-called "promotional" actions. First, no brand may use a promotional action set two weeks successively. Second, only one brand may use a promotional action set in any week. The chain acts as the moderator among the brand managers, who each propose their next week's action set and acquiesce in the supermarket's choice of which brand to promote next week.

Competing against each other, the brand managers are trying to maximize their average weekly profits. The supermarket chain is competing against other chains for sales, although we do not model this rivalry explicitly here. Instead, we model the supermarket as trying to maximize "total category volume" of coffee sales. The reason is that coffee is one of many supermarket categories, but one that might attract more customers to the chain and so help to sell higher volumes across many categories. We model supermarket moderation in several ways, as discussed in detail below.

The competition among brand managers is asymmetric, because each of the brands is distinct, with distinct price elasticities of demand, distinct unit costs of provision, and distinct responses to the market. Moreover, solution of the Nash equilibrium of the one-week game, let alone solution of the Nash equilibria in the repeated game, is not amenable to calculus-based, closed-form techniques.[3] There are nine brand rivals in the chain we focus on, although only four are engaged in what we might call a "rivalrous dance" by altering their marketing actions every week. Figure 1 shows the behavior of the three major strategic brands, and six minor ones.

There are two main purposes of our research. First, we wish to calibrate and validate our model's behavior to the historical data. To this end, we use the asymmetries implicit in Casper to model the brands' sales, revenues, costs, and profits in any week, given all brands' market actions that week. We allow the model to run for 50 weeks, with up to four "strategic brands" altering their marketing actions from week to week, in response to the state of the market (defined as the set of all players' mar-

Figure 1. Weekly prices and sales (solid lines: Folgers, Maxwell House, Chock Full O' Nuts)



keting actions) the previous week. We look for several measures of the simulated competition: weekly profits, weekly Total Category Volume of coffee sales, and the marketing actions employed by the four strategic players.

The marketing actions include price, coupons, aisle display, and flier advertising. Historically, brands' prices varied from $1.50/lb. to $3/lb., with promotional prices below $2.25/lb. Coupons reduce the price paid at check-out, and are measured by percentage of stores in the chain that distribute coupons for that brand that week. We net the impact of coupons out of the retail price to simplify the action space. Similarly, aisle display and flier advertising are reported as percentage of stores in the chain that include them for any brand in any week. In practice, as discussed above, the store permits only one brand to promote itself any week, and we see a consistent pattern in coupons, aisle display, and flier advertising: only one promoted brand per week.

We could allow the adaptive brand managers of the model to choose their price from any between 150 and 300 cents per pound, and any percentage of aisle displays and flier advertising, but in practice we believe, first, that this degree of freedom is not necessary to replicate historical performance, and second, that the practical difficulties of simulating this (such as a huge number of degrees of freedom in the definition of "market state," and the need to execute Casper each simulated week instead of using a much faster compiled look-up table) militate against it.

Instead, we use the historical data to identify, first, four sets, and second, eight sets of brand-specific actions which are representative of those chosen over the first 50 weeks of data. Later, we use eight action sets that are identical across the four strategic brands and find similar results.

The second purpose of our research is to see whether our boundedly rational artificial brand managers can surpass the performance of their historical counterparts, as measured by their weekly profits, handicapped as they are by, first, simple one-week memory, and two, constrained choice of marketing actions. Necessarily, since we do not have access to actual historical brand managers in order to pit them against our artificial brand managers in a laboratory setting, we must be content with open-loop experiments, where our artificial brand managers respond to the unfolding history of past rivalries, but where the historical actions cannot respond to our artificial agents' actions. We argue below that both aims are attained.

The structure of the chapter is as follows. After a discussion of the GA, we describe our historical market data, and then describe the results of a set of computer experiments, as we increase the number of strategic brands from three to four, and the number of possible marketing actions per brand from four to eight. We present the open-loop results of playing our best co-evolved artificial brands against history, and introduce the Holyfield-Tyson effect of pitting more evolved agents against less evolved agents. We discuss the implications of our results for insights into managerial learning.

BORROWING FROM NATURE: THE GENETIC ALGORITHM

Axelrod (1987) modeled players in his discrete repeated prisoner's dilemma (RPD) game as stimulus-response automata, where the stimulus was the state of the game, defined as both players' actions over the previous several moves, and the response was the next period's action (or actions). That is, he modeled the game as a state-space game (Fudenberg & Tirole, 1992; Slade, 1995), in which past play influences current and future actions, not because it has a direct effect on the game environment (the payoff function), but because all

(or both) players believe that past play matters. Axelrod's model focused attention on a smaller class of "Markov" or "state-space" strategies, in which past actions influence current play only through their effect on a state variable that summarizes the direct effect of the past on the current environment (the payoffs). With state-space games, the state summarizes all history that is payoff relevant, and players' strategies are restricted to depend only on the state and (perhaps) the time.

We have been using versions of the GA since 1988 to explore oligopolistic behavior.[4] As we describe above, we model the artificial brand managers as stimulus-response automata, in effect, where the stimulus is this week's market state (defined by the marketing actions of all players, and particularly the four strategic brands) and the response is the brand's proposed market actions next week. The eventual market actions per brand are the outcome of a moderating process performed by the supermarket chain, responding to the proposals of the four brand managers.

We use the GA to search simultaneously for better automata for each of the four strategic brands, using their weekly profits as a measure of performance or fitness. Each brand manager is modeled as a binary string. If there are eight possible marketing actions to choose from (correlating aisle display and flier advertising with promotional prices), then we can use three bits on the string to code for next week's marketing action. How many triples are sufficient for the model? With four strategic players, each with eight possible marketing actions, there are a^{mp} possible states (Midgley, Marks, & Cooper, 1997), where a = the number of actions (8), m = the number of weeks remembered (1), and p = the number of strategic players (4), a total of 4,096 possible states, each state mapping to a triple of bits on the artificial player's bit-string "chromosome," which requires each string to be 12,288 bits long. Adding an additional 12 bits for the "phantom memory" at the first of the 50 weeks (to endogamies the initial conditions of the brand's belief in the previous week's market state) gives us 12,300 bits per string. This work is a generalization of Axelrod (1987) and Marks (1989), and uses the ability of the GA to search the highly disjoint space of strategies, as Fudenberg and Levine (1998) have suggested.

As is well known (see Goldberg, 1989; Mitchell, 1996; or the second edition of Holland, 1992), the GA borrows from our understanding of evolution to search for solutions to problems not easily solved otherwise. An initial population of solutions is generated, the fitness score of each individual is determined, a subset of individuals is selected to be the "parents" of the next generation, the "crossover" of pairs of parents is simulated, and each bit is flipped from zero to one or vice versa ("mutated") with a small probability (here 1%). The fitness of each member of the new population is determined. And the process repeats until convergence.

The GA has been used by engineers as an optimization tool. Social scientists have used it in a slightly different way: as a means of simulating co-evolution. In our model, each brand manager learns from its rivals' behavior and from its rivals' responses to its own actions. This mutual leaning means that the competitive environment changes, even as each artificial brand manager learns to compete more effectively. As a result, there is no necessary increase in weekly profits, even as the GA winnows the succeeding generations of their worst performing strings.

Co-evolution requires a separate population for each of the strategic players.[5] A single population would allow extra-market communication and learning to occur via the genetic operations of selection and crossover. Not only would this be illegal under antitrust laws, but such social learning (Vriend, 2000) is not what

we want to model. Necessarily, four separate populations require a much more complex GA program, but only a co-evolving GA is appropriate. We extensively rewrote the GA software (GAucsd, based on John Grefenstette's GENESIS package; Schraudolph & Grefenstette, 1992) to allow the simultaneous simulation of up to four populations of agents (modeled as bit strings).

We use a population size of 25, each string being 12,300 bits long, with four populations.[6] This is a non-trivial simulation, but we manage to obtain 2,500 generations, each of 5.5 million weekly interactions, every 50 minutes on a Mac G5 dual-2Ghz Unix workstation.

THE HISTORICAL DATA: THE RETAIL GROUND-COFFEE MARKET

The data refer to a local U.S. retail market for ground-caffeinated coffee. There are nine brands or players. Table 1 gives the average prices ($/lb.) and market shares for each of the nine. Table 2 presents further data on the heterogeneity of the strategic players: their own-price elasticities of market share and their average variable costs (AVC). Figure 1 shows the historical prices (top half) and quantity of sales (bottom half) by brand over 75 weeks. The solid lines map the prices and sales of the three strategic brands: Folgers, Maxwell House, and Chock Full O' Nuts; the dotted lines map the other brands. The data are aggregated on a supermarket chain. As mentioned above, each marketing action comprises four "marketing instruments":

1. prices (the price that week of the brand);
2. flier features (the percentage of stores in the chain featuring the brand's item in their distributed advertising);
3. in-store aisle displays (the percentage of stores in the chain featuring the brand's item as an aisle display); and
4. coupons, which are distributed to households in the district, for redemption of the brand's product at the supermarket chain. We adjusted the price in any week by the percentage of coupons distributed.

COMPUTER EXPERIMENTS

We model the brand managers as artificial agents. The computational experimenter can control the agents':

- information (what they know when);
- learning (how information about their own and others' behavior alters their future responses);
- degree of bounded rationality (in particular, their memory of past weeks' actions and outcomes, perhaps aggregated into coarser partitions);
- sets of possible actions (their deterministic responses to the perceived state of the market); and
- payoffs (which, like their information, learning, memory, partitioning, and actions, are asymmetric).

Simulation, although it cannot in general establish necessity, does enable exploration of the *sufficient* conditions for the emergence of particular aggregate market phenomena, given players' micro-behavior.

First Results

This chapter builds on work reported in Midgley et al. (1997). There we considered the three most interactive players in the market: Folgers, Maxwell House, and Chock Full O' Nuts

Table 3. The four sets of actions of the three strategic brands

A	Folgers			Maxwell House			CFON		
	P ($/lb.)	F (%)	D (%)	P ($/lb.)	F (%)	D (%)	P ($/lb.)	F (%)	D (%)
p_1	1.87*	95*	69*	1.96*	95*	69*	1.89*	100*	77*
p_2	2.07	83	0	2.33	83	0	2.02	100	65
p_3	2.38	0	0	2.46	0	0	2.29	0	0
p_4	2.59	0	0	2.56	0	0	2.45	0	0

Note: Asterisked actions are subject to store moderation. A is Action, P is Price, F is advertising Feature, D is aisle Display.

(CFON). We allowed each agent four action sets, as derived from an analysis of their historical prices and other marketing actions. Table 3 shows the four possible action sets for each of the three agents.

Our intention was to pit the three strategic brands against each other, while the other brands were unchanging or non-strategic players, in order to examine the co-evolution of the three agents' behavior. We would need to distinguish convergence of behavior (phenotype) from structure (genotype).

We used the Casper market model to derive the three asymmetric $4\times4\times4$ payoff matrices for the three strategic players. The payoff matrix indicates any brand's weekly profit for each of the 64 combinations of price given in Table 3, given the non-strategic prices of the other six brands ($/lb.) (see Table 4).

With one-week memory, the agents were modeled as bit strings of length $2\times4^3+6=134$ bits. (The 6 bits of phantom memory endogenize initial conditions: each agent has four possible actions coding to 2 bits, and there are three strategic players.)

Table 4. The fixed prices of the other six brands (%/lb)

Master Blend	Hills Bros.	Yuban	C&S	AOB	APL
2.90	2.49	3.39	2.39	3.68	2.19

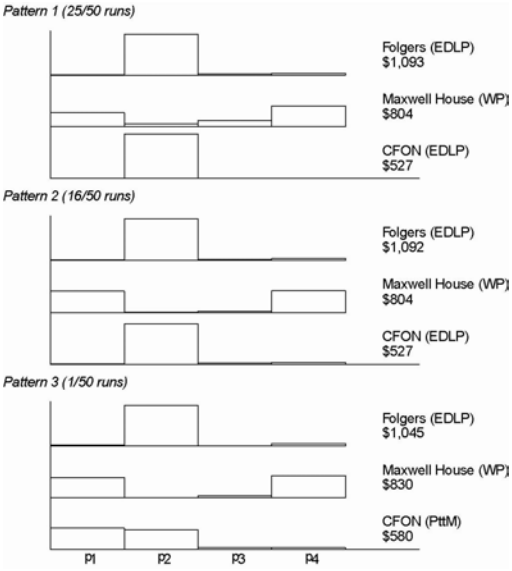
Each agent played a 50-round game with each possible combination of the other two players. The GA used 25 mappings (or strings) per population for each agent. Therefore, testing each generation required 8,125 50-round games, or 325 games per string per generation. Each agent had complete information of all previous actions in each 50-round game, but not others' weekly profits (payoffs).

Figure 2 shows three patterns and average weekly profits with three distinct populations. For most of the runs, the agents' behavior is very similar (Folgers and CFON pricing at an Every Day Low Pricer (EDLP); Maxwell House exhibiting Wide Pulsing (WP). In Pattern 3, CFON is exhibiting Promote to the Max (PttM).

Consult Midgley et al. (1997) for a discussion of the patterns of behavior of the unconstrained and constrained brands, and the issue of demand saturation over time that the single-week estimates of Casper evoke. After constraining the brands (as discussed above) and accounting for demand saturation, our three-brand, four-action model generates patterns of behavior similar to Figure 1: brands alternate (roughly) in pricing at p_1, while the other two price at p_2, p_3, or p_4.

Having co-evolved populations of each of the three strategic agents over 100 generations, we decided that one way to demonstrate the extent to which the agents had learned to act effectively was to use the most profitable agent

Figure 2. Three agents, four actions



by brand from the hundredth generation and play it against the history of play of the other strategic brands. In order to do this, we had to partition the historical actions into four intervals for each of the three strategic brands. We measured performance by the average profits over the 75-week history.

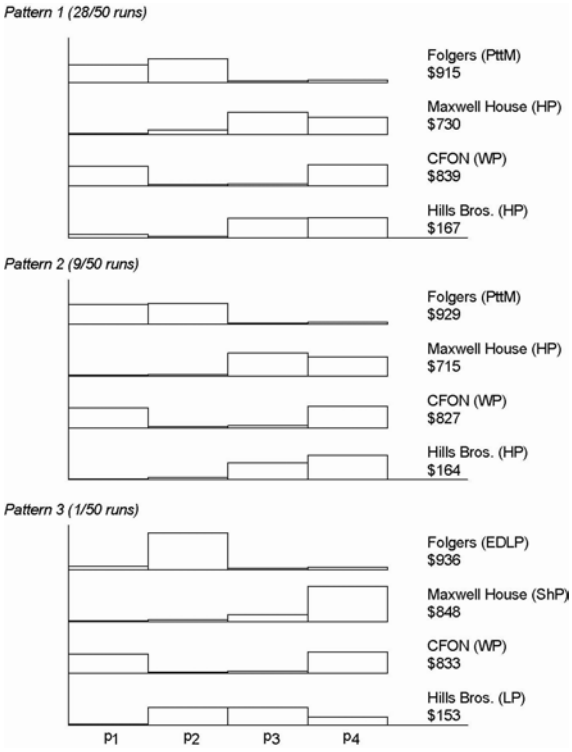
For Folgers and CFON, the agents improved on their historical performance, but Maxwell House sometimes did worse, even on average. But this was an "open-loop" simulation: the historical managers had responded to the historical actions of *all* others, but here could not respond to the agents' actions. Nonetheless, our very simple agents generated reasonable performance in a noisy environment.

Four Strategic Players

Previously, we modeled the oligopoly with three strategic players, each with four possible actions, remembering one week back. As discussed above, the agents were modeled as bit strings of length 134 bits. To improve the realism of the simulation, we increase the number of strategic brands to four, by including Hills Bros. This increases the bit-string length from 134 bits to 520 bits.[7] We chose Hills Bros., despite its small market share, as the fourth strategic agent, because the fourth largest brand (Master Blend) is not independent of Maxwell House, and so its strategic actions could be orchestrated by the owner.

The results of introducing the fourth strategic brand are striking. Even though Hills Bros. has a small market share (4%), its introduction is quite significant. The market changes in significant, complex, and asymmetric ways. There are changes in the other brands' behavior as well as in other brands' average weekly profits. Figure 3 shows three patterns and weekly profits that comprise 38 of 50 Monte

Figure 3. Four agents, four historical actions—hundredth generation



Carlo runs. The new strategic agent apparently takes up some of the fixed number of opportunities for major promotions and has differing competitive impacts on the other brands. Surprisingly, the total weekly profits of the first three brands rise when a fourth player is introduced, at least for the 40-odd patterns of Figures 2 and 3. What these simulations demonstrate is that a small player (as measured by market share) is not necessarily insignificant strategically. In Pattern 1, Maxwell House is exhibiting High Pricer (HP), and in Pattern 3, Shelf Price (ShP).

Eight Actions per Player

Heretofore the strategic agents (whether three or four) have been constrained by the four possible actions, chosen from the historically observed actions of the actual brand managers. In effect, the agents were given a choice of pricing high or low, with minor variation around the two positions, and they were constrained by the corporate memory and prior learning of the actual brand managers, who had, we assume, learned not to price too high (and sell very little) or too low (and earn little and perhaps spark a price war).

We wanted to increase the choices of the agents. The simplest way was to double the number of possible actions per agent from four to eight. The effect of this on the bit-string length will depend on the number of strategic agents: for three agents, with one-week memory, allowing eight possible actions instead of four increases the length from 134 bits to 1,545; for four agents, the length increases from 520 bits to 12,300 bits.[8]

By increasing the number of actions to eight, we hoped to give our agents the opportunity to demonstrate that the four actions used earlier were robust, and that our assumption of a mature oligopoly was correct, at least in terms of the combinations of prices and other marketing actions encountered.

Moving to eight possible actions, especially including some beyond the observed range of actions of the historical brand managers, introduces the possibility of the agents learning anew what was embodied in the historical range: not to price too high or too low.

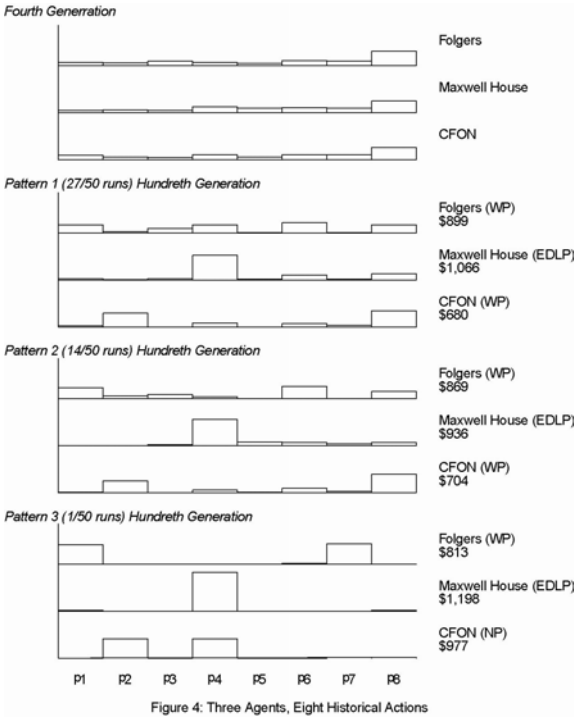
Figure 4 shows the weekly profits and patterns of behavior, as reflected by the frequency of actions across the three strategic agents. The data refer to 50-run Monte Carlo simulations. (The black diamonds ♦ in the figures correspond to the asterisks in Table 5: actions subject to store moderation.)

After four generations, starting from a uniform distribution of actions (because the bit

Table 5. Four brands: Sets of eight possible marketing actions

A	Folgers			Maxwell House			CFON			Hills Bros.		
	P ($/lb.)	F (%)	D (%)	P ($/lb.)	F (%)	D (%)	P ($/lb.)	F (%)	D (%)	P ($/lb.)	F (%)	D (%)
p_1	1.62*	67*	67*	1.60*	97*	97*	1.64	0	0	1.86*	100*	74*
p_2	1.83*	97*	96*	1.87*	94*	91*	1.89*	97*	97*	1.91	0	73
p_3	1.96	0	0	2.06*	88*	76*	1.89*	98*	29*	1.95*	100*	87*
p_4	2.03*	79*	77*	2.33	79	0	2.01	0	0	2.09*	100*	0*
p_5	2.04*	85*	0*	2.38	54	0	2.02*	97*	62*	2.19	0	0
p_6	2.22	96	33	2.52	0	0	2.31	0	49	2.42	0	0
p_7	2.57	0	0	2.53	0	53	2.33	0	0	2.49	0	100
p_8	2.78	0	0	2.59	0	13	2.49	0	0	2.56	0	14

Figure 4. Three agents, eight historical actions



strings are chosen randomly to begin with, apart from filtering against the actions of promoting two weeks in succession), we see that the frequencies of actions are still almost uniform. After 100 generations, however, the agents have focused on only two or three main patterns of interaction, with many fewer than eight possible actions used frequently: agents have *co-learned* the two or three actions that are most profitable, given others' behavior. The actions are brand specific.

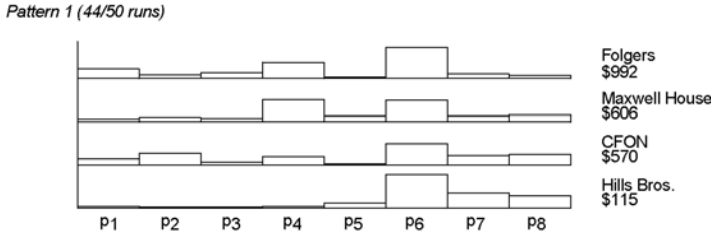
Specifically, with three strategic agents: CFON is pulsing between Shelf Price (high) and Promotional Price (low). Folgers exhibits three pulsing patterns: P2—pulsing three actions; P1—more diverse pulsing, with four actions; and P3—pulsing with two actions. Maxwell House exhibits a less dynamic choice of Every Day Low Price, and avoids the store constraints. CFON is pulsing with two actions: wide or narrow.

From a 50-run Monte Carlo simulation of four agents and eight possible actions, we observe in Figure 5 for 44 runs that the four agents exhibit different behavior: Folgers and CFON show Wide Pulsing, from high to low, promotional prices (indicated by the black diamonds), but Folgers, with 42% of its actions promotion (of a possible maximum of 50%) is almost Promoting to the Maximum, whereas CFON is promoting only 22% of the time; Maxwell House shows High Pulsing, seldom (15%) promoting at low prices; and Hills Bros. shows Shelf Price (p_6) or higher, promoting only 8% of the time.

Overall, we can say that, with the eight possible actions of Table 5, a greater degree of homogeneity emerges, with 44 of 50 Monte Carlo runs being identical. Moreover, adding a fourth strategic agent increases the degree of competition in the market, which is here reflected in lower average profits for the first three brands, as well as different behavior.

Moderation in the runs of Figure 5 is achieved randomly (by a "zero-intelligence" chain moderator), but we explored changing

Figure 5. Four agents, eight historical actions—2500th generation



this in two ways: first, by altering the possible actions of Table 5 by eliminating the lowest prices, and second, by estimating from the historical data just how moderation was achieved and the chain's preferences across brands revealed. We do not report these experiments in detail here, but brands' profits fell, as did the volume of coffee sold.

When we repeated the open-loop plays between the best of the co-evolved three agents with eight possible actions and the historical brand managers, we found that the best agents clearly outperformed their historical counterparts: for Folgers by 156%, for MH by 32%, and for CFON by 42%.

- **The Frankenstein Effect:** Agents that showed only a few behaviors in the co-evolutionary "lab" were able to evince a wider repertoire when faced with a more variable environment (the history of actual managers' behavior). We dub this the Frankenstein effect because the artificially bred agents were more interesting in the wild than in the lab.
- **The Holyfield-Tyson Effect:** The artificial agents "learn" through application of the evolutionary techniques of the GA. This is clear when the agents are solutions to a static problem, as has been the most usual application of GA techniques in, say, engineering. It is also the case that the first application of GAs in economics (Axelrod, 1987) was static, even if stochastic: Axelrod used GAs against a non-evolving but mixed-strategy niche of algorithms derived from the early computer RPD tournaments (Axelrod, 1984). But Marks (1989) and others following have bred artificial agents against each other, a process that Marks called "bootstrapping" and biologists term "co-evolution."

Against a static environment, progress of the artificial agents is readily revealed by their improving fitness scores, but against a dynamic environment composed of like artificial agents, scores may not rise from generation to generation. Two questions: Do highly co-evolved players become effete? Will a "naïve" outperform a "sophisticate"?

Apart from the growth in average weekly profits, there are at least two further ways to demonstrate that the artificial natural selection has improved the agents' performances. In our earlier work we attempted to show the greater competence of our artificial agents by pitting them against the historical histories of play of their opponents, but some criticism has been made that this overstates the skills of the artificial agents and understates the skills of the historical agents, who have no opportunity to respond to the actions of the artificial agent: their plays are given, or open-loop.

Here we attempt to show how the artificial agents have learned by taking agents after 2,500 trials (100 generations) and playing them against not the frozen moves of their historical opponents, but the agents after only 200 trials (8 generations)—a process we have termed pitting a sophisticated agent against naïve agents. How do we show that the co-evolved agents are learning to respond better (are truly fitter)? Previously we considered the mean weekly profits; now, in turn, we replace the best naïve (at 8 generations) Folgers (respectively, Maxwell House and CFON) string with the best sophisticate (after 100 generations) Folgers (respectively, Maxwell House and CFON) string.

The procedure followed was:

1. After 8 generations, identify the best string from each of the 3 or 4 populations.
2. Play these 3 or 4 against each other for a 50-week repeated game; note average weekly profits.

Table 6. Performance of hundredth-generation agents competing with each other

Experiment	Folgers	Maxwell House	CFON	Hills Bros.	Total
3 pop., 4 actions	1,053	793	534	n/a	2,380
3 pop., 8 actions	889	985	694	n/a	2,568
4 pop., 4 actions	915	729	835	164	2,479
4 pop., 8 actions	992	606	570	115	2,284

Note: Average weekly profits computed from 50 Monte Carlo simulations and all combinations of agents. Historical-action sets.

3. Allow the 3 or 4 populations to continue co-evolving via the GA.
4. After 100 generations, identify the best strings from the 3 or 4 populations, play them against each other as before; note average weekly profits. Table 6 shows these results.
5. Replace the best Folgers string after 8 generations with the best Folgers string after 100 generations (i.e., replace the best primitive string by the best sophisticate).
6. Play all combinations of 3 or 4 strategic brands, and consider string-by-string the change in average weekly profits with the sophisticated player and without the sophisticated player in one brand.
7. Repeat steps 5 and 6 for the remaining 2 or 3 strategic brands. Repeat steps 1-7 50 times. Table 7 shows the performances.

Table 8 shows the three combinations of results.

We would have expected positive diagonals (i.e., that sophisticates do better) and negative off-diagonals (i.e., that others' profits fall). Instead, we see that the CFON sophisticate is the only one to improve on the replaced naïve's performance. In the cases of Folgers and Maxwell House, the sophisticates did worse than the naïve.

Table 7. Performance of best agents competing with the managers' histories

Experiment	Folgers	Maxwell House	CFON	Hills Bros.
Historical actions	188 *a*	198 *a*	69 *a*	?
3 pop., 4 actions	410 *b*, 468 *c*	271, 329	107, 113	
3 pop., 8 actions	523, 806	295, 514	104, 124	
4 pop., 4 actions	430, 469	191, 286	103, 111	?
4 pop., 8 actions	481, 944	262, 559	98, 110	12, 13
(60th gen.)				

a—average weekly profits computed from historical actions; b—average weekly profits computed from playing the best agents from 50 Monte Carlo simulations against historical actions; c—single best performance observed. Note: The profits derived from historical actions will not be the same as single-period Casper results because of the demand-saturation constraint.

The results of Table 8 are unexpected. One possibility is genetic drift, a phenomenon where lack of selective pressure on many alleles (sites) on the bit strings (because of convergence of behavior, generation after generation, which means that only a small subset of possible states occur, and hence only a small subset of alleles (sites) are triggered) means that those bits may, through chance and recombination, flip, which is only obvious when, in the hurly-burly of rivalry against the naïve, these states are encountered again, after many generations, and the perhaps effete sophisticates do not

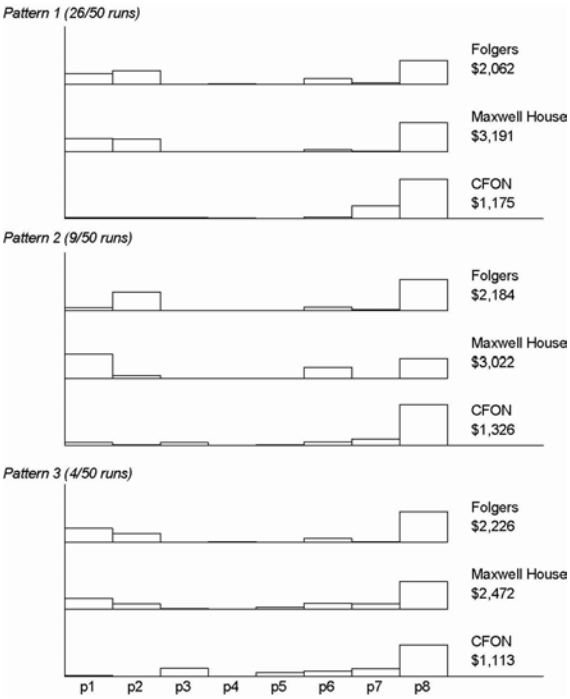
Table 8. Mean changes in average profits with the best sophisticates

	ΔF	ΔMH	ΔCFON
Folgers	–15.0	41.4	42.0
MH	2.0	–20.0	37.8
CFON	13.9	–29.0	82.3

always cut the mustard. We have dubbed this the Holyfield-Tyson effect after the notorious championship bout between the two heavyweights, in which Tyson bit off part of Holyfield's ear.[9]

Genetic drift is inversely proportional to the number of individuals in the population. We increased the population size per brand from 25 strings to 250. This led to very slow convergence, even with the short strings in the three-agent, four-action simulations: not only was there a thousand-fold increase in the number of three-way interactions per generation, but there was apparently lengthy spiraling towards convergence of the GA—only a single run was performed, not a Monte Carlo. The GA was still converging at 80 generations, and the results after 160 generations were no better: the GA had still not converged. We cannot confirm genetic drift as an explanation. Another possibility is the Red Queen effect (Robson, 2005).

Figure 6. Three agents, eight-random-action sets—hundredth generation



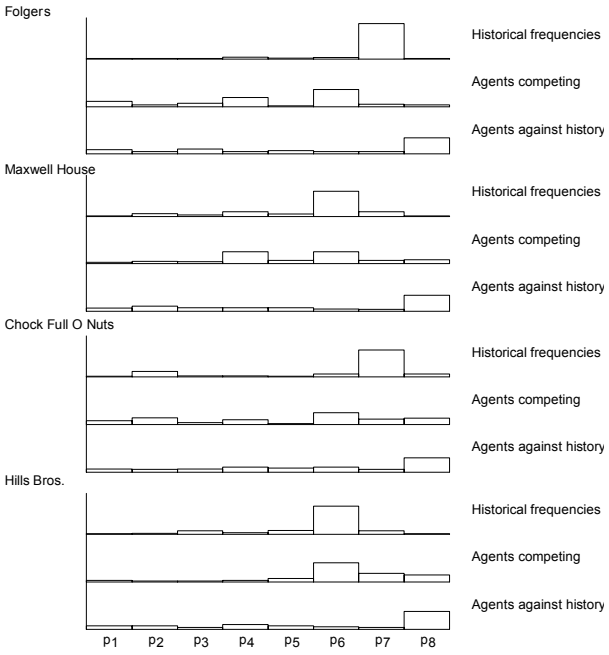
MANAGERIAL LEARNING

The eight-action sets per player of above were derived from historical actions and so embodied prior learning What if we give the artificial agents a different repertoire of actions—one developed without reference to the historical actions of managers? We used a random experimental design, where the price per pound is stepped in 10-cent increments between $1.60 and $2.80, and feature and display can take on the values of either 0 or 100%.

Figure 6 shows three patterns that accounted for 39 of 50 Monte Carlo runs. Note that average weekly profits are much higher than with historical, learned action sets. Note too that in general the agents shun low-price promotions and maintain high prices throughout most interactions. The levels of competition are much lower than with historical-action sets—with these randomly chosen action sets, the agents are engaging in the sort of collusion that we expected to see in the first simulations above. But we speculate that these results show that inter-chain competition is what our model (and Casper) lacks—the demand curve for coffee from our supermarket chain must be kinked when potential customers go elsewhere to avoid paying the high prices our artificial agents would like to charge in implicit collusion.

Results of three-player, eight-possible-action simulations reveal two major patterns: much higher average weekly profits, and almost no low, feature pricing, with profits earned at very high pricing. This result is seen in Figure 7, which shows the patterns for the four strategic players under the three regimes: historical frequencies of the brand managers, co-evolved

Figure 7. Comparison of patterns



agents competing against each other, and the best co-evolved agents competing against history. Notice that for Maxwell House and Hills Bros., the co-evolved agents' frequencies of actions are very similar to the historical brand managers' frequencies of actions; and for Folgers and CFON, the two patterns are similar, with a slightly higher shelf price for the historical managers.

CONCLUSION

We can summarize our experiments on rivalry in a mature differentiated Bertrand oligopoly in two ways: the average weekly profits of the agents, and the patterns of actions. Table 6 summarizes the average weekly profits of the four strategic brands under the different combinations of strategic brands and four- or eight-action sets (all derived from the historically observed actions of the brand managers). Figure 7 summarizes the frequencies of chosen actions (eight-action sets, derived from the historically observed actions) under the three conditions of: (1) historical actions (from Figure 1), (2) co-evolved agents competing (from Figure 5), and (3) agents competing against history (playing the 50 best agents per brand against the historical actions of their three competitors). The competitive behavior of one of our artificial brand managers (Hills Bros.) is similar to the historical frequencies, but the other three artificial brands reveal more strategic behavior than the historical brands engaged in. For at least one brand, a simple set of possible actions and one-week memory are sufficient to simulate historical behavior, suggesting a lack of sophistication on the part of historical brand managers. Later work will explore this issue of "zero-intelligence" behavior (or simple heuristics) further.

Our experiments have revealed some restrictions on the historical brand managers which were not immediately apparent, but more significantly, we have shown that the patterns of interaction among the brand managers were not as profitable as they might have been, even if all strategic players in the oligopoly had been using strategies as finely tuned as our agents had learned to use, in the simulations learned using the GA. We hypothesize that the techniques used here could shed light on the behaviors in similar asymmetric oligopolies, and on how the actors in those markets might have been able to improve their profits in the past and perhaps in the future.

When John Holland (1975) invented the GA, his original term for it was an "adaptive plan" which looked for "improvement" in complex systems or "structures which perform well." Despite that, most research effort, particularly outside economics, has been on its use as a

function optimizer. But, starting with Axelrod (1987), the GA has increasingly been used as an adaptive search procedure, and latterly as a model of human learning in repeated situations (Duffy, 2006). In the 1992 second edition of his 1975 monograph, Holland expressed the wish that the GA be seen more as a means of improvement and less for its use as an optimizer. The work we report on here is an example of the usefulness of the GA in a continuing research program about the behavior of sellers competing in an oligopoly, where the sellers are modeled as automata responding to the past actions of all sellers.

ACKNOWLEDGMENT

Research support was provided by the Australian Graduate School of Management and the Australian Research Council. Earlier versions of this chapter were presented at the 3rd International Conference on Computing in Economics and Finance, Stanford, June 1997; the Computational Modeling Workshop, UCLA Anderson School & Division of Social Science, Los Angeles, March 1999; The Genetic and Evolutionary Computation Conference, Orlando, July 1999; and The Industry Economics Conference, Sydney, July 2000. An earlier version also appeared as "Searching for Markov Perfect Equilibria in Oligopolistic Price Wars," INSEAD Working Paper 2000/65/MKT. Two anonymous reviewers gave valuable advice.

REFERENCES

Arifovic, J. (1994). Genetic algorithm learning and the cobweb model. *Journal of Economic Dynamics and Control, 18*(1), 3-18.

Axelrod, R. (1984). *The evolution of cooperation.* New York: Basic Books.

Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. In L. Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 32-41). London: Pittman.

Cooper, L.G., & Nakanishi, M. (1988). *Market share analysis: Evaluating competitive market effectiveness.* Boston: Kluwer.

Duffy, J. (2006). Agent-based models and human subject experiments, In L. Tesfatsion & K. L. Judd (Eds.), *Handbook of computational economics, volume 2, agent-based modeling* (pp. 950-1011). Amsterdam: Elsevier.

Fudenberg, D., & Levine, D. K. (1998). *The theory of learning in games.* Cambridge, MA: MIT Press.

Fudenberg, D., & Maskin, E. S. (1986). The Folk Theorem in repeated games with discounting or with incomplete information. *Econometrica, 54,* 533-556.

Fudenberg, D., & Tirole, J. (1992). *Game theory.* Cambridge, MA: MIT Press.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization & machine learning.* Reading, MA: Addison-Wesley.

Holland, J. H. (1975). *Adaptation in natural and artificial systems.* Ann Arbor: University of Michigan Press. (A second edition was published in 1992: Cambridge, MA: MIT Press.)

Holland, J. H., & Miller, J. H. (1991). Artificial adaptive agents in economic theory. *American Economic Review: Papers and Proceedings, 81*(2), 365-370.

Judd, K. L. (1998). *Mathematical methods in economics.* Cambridge, MA: MIT Press.

Marks, R. E. (1989, June 4-7). Breeding hybrid strategies: Optimal behavior for oligopolists. In J. David Schaffer (Ed.), *Proceedings of the*

3rd International Conference on Genetic Algorithms, George Mason University, USA (pp. 198-207). San Mateo, CA: Morgan Kaufmann.

Marks, R. E. (2002). Playing games with genetic algorithms. In J. Kacprzyk (Ed.), *Studies in fuzziness and soft computing* (Vol. 100, pp. 31-44). Heidelberg; New York: Physica-Verlag.

Midgley, D. F., Marks, R. E., & Cooper, L. G. (1997). Breeding competitive strategies. *Management Science,* (3), 257-275.

Mitchell, M. (1996). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.

Robson, A. J. (2005). Complex evolutionary systems and the Red Queen. *The Economic Journal, 115,* F211-F224.

Schraudolph, N. N., & Grefenstette, J. J. (1992). *A user's guide to GAucsd 1.4*. Technical Report CS92–249, CSE Department, University of California San Diego, USA. Retrieved from www.agsm.edu.au/1~bobm/teaching/SimSS/GAucsd.pdf

Slade, M. E. (1995). Empirical games: The oligopoly case. *Canadian Journal of Economics,* (2), 368-402.

Vriend, N. J. (2000). An illustration of the essential difference between individual and social learning, and its consequences for computational analyses. *Journal of Economic Dynamics and Control, 24,* 1-19.

Yao, X., & Darwen, P. J. (1994). An experimental study of *N*-person iterated prisoner's dilemma games. *Informatica,* 435-450.

ENDNOTES

1 We can make available the C sources for our programs and the 75 weeks of historical market data on request.

2 With up to four hereogeneous players, each facing a set of up to eight possible actions, the asymmetric (8×8×8×8) payoff matrix is much too large to reproduce here.

3 The Folk Theorem of repeated games (Fudenberg & Maskin, 1986) tells us that there is a multiplicity of N.E. of the repeated game; in essence, any individually rational outcome can be an N. E. with a sufficiently low discount rate.

4 Differentiated Bertrand oligopolistic competition closely resembles an asymmetric *n*-person prisoner's dilemma (Fudenberg & Tirole, 1992).

5 Were our players identical, we would have a symmetric game, and could follow the modeling simplification of Yao and Darwen (1994), as many computer scientists have done. But our players are not identical: their identity matters, as seen in Tables 1-4.

6 The GA parameters include: Crossover Rate = 13.0, Mutation Rate = 0.01; see Schraudolph and Grefenstette (1992).

7 Four actions require 2 bits per action; 4 actions, 4 players, and 1-week memory implies $4^4 = 256$ possible states; phantom memory is $4\times2 = 8$ bits. So $2\times256 + 8 = 520$ bits per string.

8 Eight actions require 3 bits per action; 8 actions, 3 players, and 1-week memory implies $8^3 = 512$ possible states; phantom memory is $3\times3 = 9$ bits. So $3\times512+9 = 1{,}545$ bits per string. Eight actions per player and 4 players (while retaining 1-week memory) require $3\times8^4+4\times3 = 12{,}300$ bits per string.

9 We should like to thank Bernhard Borges for this name.

Chapter LIII
Social Anti-Percolation and Negative Word of Mouth

Tom Erez
Institute for Scientific Interchange, Italy

Sarit Moldovan
Hebrew University of Jerusalem, Israel

Sorin Solomon
Hebrew University of Jerusalem, Israel
Institute for Scientific Interchange, Italy

ABSTRACT

Many new products fail, despite preliminary market surveys having determined considerable potential market share. This effect is too systematic to be attributed to bad luck. We suggest an explanation by presenting a new percolation theory model for product propagation, where agents interact over a social network. In our model, agents who do not adopt the product spread negative word of mouth to their neighbors, and so their neighborhood becomes less susceptible to the product. The result is a dramatic increase in the percolation threshold. When the effect of negative word of mouth is strong enough, it is shown to block any product from spreading to a significant fraction of the network. So, rather then being rejected by a large fraction of the agents, the product gets blocked by the rejection of a negligible fraction of the potential market. The rest of the potential buyers do not adopt the product because they are never exposed to it: the negative word of mouth spread by initial rejectors suffocates the diffusion by negatively affecting the immediate neighborhood of the propagation front.

INTRODUCTION

Many new products fail to meet their expected market share. While preliminary market surveys may report a large portion of potential buyers, the actual sales might reach only a negligible fraction of the market (Bobrow & Shafer, 1987; McMath & Forbes, 1998). Massive scientific research, as well as large financial, human, media, and technological resources,

have been invested to improve market sampling. However, there seems to be a "glass ceiling" to the success rate of sales prediction. In this chapter, we explain this phenomenon in terms of a "market percolation phase transition." We show that the eventual market share of a product depends crucially on the *nature* of interactions between potential buyers, more so than simply on their number (Bass, 1969) or even their network of connections (Solomon, Weisbuch, Arcangelis, Jan, & Stauûer, 2000; Solomon & Weisbuch, 1999; Weisbuch & Solomon, 2002).

Percolation Theory

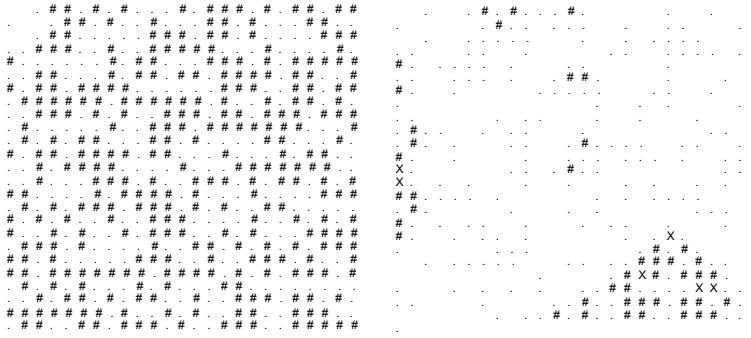
In general, percolation theory describes the emergence of connected clusters. Historically, percolation problems were first studied in chemistry, when Flory and Stockmayer studied Gelation as a percolation process on a Bethe lattice (Stockmayer, 1943). Since then, percolation theory has been developed extensively by both mathematicians and physicists (Stauûer, 1985), and was applied to a variety of other subjects, from epidemiology to oil fields and forest fires (Bunde & Havlin, 1999).

The main phenomenon studied in percolation models is the emergence of a phase transition: a dramatic change in the qualitative behavior of the system, triggered by an infinitesimal change in the parameters. For example, a small difference in virulence can make the difference between a seasonal flu and a global epidemic. Percolation theory offers a broad body of knowledge for the study of such phenomena, and by casting a problem in percolation terms one can gain access to a wide set of intuitions and rigorous results.

In short, percolation models involve agents that interact across a network. The interaction consists usually of influencing the state of a neighbor agent (i.e., a "sick" agent can potentially change the state of its "healthy" neighbors by infecting them). Obviously, the affected neighbor can now further affect one of its neighbors, and so allow the effect to diffuse across the network. Percolation theory studies the conditions in which the set of affected agents reaches a macroscopic size (a non-vanishing fraction of the entire set of susceptible agents). Interestingly enough, percolation theory predicts that often such a "global" diffusion will not take place, as the propagation may die out before any significant fraction of the system is reached by the diffusion dynamics. The transition to the percolating regime, where almost all susceptible agents are infected, is usually very sharp. The values of the parameters at which this happens are called "critical values." As one varies the parameters through their critical values, the system abruptly passes from the "seasonal flu" phase to the "epidemic" one. This is the famous percolation phase transition.

Mort (1991) suggested the application of percolation theory to marketing: a product spreads among adopters and can be said to percolate (or not) through the social network. Solomon and Weisbuch (1999) proposed looking at "Social Percolation"; they regard society as a network through which a social phenomenon (information/belief/product/behavior) may or may not percolate. In the right conditions a macroscopic cluster of adopters emerges, and most of the susceptible people will eventually be influenced. However, if the adoption rate ("social virulence") or the typical number of neighbors per agent are below their critical values ("percolation threshold"), the spread would stop before any significant fraction of the susceptible adopters is reached. As seen in Figure 1, if the proportion of susceptible agents is below the percolation threshold, they form disjoint little islands (Figure 1(a)). In this case a propagation that starts on one of the islands can

Figure 1. (a) A 25x25 regular grid in which a half of the cells were randomly marked with #. The resulting cluster structure is one of several disconnected islands. If a propagation starts from the upper left corner, it will not diffuse beyond the cluster marked in black. The propagation cannot reach other clusters, some of them marked in grey. (b) If an additional 10% is marked with X, the cluster structure changes dramatically, and a cluster proportional to the entire grid size emerges. Now a propagation from the upper right corner can span the entire network. (Neighborhood links are drawn along the main axes, so every agent has four neighbors at most: N, S, E, and W.)



(a) *(b)*

"conquer" this entire island, but cannot "spill" beyond its borders. However, as the proportion of susceptible agents increases, eventually bridges will form between a significant fraction of the islands. This is the percolation transition: the clusters expand to where they touch each other and form one giant percolating cluster (Figure 1(b)).

It is not simple to introduce percolation models to marketing science, mostly because percolation theory deals with "micro" data, on a fine-grained scale, and so one needs spatial micro-data in order to tune the model's parameters. For social networks, such information on the micro-structure is usually unknown; spatial micro-distribution of sales data is usually not publicly available. However, the approach has gained support in some recent studies—for example, Rosen (2000) reports that when Hotmail e-mail service was launched, it was quickly spread through word of mouth, and although e-mail is not a "local" service, it was still spreading in local areas. In addition, scientists working in percolation theory are usually not familiar with marketing science, its basic questions, or its methods. State-of-the-art simulations in percolation theory may look like video games to a marketing scientist, while for a physicist who is not familiar with the particularities of marketing literature, the quantitative message might look unrecoverably buried in particular details specific to the case under study.

In this chapter we extend the standard percolation model by introducing effects of negative word of mouth (NWOM). The resulting model was never considered before by percolation theory, probably because in the domains usually studied by it, there are no negative effects between neighboring agents. We do not claim that the simplified model we present can be directly applied to a marketing scenario—

this would require calibrating the model against data on the spatial and temporal distribution of sales (which the companies are very reluctant to release). However, since the main dynamical implications of the model are quite dramatic (large jumps in the sales for small parameter changes), they are likely to be replicated in a wide range of real systems. The present model should be considered as setting the scene for a tighter interdisciplinary effort by providing a formal model for a crucial market force: the negative word of mouth.

Negative Word of Mouth in Marketing

From marketing research, we learn that NWOM has a profound effect on adoption patterns. For example, Leonard-Barton (1985) found that an innovation reached only 70% of its market potential due to resisting consumers. In addition, Herr, Kardes, and Kim (1991) found that NWOM may decrease product evaluations. Two main characteristics of NWOM have been stated in marketing literature: it is more informative than positive word of mouth and thus may have a stronger effect (Hauser, Urban, & Weinberg, 1993; Herr et al., 1991), and it may be contagious and spread independently of exposure to the product (Marquis & Filiatrault, 2002). Both these attributes are addressed by our model.

Some models incorporating NWOM were suggested in marketing theory (e.g., Kalish & Lilien, 1986; Mahajan, Muller, & Kerin, 1984; Midgley, 1976; Moldovan & Goldenberg, 2004), but none of these models address the context of the underlying social network. The standard way to describe market share progression in marketing theory is by using differential equations, and most models used today are extensions of the classical Bass (1969) model. Predictions on adoption dynamics are made by fitting the parameters of these differential equations according to empirical sales data. The main criticism on the ODE approach is that it is inefficient at the early stages of product spread, as it is too sensitive to early fluctuations in the adoption process (Malcolm Wright & Lewis, 1997). This renders the forecast method less efficient, because by the time enough data points were collected, most of the "damage" (e.g., over- or under-production) is already done. We believe that this failure is inherently linked to the fact that the Bass framework preassumes "mean-field" conditions (i.e., all agents may interact with all others) and ignores the underlying social network.

Marketing theory distinguishes between two types of forces affecting the consumer—the "internal" (local) force, a name for all the influences cast on individual consumers by peer consumers (e.g., word of mouth), and the "external" (global) force, the kind of influences that are cast upon all consumers equally (e.g., marketing efforts). It is accepted in marketing that in the long run, internal forces are of greater significance for the adoption pattern of a product (although the external force is important for product awareness or for "activation" of the internal force). An estimation of the effect of the two forces, internal and external, shows that after takeoff, the word-of-mouth effect is 10 times greater than marketing efforts (Goldenberg, Libai, & Muller, 2001) and may be responsible for as much as 80% of the sales (Mahajan, Muller, & Srivastava, 1990).

Despite the fact that the internal force is more powerful than the external force, the standard effector employed for marketing is advertising, often by addressing an entire market. This introduces a bias into marketing research as well, which tends to focus on the external forces. The Social Percolation (SP) framework models internal forces as local interactions between neighboring agents on a

social network, and observes the properties of the resulting adoption patterns. Investigating the effects of internal forces in this manner can provide useful insights to market behavior, even if we ignore overall effects of external forces (the framework of social percolation can easily be extended to incorporate external forces as well (Proykova & Stauûer, 2002)). In this chapter we divide the internal force into positive and negative effects, and explore the effect of negative word of mouth in the SP framework. This provides a focus on the internal force in a structured context.

THE MODEL

Classical Social Percolation

We start by repeating the ideas of the SP framework (Solomon et al., 2000; Solomon & Weisbuch, 1999; Weisbuch & Solomon, 2002). The basic element of the model is an agent, representing a consumer. Each agent i is characterized by a "preference" value p_i, and agents are placed in a social network with a fixed structure. The model is made complete by introducing a product, represented by a global "product quality" value, denoted Q. The product spreads in the network between neighbors—whenever an agent adopts the product, it exposes its neighbors to the Q, and they decide individually whether to adopt it or not. Simulating this dynamics is done by exposing a "seed" group of agents to the product, and allowing it to spread from them until its diffusion naturally ends, either because it reaches the network's boundary or because it is blocked by non-adopters. Transmission of the product between neighbors happens according to the following adoption rule:

ADOPTION—Agent i will adopt the product if a neighboring agent adopted the product AND the product's quality is higher than the agent's preference: $Q>p_i$.

Thus, the potential market for a product consists of all the agents whose p is smaller than Q, but as long as this potential market forms only disconnected clusters, the product cannot reach all "islands," and its diffusion is limited.

Although the parameterization of the model with a fixed Q suggests a false interpretation that a unique well-defined value of quality may be assigned to a product, we do not claim this to be the case. Even as different consumers may have different perceptions of a product's quality, we choose to incorporate all such interpersonal diversity in the variability of the random values p_i. Since adoption is determined by comparing the "global" Q to the "private" p_i, introducing variance to Q over the agents is equivalent to introducing variance to the agents' p_i.

Incorporating Resistance into Social Percolation

Now we introduce our variant of the model, including NWOM. In what we described so far, the case $Q<p$ had no consequences: failing to meet a consumer's standards only meant that the product was ignored and the information about the product was not passed on to the consumer's neighbors. In the present model, we see this case as the equivalent of "disappointment," one of the roots of NWOM.

In order to introduce NWOM to the model, we propose to look at another property of the product, uncorrelated with its "value" (that is represented by Q). We wish to refer to the product's "susceptibility to NWOM," and parameterize it as a number between 0 and 1, following the (trivial) assumption that certain products are more sensitive to NWOM than

others. The value chosen may be related to the marketing strategy, the degree of novelty, and to many other aspects of the product domain. This parameter pertains to the response created in a potential customer after an encounter with the product that results in non-adoption: if agent i was exposed to the product, but has not adopted it ($Q<p_i$), we may say that the agent "resisted" the product. In that case, the agent may spread resistance to the product over her neighbors, generating NWOM. The effect of such negative spread is that the affected neighbors become less receptive towards the product—that is, their chances of adopting it decrease. We model this effect in the following way: in the case of $Q<p_i$, we denote $p_i - Q$ $p_i - Q$ as D_i (note that $D_i > 0$ always). We introduce a parameter a, denoting the extent one agent's resistance influences its neighbors. The spread of resistance (i.e., NWOM) is modeled by increasing p_j by aD_i for all agents j that are neighbors of i.

This increase of p_j will have no effect if j has already adopted the product. Yet, if $p_j < Q$ and $p_j + aD_i > Q$, then this agent is said to have been blocked by NWOM. The increase of p_j is additive—if several agents "project" resistance on one agent, all their individual NWOM effects add together on top of the original p_j. So, even if an agent was blocked, subsequent NWOM events will still make things worse for the product, because the NWOM this agent is prone to spread will be bigger (because p_j, and consequently D_j, will be bigger); Also, if $p_j + aD_i < Q$, this change has no immediate effect (as j may still adopt if one of her neighbors will expose her to the product), but this agent is now more prone to blocking by subsequent NWOM.

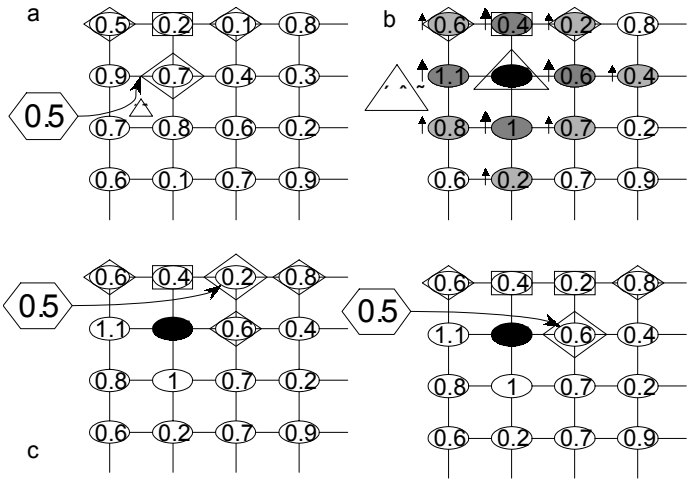
Since Q is limited to the range [0,1], an agent i with $p_i > 1$ will reject all products, the same as if p_j were 1. Yet, since p_j is allowed to grow freely beyond one, the resistance (D_i) that this agent may cast on its surrounding is essentially unbound. This allows the spread of NWOM to "block" completely the affected agents. In addition, we chose to model NWOM spread as occurring on a faster time scale than exposure to the product. This means that in case of resistance spread, the increase of all p_j happens instantly, before any further exposures of new agents to the product are considered. The rationale is simple: the typical time scale for casting NWOM is one conversation with one friend. Exposure to the product, on the other hand, is a slower process—the potential customer has to act (e.g., visit the point of sale) in order to potentially acquire the product (in a sense, one can say that "bad news travels faster").

In accordance with Marquis and Filiatrault (2002), we characterize another parameter b, which pertains to the product's susceptibility to "bad rumors"—NWOM that is not based on actual exposure to the product. Hence, the effect of resistance may travel to second-neighbors as well, and their p is increased by bD_i. At first glance, this modeling scheme seems primitive—why not extend this further and parameterize the effect to the n-th neighbor? Yet we wanted to avoid "giving wings" to such rumors—if NWOM is allowed to propagate freely on the social network, unrealistic dynamics occur as the NWOM behaves like another product. Therefore we restrict our attention to spread of NWOM up to second-order neighbors only. Although it may be argued that a and b are dynamic properties and bear inter-personal variance, in our simplified abstract model they are taken as a static scalar parameter, depending only on the product.

In summary, we introduce the following rule of interaction to the SP framework, in addition to ADOPTION:

RESISTANCE—If i was exposed to the product and $p_i > Q$, then for every agent j neighboring i, p_j is updated to $p_j + aD_i$

Figure 2. (a) Q, marked as a hexagon, is 0.5. The agent marked in square is the seed, with $p = 0.2$*, and so it adopts the product, and all its neighbors are added to the front (marked in diamonds). Next, the agent marked in a bold diamond is evaluated. Since its p =* 0.7 *is bigger than Q, this agent does not adopt, but rather spreads resistance (D =* 0.2*), marked in a triangle. (b) The agent is now marked in black, indicating that it will not adopt anymore. The resistance is immediately spread from the agent in a triangle to all its neighbors: for a =* 1 *and b =* 0.5*, the first neighbors' p increases by* 0.2 *(marked in dark grey), and the second neighbors' p increases by* 0.1*. Red arrows to the left of the agents mark the increase of p. Note how the agent to its left, whose p was* 0.4*, now has p =* 0.6*. This agent is no longer a potential buyer, but a potential spread of resistance; it was blocked by the NWOM. (c) Next, another agent is selected from the front. This agent is now marked in a bold diamond, and since its p < Q, it adopts the product and all its neighbors are added to the front. (d) Another agent is selected from the front. This time its p > Q, and this will cause spread of resistance. This agent was originally part of the potential market, but was blocked by NWOM and now will generate more NWOM. This illustrates the auto-catalytic nature of NWOM.*



AND

for every agent *k* that is a second neighbor of *i*, p_k is updated to $p_k bD_i$.

Thus, every case of adoption generates potential for further adoption, and every case of disappointment casts a "cloud" of NWOM around it. Figure 2 summarizes all rules of interaction.

SIMULATIONS

Method

The indexed set $\{p_i\}_{i=1}^{N}$ that specifies the "personal" values of p for every agent was randomly generated at the beginning of every simulation, with every such p_i chosen uniformly from the range [0,1]. At every iteration, values are fixed for *Q,a* and *b*, and the simulated dynamics start from a "seed" of one adopting

agent selected at random who spreads the product to her neighbors. These agents are added to the "adoption front," a list of agents who are waiting for evaluation of the product. At every time step, an agent i from the adoption front is randomly selected, and her current value of p_i (possibly increased by previous spreads of resistance) is compared to Q. If $Q > p_i$, the agent is marked as 'adopter', and all her neighbors are put on the adoption front, in line for exposure to the product. If $Q < p_i$, aD_i is spread to i's first neighbors and immediately added to their p_j, and bD_i is spread to i's second neighbors, immediately added to their p_k, the front expands away from the starting point (seed) in a random fashion, and the iteration ends when the front exhausts itself. This happens either when it traverses the entire lattice and reaches the boundaries, or when all the agents on the front rejected the product (so their neighbors were not added to the front). The first case corresponds to the product meeting its market potential, passing the percolation threshold, and traversing the network, and the second corresponds to the product being blocked by the population of agents.

Percolation Threshold Measurement

The product's "strength" may be measured in two ways: a percolation measure—whether the product percolated successfully through the lattice 'from side to side'; and market penetration—what is the size of the adoption cluster? Notice that these two measurements may not coincide, as a product may percolate and still achieve a small adoption rate. In regular lattices of three dimensions and up, this happens near the percolation threshold, where the percolating cluster has a fractal structure and a minimal density.

In the present work we focused on measuring the percolation threshold in order to determine whether a product 'percolated'; we checked how far it spread across the network, in terms of the distance from the initial seed. We looked at the "shell" structure of the network around the initial node, marking all the nodes found at distance d from the initial seed as belonging to shell d. Then we identified the largest shell, and percolation was marked successful if at least one agent from the largest shell adopted the product. This definition is equivalent with classical measures of percolation on regular lattices, and it generalizes it to cases of irregular networks as well.

In the current chapter we focus our attention on a regular lattice with non-periodic boundary conditions; we consider it an idealized "zero-order" approximation for a social network. Through this simplifying assumption we render our model less realistic, but more accessible to theoretical analysis. However, the model's definition is not dependent on the topology of the network, and the same model can be run on networks with arbitrary topologies.

We employed a binary search method of dichotomy for the estimation of the percolation threshold. For every instance of random values $\{p_i\}$, the threshold Q_c is a number such that for $Q > Q_c$ the product percolates, and for $Q < Q_c$ the product does not percolate. We estimated it by repeatedly exposing the lattice to different Qs, and changing Q in an adaptive way, according to the success or failure of the last iteration. If at iteration i the product of quality Q_i percolated, in the next iteration Q, it will be increased: $Q_{i+1} = Q_i + \frac{1}{2^{i+1}}$; accordingly, if at iteration i the product was blocked, Q_{i+1} will be decreased by the same amount.

Due to resistance spread, the values of $\{p_i\}$ change in the course of an iteration, and so we reset them to their original values between iterations. For every instance of random values $\{p_i\}$ and fixed a and b, we did 10 iterations of adaptive Q, and so effectively estimated the threshold within an uncertainty margin of size $\frac{1}{2^{10}}$.

RESULTS

In the present work we measured how negative word of mouth affects the percolation threshold. This was simulated by applying different values to *a* and *b*, and measuring the percolation threshold under these conditions. Figure 3 shows the threshold's dependence on *a* and *b* for lattices of various dimensions.

At the corner of *a* = b = 0, the measured threshold corresponds to the classical percolation threshold for these topologies (Stauffer, 1985). As *a* and *b* increase, the threshold increases, until it saturates near 1 for high enough values. The increase of both *a* and *b* causes an increase of the threshold; however, it is clear that the dominant effect is that of *b*, with an effective threshold of >0.9 for *b* as low as 0.55 in the 4D case (Figure 3).

Of course, the number of second neighbors increases with lattice dimension (8 in 2D, 18 in 3D, and 32 in 4D), but on a careful observation, a stronger, autocatalytic effect can be noticed, which is a result of the synergy between the neighbors. Since the front spreads from one point out, two nearby agents have big probability to be on the front at the same time. Since resistance spreads from agents who are on the front, their neighbors, hit by the local effect of NWOM, are probably on the front as well. These are exactly the agents that will be evaluated next, and therefore an increase to their *p* is the most significant, because they will certainly be exposed to the product and may spread

Figure 3. These plots show the percolation threshold (Z axis) as dependent on the parameters a *and* b *(the X-Y plane). In the corner of* $a = b = 0$*, the percolation threshold corresponds to the standard results, and it increases with both* a *and* b*. However, it is evident that b has a greater influence, and above a certain level, the network becomes totally impenetrable to the product. Figure 2a is for two dimensions:* $1000^2 = 1{,}000{,}000$ *agents; Figure 2b is for three dimensions:* $80^3 = 512{,}000$ *agents; Figure 2c is for four dimensions:* $40^4 = 2{,}560{,}000$ *agents.*

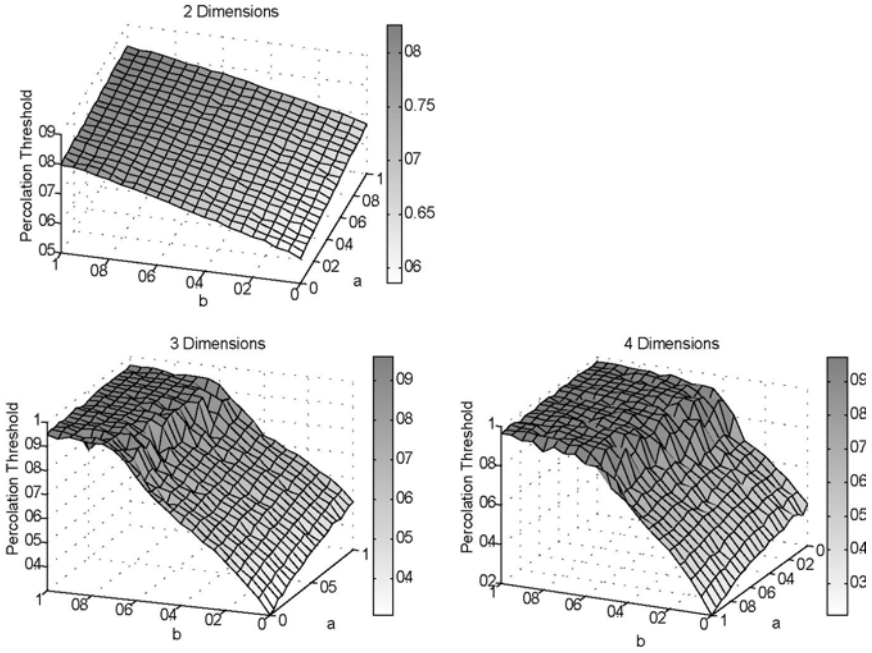
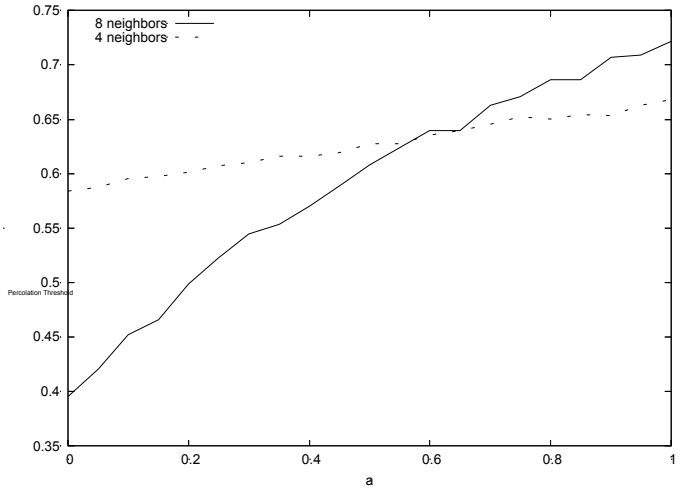


Figure 4. Here, the importance of the clustering coefficient is demonstrated by comparing the effect of a on two different two-dimensional topologies: 4 neighbors (dashed line) and 8 neighbors (full line). In the 4-neighbor topology, the front is not a disconnected set, since newly added agents would have links to other agents on the front already; but the front is more interconnected in networks of greater clustering coefficient. Thus, the network is more susceptible to effects of a, because more agents on the front would be affected by every event of resistance.



NWOM (see Figures 2(c) and 2(d)). In a sense, the NWOM hits the product at the most sensitive group of consumers—if the resistance would be spread to far-away agents, the front might only get there in a long time, or never get to that part of the network, which will render the blocking impact of the resistance less effective. Without any long-range effects, the local spread of resistance ensures maximum negative impact to the NWOM, while affecting only a finite number of agents (and so their proportion approaches 0% in the limit of large networks).

In the standard percolation model, the threshold decreases as the dimensionality of the lattice increases. This is due to the increase in the average number of neighbors, which brings about an abundance of paths between every two nodes. This number increases with the dimension, and therefore the product is less likely to be blocked, because there is greater probability of finding at least one path through which the product can percolate. At the extreme case of a highly connected irregular network, for example with a power-law distribution of the degrees, the percolation threshold was shown to be exactly 0 (Schwartz, Cohen, Ben-avraham, Barabasi, & Havlin, 2002). However, when NWOM is introduced, the increased connectivity of the network has the opposite effect—since resistance can spread to more neighbors, it is prone to have a greater impact on the front, and hence on the progress of the product. Since resistance is spread only from agents who are evaluated (i.e., on the front), the impact of NWOM on the front is greater when more agents on the front are close to each other. This number increases with the dimension: in 2D, every one of the 8 second neighbors has 4 of the others as her own second neighbor; in 4D, each one of the 32 second neighbors has 12 of the others as second neighbors.

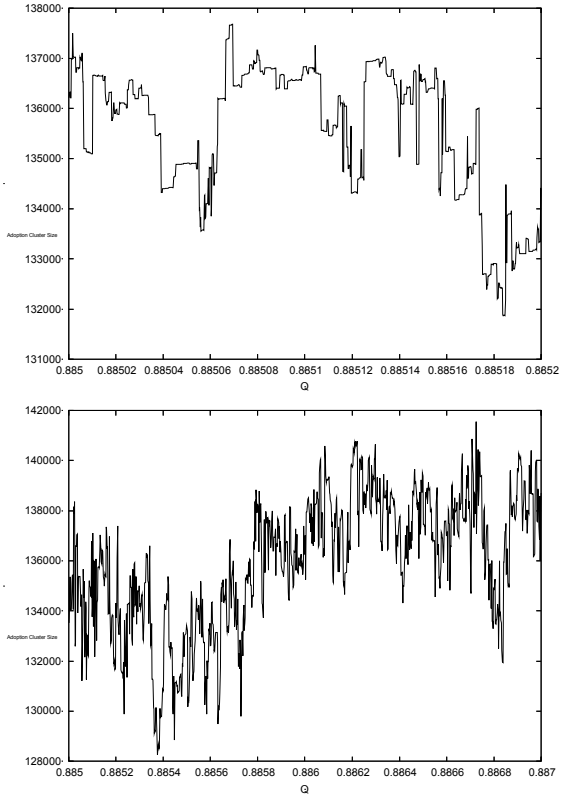
It is worth noting that all the above regular lattices are without triangles. Since only first neighbors are added to the front in case of adoption, every group of agents being added to the front has no links between its members. A more interconnected front is more susceptible to effects of *a*, because every event of resistance would spread NWOM to more agents on the front. In a network with a greater clustering coefficient, *a* is prone to have a larger effect. To demonstrate this, we compare a regular 2D lattice with a 4-neighbor topology (von-Neumann) and 8-neighbor topology (Moore). The results are presented in Figure 4, where it can be seen that the increase of the percolation threshold due to the increase of *a* in the 8-neighbor lattice is greater.

Another result is the emergence of local resistance leaders. In the classic percolation model, cluster size increases monotonically with *Q*. This is not the case when NWOM is incorporated into the model: for a given randomization, the cluster size may decrease with increasing *Q* (Figure 5).

This surprising result of the NWOM dynamics is due to particular micro-level setting: if in a particular section of the network, the first agents exposed to the product are of high *p*, they are bound to spread significant resistance and perhaps block the entire section. We call such agents "Resistance Leaders," as it is certain that they would be resistant to most products.

The blocking effect of such resistance leaders depends very much on the particular circumstances of the product spread—the question whether a resistance leader, or rather her neighbor, would be exposed first to the product might have a crucial effect on the progress of the product's spread in that section, and this depends totally on the micro-conditions there. If most of the agents in her neighborhood already adopted the product, the NWOM spread by a resistance leader may have little effect. This novel feature of the model happens both above and below the percolation threshold.

Figure 5. The dependence of the percolation on Q is non-monotonic, and a better product may have smaller sales; this is due to the effect of local resistance leaders, agents with high p that block their entire neighborhood if the product reaches them first.



DISCUSSION

As for the value of the current model per-se to marketing science, we can propose the following perspective: if a product is advertised, dif-

ferent parts of the lattice may be exposed to it, and the chances of a product being completely blocked are lower (although they exist; e.g., Goldenberg, Libai, Moldovan, & Muller, in preparation). The present model might be more directly applicable for products and services that are hardly advertised, but instead spread through word of mouth (e.g., restaurants, shops, high-risk and radical innovations).

The model we presented here introduces two novel parameters to the percolation model: the negative impact of non-adoption on the first neighbors (*a*) and on the second neighbors (*b*). These parameters are not a vague idealization; their real-world counterparts are easy to define: the parameter *a* is related to the nature of the disappointment caused by an event of non-adoption. Consumers can be disappointed and reject an innovation due to several reasons, such as a publicity campaign that creates high expectations that are not confirmed (Bearden & Oliver, 1984), fear of the new and unfamiliar, or resistance to change (Mukherjee & Hoyer, 2001; De-Jager, 2001), low expectations from a new technology, no benefits, or high pricing (Abrahamson, 1991; Bearden & Oliver, 1984). The main point is that disappointed consumers tend to spread more NWOM and have a higher effect on other consumers (Herr et al., 1991; Richins, 1983).

Moreover, consumers may spread NWOM to their friends just on the basis of their exposure to negative information, and without any trial or contact with the product. The parameter *b* represents the extent to which consumers tend to tell each other stories about products they never tried. The above reasons for rejecting the innovation can serve as reasons to spread NWOM further on, a phenomena that was found in previous studies (Marquis & Filiatrault, 2002). Leonard-Barton (1985) found that 20% of dentists were familiar with, yet rejected, a *successful* dental innovation; many of them were not even willing to try it as a result of NWOM. Consistent with the theme of "bad news travels faster," consider the following anecdotal report of a major taxi company in New Zealand which lost almost 60% of its business as a result of an angry customer spreading her story to thousands of women throughout New Zealand (Cecil, 2001).

These cases can be documented, but can hardly be understood or predicted in the classical (Bass-like) framework. Standard techniques of estimation of the potential market give little or no attention to the emergent effects of consumer interaction. Focus groups and random sampling may give an accurate estimate of the potential in a naïve market, but as soon as the product is actually introduced, the naïve market changes shape by effects of word of mouth whose source are the early adopters. A product may seem to be good enough at the preliminary probing of the market, and yet fail due to the devastating effect of NWOM.

Therefore, it would be very helpful if marketing research could estimate the parameters of the endogenous social interaction of the market. A more complexity-aware probing of the market should also estimate the "interaction value" of a disappointed customer, paying attention to the probability of people discussing the product without ever being exposed to it. Our model shows that these aspects of the product and the market context have great impact on the eventual success or failure of the product.

Since this novel approach challenges the traditional methods for sales forecast, we do not expect it to pass without resistance. Nonetheless, the scientific method revolves around falsifiability: since there is a radical difference between the theoretical underpinning of social percolation and aggregate, Bass-like models, one should seek experimental settings where distinct predictions can be offered by the rival approaches. Such tests are the only reliable way to discern between the two theoretical

approaches and transcend the bias inherited by disciplinary tradition. The main prediction of the present anti-percolation framework is that in the case of a failed product, many of the potential adopters will never feel the wave of propagation and are not at all acquainted with the product. This is at variance with the predictions of aggregate models, where the entire population is either adopting or explicitly rejecting the product. Thus, it could be quite straightforward to discriminate empirically between the various model predictions.

In summary, we believe that percolation theory (with the present anti-percolation amendments) has much to contribute to marketing research. We hope that the marketing world will recognize the potential in this mature and rich theory, embrace the language of agent-based models and simulations, and focus its attention on designing and testing the predictions of increasingly realistic models.

ACKNOWLEDGMENTS

The authors thank Jacob Goldenberg and Dietrich Stauffer for their help. This research was supported in part by a grant from the Israeli Academy of Sciences and by the Yeshaya Horowitz Association through the Center for Complexity Science.

REFERENCES

Abrahamson, E. (1991). Managerial fads and fashions: The diffusion and rejection of innovations. *Academy of Management Review, 16*(3), 586-612.

Bass, F. M. (1969). A new product growth model for consumer durables. *Management Science, 15,* 215.

Bearden, W. O., & Oliver, R. L. (1984). The role of public and private complaining in satisfaction with problem resolution. *Journal of Consumer Affairs, 19*(Winter), 222-240.

Bobrow, E. E., & Shafer, D. W. (1987). *Pioneering new products: A market survival guide.* New York: Dow Jones-Irwin.

Bunde, A., & Havlin, S. (Eds.). (1999). Percolation and disordered systems: Theory and applications. *Physica A, 266.*

Cecil, J. (2001). The vengeance factor. *Rough Notes Magazine, 144*(1), 44-45.

De-Jager, P. (2001). Resistance to change: A new view of an old problem. *The Futurist, 35*(3), 24-27.

Goldenberg, J., Libai, B., Moldovan, S., & Muller, E. (In preparation). *The NPV of bad news*.

Goldenberg, J., Libai, B., & Muller, E. (2001). Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters, 12*(3), 209-210.

Hauser, J. R., Urban, G. L., & Weinberg, B. D. (1993). How consumers allocate their time when searching for information. *Journal of Marketing Research, 30,* 452-466.

Herr, P. M., Kardes, F. R., & Kim, J. (1991). Effects of word-of-mouth and product-attribute information on persuasion: An accessibility-diagnosticity perspective. *Journal of Consumer Research, 17,* 454-462.

Kalish, S., & Lilien, G. L. (1986). A market entry timing model for new technologies. *Management Science, 32*(2), 194-206.

Leonard-Barton, D. (1985). Experts as negative opinion leaders in the diffusion of a technological innovation. *Journal of Consumer Research, 11*(4), 914-926.

Libai, B., Muller, E., & Peres, R. (2005). The role of seeding in multi-market entry. *International Journal of Research in Marketing, 22*(4), 375-393.

Mahajan, V., Muller, E., & Kerin, R. A. (1984). Introduction strategy for new products with positive and negative word of mouth. *Management Science, 30,* 1389-1404.

Mahajan, V., Muller, E., & Srivastava, R. K. (1990). Determination of adopter categories by using innovation diffusion models. *Journal of Marketing Research, 27,* 37-50.

Malcolm Wright, C. U., & Lewis, T. (1997). A validation of the bass new product diffusion model in New Zealand. *Marketing Bulletin, 8,* 15-29.

Marquis, M., & Filiatrault, P. (2002). Understanding complaining responses through consumers' self-consciousness disposition. *Psychology & Marketing, 19*(3), 267-292.

McMath, R. M., & Forbes, T. (1998). *What were they thinking?* New York: Times Business-Random House.

Midgley, D. F. (1976). A simple mathematical theory of innovative behavior. *Journal of Consumer Research, 3*(1), 31-41.

Moldovan, S., & Goldenberg, J. (2004). Cellular automata modeling of resistance to innovations: Effects and solutions. *Technological Forecasting and Social Change, 71*(5), 425-442.

Mort, J. (1991). Perspective: The applicability of percolation theory to innovation. *Journal of Product Innovation Management, 8,* 32-38.

Mukherjee, A., & Hoyer, W. D. (2001). The effect of novel attributes on product evaluation. *Journal of Consumer Research, 28,* 462-472.

Proykova, A., & Stauffer, D. (2002). Social percolation and the influence of mass media. *Physica A, 312,* 300-304.

Richins, M. L. (1983). Negative word-of-mouth by dissatisfied consumers: A pilot study. *Journal of Marketing, 47,* 68-78.

Rosen, E. (Ed.). (2000). *The anatomy of buzz: How to create word of mouth marketing*. New York: Doubleday-Currency.

Schwartz, N., Cohen, R., Ben-avraham, D., Barabasi, A.-L., & Havlin, S. (2002). Percolation in directed scale-free networks. *Physical Review E, 66*(015104), 1-4.

Solomon, S., & Weisbuch, G. (1999). *Social percolation.* Retrieved from http://adap-org/9909001/

Solomon, S., Weisbuch, G., Arcangelis, L., Jan, N., & Stauffer, D. (2000). Social percolation models. *Physica A, 277,* 239-247.

Stauffer, D. (1985). *Introduction to percolation theory* (1st ed.). London: Taylor and Francis.

Stockmayer, W. (1943). Theory of molecular size distribution and gel formation in branched polymers. *Journal of Chemical Physics, 11,* 45-55.

Weisbuch, G., & Solomon, S. (2002). Social percolators and self-organized criticality. In *Handbook of graphs and networks: From the genome to the Internet* (pp. 342-353). Weinheim: Wiley-VCH.

Chapter LIV
Complexity-Based Modelling Approaches for Commercial Applications

David Collings
BT PLC, UK

Nicola Baxter
BT PLC, UK

ABSTRACT

Understanding complex socio-economic systems is a key problem for commercial organizations. In this chapter we discuss the use of agent-based modelling to produce decision support tools to enhance this understanding. We consider the important aspects of the model creation process which include the facilitation of dialogue necessary to extract knowledge, the building of understanding, and the identification of model limitations. It is these aspects that are crucial in the establishment of trust in a model. We use the example of modelling opinion diffusion within a customer population and its effect on product adoption to illustrate how the agent-based modelling technique can be an ideal tool to create models of complex socioeconomic systems. We consider the advantages compared to alternative, more conventional approaches available to analysts and management decision makers.

WHAT IS THE PURPOSE OF A MODEL?

"The best material model of a cat is another, or preferably the same, cat" (Norbert Wiener, Philosophy of Science, 1945).

Models in business are tools used to gain insight and understanding of a real system relevant to the business, and aid appropriate decision making.

Most often, businesses must understand complex socio-economic systems such as the economic environment in which they operate, the social and organisational systems that make up their operations, and the customers they serve. It is unusual for any model to be totally predictive, and obviously the definition of complete prediction is itself subjective. Socio-economic systems present additional problems. There are often problems in defining a model;

also measuring quantities in the real system, which correspond to input and output parameters in the model, can be very difficult. Additionally these systems are often non-linear, dynamic, and subject to stochastic influences.

An element of prediction is clearly important, however what is paramount is that the model should give useful understanding.

WHAT IT MEANS TO BE USEFUL?

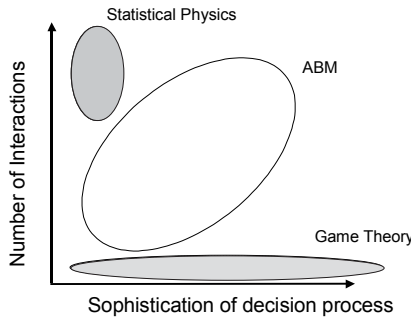
All modelling techniques necessarily involve some degree of abstraction. There are simplifications of the processes described by the model, as well as simplifications due to the necessary drawing of boundaries in the modelling space. There may be limitations in the language used to describe the processes in the model, or restrictions on thinking due to the blinkered adherence to specific intellectual norms. There may be limits on the knowledge of the problem due to finite human cognitive ability or lack of accurate data on the system under study.

These sorts of limitations are common to all forms of representation of ideas or concepts, whether the ideas are in art, literature, or science. To be useful, these representations must effectively persuade and communicate the ideas to others. In modelling there can be an additional dimension to the representation process. The process of the creation of the representation can and should involve the user of the model. An effective model is a tool that facilitates dialogue and the generation of understanding in the minds of the users of the model. The construction of the model should lay down the knowledge of the system held by the user, and act as a tool to explore the extent and precision of that understanding. It is a process of converting tacit internal mental models to overt models that can be challenged and discussed. The modelling process should also reveal the assumptions that are made and any of the limitations outlined above.

THE USE OF COMPLEXITY APPROACHES IN MODELLING

Socio-economic systems are inherently complex. They are characterised by interactions. The form and outcome of these interactions can vary in quantity and sophistication. There are a number of different analytical techniques that could be applied to understand these sorts of systems. Where there are small numbers of participants, game theory can provide useful insights. This technique can represent decision processes that range in sophistication from the simple to the arbitrarily complex. At the other extreme, statistical physics approaches can provide understanding where there are large numbers of interactions, but with low levels of sophistication in the description of the decision process. These two extremes are limited in their application; they can be considered non-complex in the sense that they represent equilibrium solutions and there is little modification of behaviours by the participants. Most socio-economic systems display more complex

Figure 1. Illustration of the types of problems that can be addressed by different analytical techniques



behaviour with large numbers of interactions. Agent-based modelling (ABM) sits between the two extremes of statistical physics and game theory, and is ideal for representing the natural complexity of such systems. Bonabeau (2002) and Tesfatsion (1997) give interesting examples of the approach in business and economic applications.

Socio-economic systems such as firms within economies, consumers within a population, or employees within an organisation can all be thought of as a collection of discrete entities with separate rules of behaviour, and which interact with each other and the environment that contains them. The behaviours of the individuals can be modified as a result of the interactions. These complex adaptive systems are difficult to model and understand. They are often non-linear and dynamic, can involve or are subject to stochastic influences, and can display self-organisation or emergent behaviour. ABM is a technique that can produce a 1:1 mapping of the complex adaptive systems in the system under study and its representation as a model. The ABM technique creates a population of entities whose rules of behaviour and rules of interaction mimic those of the system being modelled. The models that result retain the dynamics of the real system. The ABM technique allows the creation of a model that goes beyond simple cause-and-effect relationships that human intuition is usually limited to, and can include complex, higher-order, path-dependant effects. The ABM technique permits spatial and network effects to be naturally captured. Information flows within social networks can be represented as a critical factor in socio-economic systems. Network effects such as network externalities can be explicitly described. With the expression of individual behaviours, models that go beyond the rational economic agent become possible.

The ABM technique retains the details of the interactions that occur, which not only gives a more accurate rendering of the system under study, but produces a framework for understanding the processes involved in a very direct and intuitive way. The accessibility of the concepts the model is built upon allows the limitations and boundaries to be understood. The rules of behaviour and structure of the interactions can be constructed using the knowledge and experience of the model user. These are all critical factors in making a model useful, effective, and trusted by the user.

THE USE OF AGENT-BASED MODELLING IN UNDERSTANDING CUSTOMER BEHAVIOUR

An ideal example of the use of complexity approaches in modelling a socio-economic system relevant to business is modelling the diffusion process within a customer population which can be applied to product adoption. We compare the standard, alternative approach, highlighting the advantages of the complexity-based approach and how it can be used to create a more effective model.

THE CONVENTIONAL NON-COMPLEX WAY

According to Mahajan and Peterson (1985),

the diffusion of an innovation (knowledge or actual take up of a new product or service) is the process by which that innovation is communicated through channels over time among the members of a social system.

Therefore, in order to understand the diffusion process, and how to influence it, it is key to understand the methods and nature of commu-

nications between individuals within a population. This can be combined with the processes those individuals use to assess a product to determine if it satisfies their needs sufficiently for adoption to occur.

There have been a number of different attempts to produce models for innovation diffusion. One of the earliest is that of Fourt and Woodlock (1960). Other well-known diffusion models, such as the Bass (1969) model, describe the diffusion process by an equation such as equation 1.

$$\frac{dN(t)}{dt} = g(t,N)[N_T - N(t)] \quad (1)$$

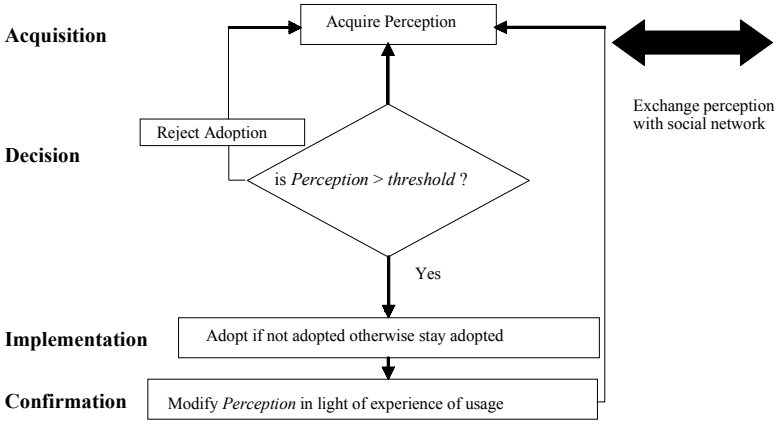
where $N(t)$, $g(t, N)$, N_T, are the number of adopters, the coefficient of diffusion at time t, and total number of potential adopters in the social system. This shows that the rate of change in adopters with respect to time is proportional to the number of people who have not yet adopted. It is simply the retrospective application of a suitable mathematical form to empirical data.

The coefficient g describes the characteristics of the diffusion process and is determined by the nature of the innovation, the communications channels used, and the social network. Variation of this parameter allows tailoring of the model. g can be a constant, a function of N, or a combination of the two parameters. With the coefficient as a constant, it describes the external influences (i.e., outside of the social system) such as the effect of mass media. With the coefficient g, a function of the number of adopters, the diffusion is influenced by factors internal to the social system—that is, imitation by consumers is represented. This basic form is observed in quantitative studies in real life. It can be considered as an observed macroscopic property of the diffusion system. These diffusion models also use a number of assumptions limiting the realism of the resulting models. An attempt has been made to address the problem of the fundamental lack of flexibility of the diffusion models. However these alterations are little more than mathematical tinkering to give functions with the assumed "correct" properties. They remain phenomenological models, and the mathematics is not based on the fundamental behaviour of the system being described. It is exactly this lack of transparency in the assumptions that reduces trust and limits understanding of how meaningful modifications of the model can be made.

In order to produce a more useful and realistic model of the diffusion process, the ABM approach can be adopted. This technique involves creating a population of discrete entities, or "agents," each representing an individual member of the real population of consumers in question. Each of the agents contains a set of goals, beliefs, and actions, and can interact with other agents or the environment in which the population exists. Agent-based modelling enables the problem to be addressed using a bottom-up approach. The goals, beliefs, actions, and interactions are microscopic attributes of the system. The overall macroscopic behaviour appears as a result of the combined effect of all the microscopic attributes and the complex interactions between them. There are no assumptions as to the macroscopic properties; instead any higher-order effects come from the description of the behaviour of the individual agents and their interactions.

The model allows the explicit incorporation of easily articulated ideas from sociology and psychology applied to product adoption. These concepts can be discussed and the details can be refined with the model users. Not only does the accessibility of the concepts enhance trust in the model, but the parameters or levers that the user can control in the real system understudy are revealed, adding to the acceptability and utility of the model.

Figure 2. Flow chart of implementation of multi-stage adoption process: The stages are represented in capitals and the variables are italicised.



We now consider the key elements of an ABM describing a customer adoption model.

THE KEY COMPONENTS OF THE MODEL

The key elements of the model are the process by which a consumer decides to adopt a product, the cognitive process, and the network of connections that exist within the population through which information about the product or service is passed—that is, the social network.

THE COGNITIVE PROCESS

There are a number of theories from the social sciences that shed light on the cognitive process involved in product adoption. These include theories on human need, social comparison theories, conditioning theories, decision and choice theories, amongst others (Jager, Van Asselt, Rotmans, & Vlek, 1997). In most cases, the decision theory described by Rogers (1995) gives clarity of understanding and ease of application. This is the framework we adopt for the learning process. We also discuss fads and other network externalities as separate cognitive processes. The separation of these processes is useful for facilitating the understanding of the processes that exist within the system under study and their relative importance.

Learning

Roger's theory was developed independently of the needs of a computer-based simulation, however its structure lends itself well to implementation in an ABM. The model describes a multi-stage process which can be represented by the flow chart in Figure 2. In the simulation, each individual follows its own instance of this process. It has its own, in general, unique values for the parameters within the flow chart

The stages have been interpreted as follows:

- **Acquisition:** During the first stage of the decision process, an individual receives information that alters his or her percep-

tion of the product or service on offer. This change of perception is due to a combination of inter-agent communication and external factors such as marketing and competition.

- **Decision:** The individual makes the choice of adopting or not depending on whether they have a sufficiently high perception.
- **Implementation:** This stage represents the explicit act of adoption or rejection.
- **Confirmation:** In this step the individual's parameters are adjusted according to the experience the individual has using the product.

THE NETWORK CONNECTIONS

Fads and Other Network Externalities

In its purest form, this is a process where information on the products' functionality has no influence on an adoption decision. It is who within the population that has adopted that generates the pressure to adopt. The pressure can be positive or negative, following or avoiding trends, and the influencing individuals can be highly specific or general. Examples of these processes include threats to lost legitimacy, such as needing to demonstrate conformity or non-conformity and competitive pressures, where individuals must adopt or potentially get left behind. In some ways this can be considered as a form of increasing returns (Arthur, 1994), since the desirability of a product increases as the number of adopters increases; however, the functionality or utility of the innovation could well be independent of the base of adopters.

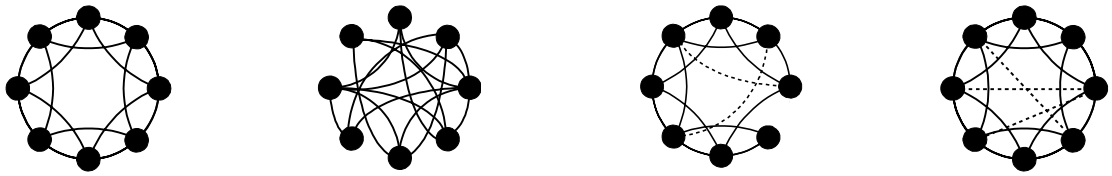
The fad process can be driven by purely economic pressures and not social. An example of this is the charging regimes for some telephone companies. These companies offer a preferential tariff for calls made within the phone company's own network. There is increased pressure to adopt a particular phone company's service, as the number of members of your regular calling circle that have also adopted that company's service increases.

In an ABM, network externalities—both social and economic—can be implemented as follows. At each arbitrary time step, the individual goes through its social network (members of the population an individual has contact with) and looks to see how many of these people have adopted. If the proportion of the acquaintances that have adopted exceed a threshold value that is a characteristic of that individual, then that individual itself adopts. This is the process used by Watts (2001).

The Social Networks

The form of the social network is crucial in driving the adoption process. An accurate description is therefore very important. Survey data would be ideal to provide the structure of the network, however it may be necessary to use theoretical constructs since the survey data, if not incomplete, could be imprecise to a certain degree. Recent theoretical work on networks (Newman, 1999) has produced many interesting results relating to the nature of the structures that exist in social systems and their theoretical properties. These results can be used to construct appropriate theoretical networks, and the knowledge of their properties act as a guide to the important characteristics, such as when cascading adoptions occur and the extent of the cascades. Also work on the robustness of communications networks can be used to gauge the reliability of a social network with regard to its ability to maintain the flow of information. These results allow identification of individuals who have a disproportionate influence and whose behaviour is crucial for the successful dispersion of information.

Figure 3. (a) Regular ring lattice with N=8, z=4 (each vertex is connected to its z nearest neighbours); (b) Random lattice with ½ Nz edges; (c) Watts—Strogatz small world (the dotted lines show the links that have been rewired); (d) Newman-Watts small world with three extra links (dotted lines)



(a) *(b)* *(c)* *(d)*

The Arrangement of Networks

The social network refers to the linkages that exist between individuals, along which information passes. Networks have been modelled, abstractly, using graphs—a collection of nodes with links connecting them. These have, in the past, been either completely ordered or completely random. The ordered graph has nodes with a uniform number of links between neighbours, and the random arrangement has links between nodes distributed within the entire population at random. Most networks that exist in the real world appear to be somewhere between these two extremes. In social networks, this is intuitively correct. If two individuals have a mutual friend or acquaintance, then those two individuals are likely to know each other as well. The linkages are not random, and you expect to observe distinct groupings or clusters. Social networks are not entirely clustered though, and linkages spanning the population can also exist. We are all familiar with the surprise people express when they meet a complete stranger, only to discover they have an acquaintance in common—it's a small world. From empirical work by Milgram (1967), the mean acquaintance distance between any two people in the United States was determined to be around six. Hence it appears that the way social networks are linked means we should not be that surprised by common acquaintances and that significant spanning linkages within the population must therefore exist. Recent work by Watts and Strogatz (1998) has taken ordered networks and introduced randomness by rewiring a number of links. These new structures are known as "small world" networks with reference to the phenomenon described above (see Figure 3). Two important parameters associated with these networks are the length of the shortest path connecting two individuals, the characteristic path length, and the average probability that two nodes with a mutual acquaintance will be connected, the clustering coefficient. Starting with an ordered lattice and introducing random short cuts, the path length falls whilst the clustering coefficient remains high. This results in clustering and short global separation between nodes—that is, a small-world character with local groupings.

It appears that the small-world model is an appropriate abstract network to use in modelling a social network. It allows us to characterise real data from surveys, where the length scale and cluster coefficient is known, or permits us to construct linkages in a realistic way when exact data for a population is absent.

For a simulated set of links, we still have to have a realistic idea as to what kind of distribu-

tion of connectivities we expect for a network of consumers. Amaral, Scala, Barthelemy, and Stanley (2000) have analysed a number of real networked systems, ranging from technological systems such as a power grid system to social network systems. The social network systems give the most appropriate surrogate for the distribution of links along which information about a product is passed in a population of consumers. An interesting observation in the paper is that the distributions for the different systems can be characterised as scale free, broad scaled, or single scaled. These respectively have connectivity distributions characterised by a tail that decays with a power law, by a power law followed by an exponential tail, and by a fast decaying distribution such as an exponential or a Gaussian. The mechanism for the creation of links determines the form of the distribution. Essentially, scale-free distributions occur when there are preferential connections occurring with popular nodes. The broad scale effects occur when there are constrains such as rising cost of connecting to popular nodes or aging of the node where the node can no longer add connections.

Amaral et al. (2000) considered empirical data for a network of movie actors, the network relationship being derived from their collaborations in particular films. They found that the distribution is power law for a collaboration number between 30 and 300, and truncated for higher values, indicating broad-scaled overall. They also considered the distribution for acquaintances for 43 Utah Mormons and 417 high school students indicating best friends, both first-two friends and first-three friends. In friends and acquaintances, the distribution is found to be Gaussian. The interpretation that the movie actor distribution is broad scaled is that there are a number of popular stars who work with a large number of other actors, but there are constraints on the preferential addition of links as actors retire and no longer can add links. It is worth bearing in mind whether any of these sorts of processes may occur in the population that is being simulated. It would appear that most friendship and acquaintance distributions in normal populations are distributed in a Gaussian fashion.

There has been considerable interest recently in the theoretical analysis of the simulated social networks, (Newman et al., 2000a, 2000b), considering random networks (Newman & Watts, 1999; Moore & Newman, 2000) and considering small-world networks. The generation function formalism used in most of these analyses is a powerful technique and can give useful statistical properties of the system. Using a fad type of adoption mechanism, the percolation transition can be determined—that is, the critical point when an innovation is adopted by the majority of the population. The speed and extent of the cascade can be determined as a function of the distribution of links, and the magnitude and distribution of thresholds. These parameters are of considerable interest in the development of marketing strategy. They enable a population of consumers to be analysed and the nature of the innovation adoption due to fads to be anticipated. This type of analysis could also give pointers as to how marketing strategy should be used to modify the distributions of links and the thresholds. These theoretical results augment the result that can be obtained from computer-based simulations of consumers (i.e., agent-based consumer models).

There has also been work considering the robustness of networks, principally technological networks, such as telecommunications and power grids. These studies such as Callaway et al. (2000) have considered the form of the network structure and the kinds of node distribution that lead to structures that are robust to deletion of individuals, either through random or targeted processes. These results could be extended to social networks. In a consumer

model, the adoption process is driven by the information passed through the linkages. A detailed knowledge of the kinds of structures vulnerable to information flow being disrupted would be of considerable interest to marketing strategists. There is also the potential to use marketing tools to modify the structure of the network to ensure that the structure is more efficient and robust, and hence achieves a result closer to expectation.

THE ICRM TOOL: A SPECIFIC APPLICATION OF THE IDEAS FROM THE ABM OF DIFFUSION AND ADOPTION

We now consider a specific example that demonstrates the use of the ABM technique. It shows the development of the model's assumptions with the user, and how the calibration process is used to further facilitate acceptance of the model and help the user understand his or her business problem and its solution. The agents use the cognitive process and word of mouth as a form of network externality. They are linked together using a small-world-type network.

THE PROBLEM

Analysts predict that companies will spend billions of dollars in the next few years on software and services to help them manage their customer interactions more effectively. Unfortunately, most organisations do not fully understand how these investments will affect their customer base, and currently only 21% of Customer Relationship Management (CRM) projects meet all expectations (Hewson Consultancy Group, 2000).

Word of mouth can be a powerful way for businesses to recruit new customers. People are often suspicious of advertisements and seek opinions from trusted friends and acquaintances before purchasing a product or service. Referrals are effective, as they usually come from someone who is familiar with the product or service but has no financial motive for recommendation. Companies generally find that referred customers require less sales time to build trust and credibility, and tend to be more loyal than those whose purchases are driven by advertisement (Griffin, 1995). Some businesses have successfully manipulated word of mouth by running referral schemes offering benefits to existing customers who recommend friends. However, just as positive word of mouth can be a highly effective marketing tool, negative word of mouth can be destructive. Typically, a dissatisfied customer will tell eight to ten people about their experience; one in five will share their dissatisfaction with twice as many (Griffin, 1995). This statistic is of particular interest in the CRM arena where technology-driven solutions are often sold on their ability to cut operating costs regardless of their impact on customer satisfaction.

Within BT, clients understood these issues; however, tools to understand the implications of improved or degraded customer opinion as a result of changes to CRM strategy, and hence the return on investment for a particular CRM investment strategy, were not available.

The intelligent customer relationship management (iCRM) tool was therefore created. This is an agent-based customer model built which uses the cognitive process discussed and a Newman-Watts small-world social network discussed in the "The Key Components of the Model" section. It is a decision support tool enabling clients to visualise the impact of their CRM strategies and explore the effects of word of mouth on customer recruitment and retention. The problem lends itself well to analysis using the ABM technique, focussing in on the influence of individuals and the interactions

between them. It can be used to recognize the future consequences of CRM implementations and estimate the nature of return on investment.

The iCRM tool captures the key drivers behind customer behaviour and choice, and can be used in a consultancy environment to facilitate dialogue. Trials have shown that using the tool can stimulate discussion and improve understanding of CRM issues. The agent-based approach offers a more realistic representation of a customer population, but it still requires assumptions and simplifications. In reality, many of the parameters needed to define a theoretical model are difficult to measure, and specific external events can cause unexpected behaviour. Results should be treated as illustrative, but can still be used to compare the impact of different CRM strategies both in terms of market share and financial performance. It is exactly this sort of problem that requires a modelling technique that permits the examination of the levers on the system but also reveals the assumptions. The ABM technique is superior to the other techniques available in these respects.

MODEL STRUCTURE

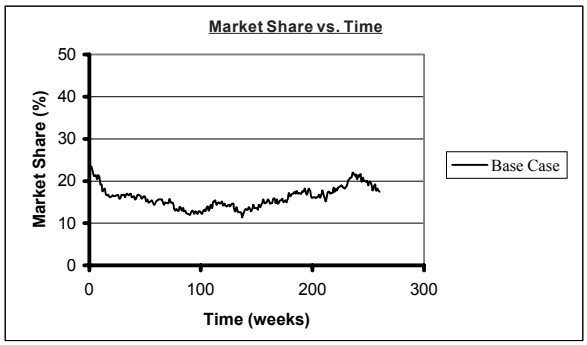
The iCRM model consists of a population of 500 customers and a single product. For simplicity, the product is restricted to two parameters: price and quality. The agents within the model are heterogeneous, and each has its own interpretation of the product's attributes, forming a distribution of perceptions within the population. When an individual has a combined perception that exceeds his or her internal threshold, he or she will adopt (or readopt) the product on offer. The underlying adoption model makes this tool particularly suitable for subscription products or services where people review their decision to purchase at regular intervals. Each agent in the model is part of a social network through which perceptions are compared and influence exerted on members of the population.

Research East (2000) has shown that long-term customers are less likely to recommend a product or service than recent adopters. New customers are more conscious of their purchase—they want to talk about it. Over time, habit and familiarity take over and recommendation rates fall, an effect captured by the iCRM model by linking the probability of social interactions to length of adoption. In addition to word of mouth, potential customers are influenced directly by external factors such as marketing, competition, and CRM interactions. At each arbitrary time step, a proportion of agents are affected by marketing and sales material; for simplicity this is always assumed to have a positive impact on perception—the number of people targeted and the magnitude of the influence can be adjusted as appropriate. Although the model only deals with a single product, the effects of competition are an important aspect of the adoption process. This pull towards alternative products or services is captured in the model as a gradual erosion of perception at each time step and can be tuned to represent markets with differing competition levels. The most important influence in the tool is the effect of CRM experiences on consumer perception. This represents personal interactions between the company and customer such as complaints, repairs, billing, and so on. These contacts could be made through a range of channels and can be defined in terms of their frequency and impact.

USING THE ICRM TOOL WITH CLIENTS

In order to illustrate the impact of a particular CRM investment, the model must first be calibrated to represent the current strategy in

Figure 4. Graph showing a fictitious base case for a business with an initial market share of approximately 25%



question (referred to as the base case scenario). This is usually an iterative process that involves working with a client to populate the model's parameters with a combination of factual data and estimates based on qualitative discussions. Once the client is satisfied that the model is giving a reasonable prediction of future market share (Figure 4), he or she can consider CRM investments that change some or all of the input parameter values.

The exact form of the base case is not critical to successful use of the model; it is improvements on this scenario due to CRM investments that are of key interest. The stochastic elements of the model are controlled using a sequence of random numbers. This list of numbers is identical for each run of the simulation, so the pattern of interactions in the base case matches that in the scenario representing CRM implementation. Any changes in market share can therefore be attributed to the CRM solution rather than statistical variations. It is possible to run the model many times using a different sequence of random numbers in order to calculate the average expected impact of a particular CRM intervention.

In the model, CRM interactions are characterised by their frequency, cost, and the impact they have on customer perception. If the benefits of a CRM solution can be expressed in these terms, then the appropriate parameters can be adjusted and the behaviour of the customer population examined. If the client can supply basic financial information, the tool can also compute an estimate of return on investment (ROI) at each time step. In a simple version of the model, customer contacts can be classified as positive or negative and given fixed impacts as appropriate. Figure 5 shows the result of increasing the probability of a positive experience by 1%, starting from the base case shown in Figure 4.

It can be seen that there is very little impact on market share for the first year after investment—ROI increases very slowly. However, during the second year the decline in market share is greatly reduced; returns increase rapidly, reaching breakeven after approximately three years. This illustrates that it can often take time for improvements in customer service to have a significant effect. Customers must interact with a company over a period of time before they become aware of changes, and it takes a further period for this improved perception to diffuse through the population. Although this example is based on fictional data, it illustrates the type of analysis enabled by the tool.

Figure 5. Market share over time for the original, base case (thin line), and after change in CRM strategy (thick line)

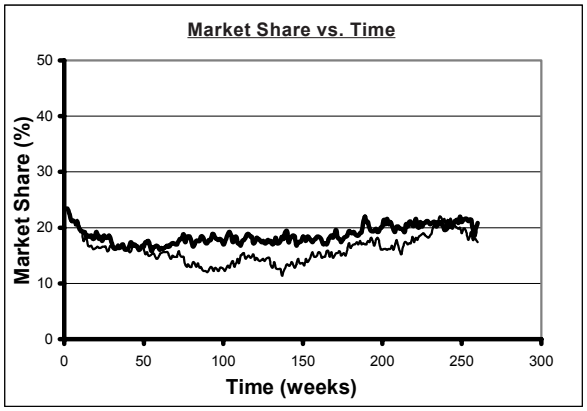
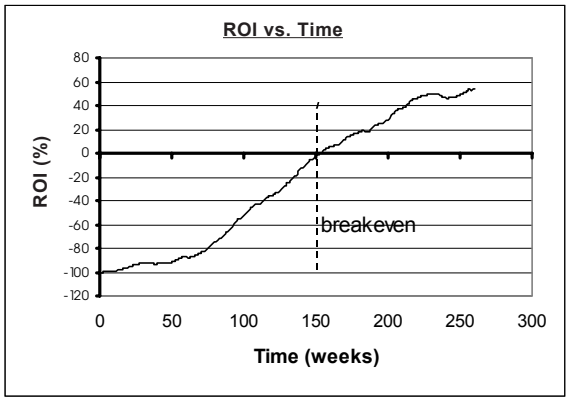


Figure 6. Return on investment as a percentage of the initial investment



CONCLUSION

The understanding of socio-economic systems is clearly a key problem faced by businesses. Modelling of these systems is one of the most effective ways of gaining this understanding. There are two main issues that are of consideration in the modelling process.

First the modelling technique should provide a useful representation of the system under study. This representation does not necessarily create a predictive model. The merit of a model is in its ability to provide insight into the problem under discussion. Outcomes sought include enhanced understanding of the problem space. This includes identification of the structures and processes that exist, the limitations of the knowledge held about the system, and the implications of the boundaries that are set on the problem space. These elements rely on the involvement of both the modeller and the end

user of the model, in the model construction. Collaboration not only extends the understanding of the problem by the user, but leads to a more accurate representation, improving the acceptance and value of the end product. Equally important is the understanding that is gained through the experimentation and exploration that a good model allows through the manipulation of the model parameters. These should ideally correspond to quantities that are manipulable or observable in the real system under consideration.

Secondly, the modelling technique should be able to capture the details, nuances, and behaviours of the real system under study. The majority of socio-economic systems are characterised by large numbers of interactions, the responses to which are adapting and evolving—in short they are complex systems. A modelling tool for these applications should be able to represent this level and type of sophistication.

Agent-based modelling is a technique that satisfies these two key issues. In our experience with clients, it provides an ideal way of representing a complex system, but also has the advantage of providing an excellent framework for the construction of a model. It can create the dialogue and understanding between the users of the model which ultimately is vital for the model's success.

REFERENCES

Amaral, L. A. N., Scala, A., Barthelemy, M., & Stanley, H. E. (2000). Retrieved from http://xxx.lanl.gov/abs/cond-mat/0001458

Arthur, W. B. (1994). *Increasing returns and path dependence in the economy.* Ann Arbor: University of Michigan Press.

Bass, F. M. (1969). A new product growth for model consumer durables. *Management Science, 15*(5), 215-227.

Bonabeau, E. (2002). Predicting the unpredictable. *Harvard Business Review, 80*(3), 109-116.

Callaway, D. S., Newman, M. E. J., Strogatz, S. H., & Watts, D. J. (2000). Retrieved from http://xxx.lanl.gov/abs/cond-mat/0007300

East, R. (2000, November 28-December 1). Loyalty: Definition and explanation. *Proceedings of ANZMAC,* Griffith University, Gold Coast, Queensland (pp. 286-290).

Fourt, L. A., & Woodlock, J. W. (1960). Early prediction of market success for new grocery products. *Journal of Marketing, 25,* 31-38.

Griffin, J. (1995). *Customer loyalty: How to earn it, how to keep it.* San Francisco: Jossey-Bass.

Hewson Consultancy Group. (2000, January). '*Making a compelling business case for CRM.* Retrieved from http://www.hewson.co.uk/sistrum/crm_management_insights/stream4/0001compelling_crm.pdf

Jager, W., Van Asselt, M. B. A., Rotmans, J., & Vlek, C. A. J. (1997). *Consumer behaviour: A modelling perspective in the context of integrated assessment of global change.* Globo Report Series #17, RIVM, Bilthoven, The Netherlands.

Mahajan & Peterson. (1985). *Models for innovation diffusion.* Newbury, CA: Sage.

Milgram, S. (1967). *Psychology Today 2,* 60-67.

Moore, C., & Newman, M. E. J. (2000). *Exact solution of site and bond percolation on small-world networks.* Working Paper 00-01-007, Santa Fe Institute, USA.

Newman, M. E. J. (1999). *Small worlds: The structure of social networks*. Working Paper 99-12-080, Santa Fe Institute, USA.

Newman, M. E. J., Strogatz, S. H., & Watts, D. J. (2000). *Random graphs with artibtrary degree disributions and their applications*. Retrieved from http://xxx.lnl.gov/abs/cond-mat/0007235

Newman, M. E. J., & Watts, D. J. (1999). Scaling and percolation in the mall-world network model. *Physical Review E, 60,* 7332-7342.

Rogers, E. M. (1995). *Diffusion of innovations.* New York: The Free Press.

Tesfatsion, L. (1997). How economists can get Alife. In W. B. Arthur, S. Durlauf, & D. Lane (Eds.), *The economy as a complex evolving system II. Proceedings of the Santa Fe Institute Studies in the Sciences of Complexity* (Vol. XXVII). Reading, MA: Addison-Wesley.

Watts, D. J. (2001). A simple model of fads and cascading failures. Working Paper 00-12-062, Santa Fe Institute, USA.

Watts, D. J., & Strogatz, S.H. (1998). *Collective dynamics of "small world" networks*. *Nature, 393,* 440-442.

KEY TERMS

Agent-Based Models: Simulations consisting of a number of discrete entities, each with their own rules of behaviour. These rules determine the interactions between the elements, and between the elements (agents) and the environment in which they are contained.

Complexity: Refers to tools, techniques, and approaches from the field of Complexity Science. Complexity Science is a highly interdisciplinary field dedicated to understanding complex systems. In this context a complex system is defined as a set of elements that often exhibit adaptation and interact in a non-linear fashion. The overall behaviour of a complex system can be counter-intuitive and difficult to predict by studying the individual components in isolation.

Emergent Behaviour: The macroscopic behaviour of a complex system emerges from the individual interactions of its constituent elements.

Network Externalities: The circumstance where a product's utility changes as the number of agents consuming it changes. A classic example of a product that exhibits network externalities is the fax machine: as more people purchase fax machines, users can communicate with a greater number of people, and the utility of the device increases.

Small-World Networks: The concept of a small-world network is now widely accepted and is defined as a social network where the chains of intermediate acquaintances required to connect any two individuals are small compared to the total number of people.

Social Networks: The links between individuals in a population.

Socio-Economic System: A type of complex system (see Complexity) that consists of social and economic elements. Descriptions of these systems tend to have significant qualitative elements and are difficult to analyse using traditional macroscopic techniques.

Section IX

Finance

Chapter LV
Genetic Programming for Spatiotemporal Forecasting of Housing Prices

Mak Kaboudan
University of Redlands, USA

ABSTRACT

This chapter compares forecasts of the median neighborhood prices of residential single-family homes in Cambridge, Massachusetts, using parametric and nonparametric techniques. Prices are measured over time (annually) and over space (by neighborhood). Modeling variables characterized by space and time dynamics is challenging. Multi-dimensional complexities—due to specification, aggregation, and measurement errors—thwart use of parametric modeling, and nonparametric computational techniques (specifically genetic programming and neural networks) may have the advantage. To demonstrate their efficacy, forecasts of the median prices are first obtained using a standard statistical method: weighted least squares. Genetic programming and neural networks are then used to produce two other forecasts. Variables used in modeling neighborhood median home prices include economic variables such as neighborhood median income and mortgage rate, as well as spatial variables that quantify location. Two years' out-of-sample forecasts comparisons of median prices suggest that genetic programming may have the edge.

INTRODUCTION

Techniques to analyze, model, and forecast spatiotemporal series are far from being established. Although statistical methods that analyze, model, and forecast time series are established, applying them to geographic or spatial data may be problematic. Analysis of spatial data using traditional econometric techniques (such as regression or maximum likelihood) may face spatial correlation, model misspecification, and spatial heterogeneity problems that jeopardize the accuracy of results. Further, advancements in spatial statistics do not offer modeling solutions. They offer measures of global spatial correlation like the Moran I and Geary's c, and of local spatial autocorrelation like G and G* (Haining, 2003).

If complex statistical modeling problems hinder analyzing spatiotemporal data, it seems logical and hopefully helpful to use techniques that circumvent statistical estimation of model parameters. This chapter examines whether use of genetic programming (GP) or artificial neural networks (ANNs) can produce applicable and capable forecasting models. The spatiotemporal variable to model and forecast is annual residential single-family home median neighborhood prices. Values of this variable are discrete time series (collected over a number of years) representing 12 neighborhoods. Model-dependent and independent variables' values associated with space (neighborhoods) are identified by i, where i = 1, …, n locations. Values collected at equally spaced time intervals are identified by t, where t = 1, …, T periods. The objective is to model and forecast the spatial univariate time series price variable P_{it}, where P_{it} is a *K* x 1 vector with *K* = n*T. A general specification to adopt is:

$$P_{it} = f(S_i, X_t, Z_{it}) \quad (1)$$

where S_i is a set of spatial variables that vary across regions (i) but not over time (t), X_t is a set of time series variables that vary over time but remain constant across regions, and Z_{it} are spatiotemporal variables that vary over both.

It is well known that forecasting residential housing prices is important. Decisions made by lending institutions, tax assessors, and homeowners or buyers are affected by the real estate market dynamics and price predictions. Yet, accurate price-predictions remain a challenge, perhaps because models that explain variations in prices over time as well as between neighborhoods and among houses can be rather complex. Most studies that forecast prices of residential homes rely on property-address-level detailed data. Details at property level furnish housing attributes upon which hedonic pricing models have been based for decades. Applications of hedonic methods to the housing markets are plenty; see Goodman and Thibodeau (2003) for a recent application. Advances based on hedonic methods include work on geographically weighted regression models (Fotheringham, Brunsdon, & Charlton, 2002) and work on local regression (or semi-parametric) models (Clapp, Kim, & Gelfand, 2002). Bin (2004) compares parametric vs. semi-parametric hedonic regression models. Hedonic models can be viewed as representations of localized nano-detailed pricing algorithms since they focus on small-scale variations. They are designed to capture the effects of differences in housing attributes on prices. Housing attributes include quantitative and qualitative characteristics of individual properties such as age, square footage, with garage or not, with air conditioning or not, and so forth, as well as location factors that impact the price of a house (Bin, 2004; Mason & Quigley, 1996). These, however, do not explain temporal or dynamical price changes.

Dynamical changes in housing prices are determined by supply and demand forces. Determinants of demand such as income and mortgage rate change over time. Such temporal changes in determinants cause prices of all residential houses to change over time, *ceteris paribus* attributes. It is dynamical changes and not attributes that cause appraisers to adjust the price of the same exact house over time. They adjust prices to reflect a known "current" neighborhood median price while allowing for differences among properties. If this is the case, it is reasonable to model local patterns of dependency that capture the impact of large-scale variations of inter-neighborhood temporal changes in economic conditions on neighborhood median prices first. What follows are hedonic models that capture the impact of intra-neighborhood variations to account for differences in age, square footage, and so on. A neighborhood median price becomes one of the

input variables in a hedonic model. Including cross-sectional economic variables along with attributes for a single time period in the same price equation is possible and has been done (e.g., Clapp, 2004). However, including temporal economic variables that remain constant among neighborhoods, as well as spatiotemporal variables that vary among neighborhoods and over time in the same equation, has not been attempted.

Special attention is given in this research to delivering timely forecasts of residential housing prices. A model that forecasts "known information" is of little value to decision makers. For example, models capable of forecasting only one-step-ahead (a month, quarter, or year) tend to deliver known information. By the time data on explanatory variables needed to produce a forecast become available, that one-step-ahead forecast value has materialized. Producing an extended forecast (i.e., one that goes beyond one period) becomes a clear and logical alternative. To produce an extended forecast, typically "predicted values" of explanatory variables are used. This may deliver inaccurate forecasts because predicted values of input variables are most probably imprecise. It is more logical then to construct models that can produce forecasts for a few steps ahead using "actual" rather than "fitted or forecasted" values of explanatory variables. In this chapter, it is shown that using distant lagged actual values of explanatory variables delivers reasonable forecasts of neighborhood residential housing prices. Attention to this problem is not new, but the solution suggested here is. Gençay and Yang (1996), Clapp et al. (2002), and Bin (2004) emphasize evaluating out-of-sample predictions when comparing performances of different forecasting models.

Over the past decade, statistical analysis of spatiotemporal series earned reasonable attention. Getis and Ord (1992), Anselin, Bera, Florax, and Yoon (1996), and Longley and Batty (1996, pp. 227-230) discuss spatial autocorrelation and other statistical complications encountered when analyzing or modeling spatial data. Anselin (1998, 1999) reviews potential solutions to some of these problems. However, few address complications that occur when spatial data is taken over time. Little to no attention was given to modeling nonlinear spatiotemporal systems. If such models account for the presence of low-dimensional nonlinear or chaotic dynamics, a rapid increase in forecast errors over time occurs due to sensitivity to initial conditions. This issue was addressed by Rubin (1992) and Maros-Nikolaus and Martin-González (2002). Nonparametric modeling techniques adopted here to capture what may be nonlinear spatiotemporal dynamics of the housing market may therefore be appropriate.

The two computational techniques applied to predict median neighborhood prices are quite different. GP is a computer algorithm that can be configured to produce regression-type models (Koza, 1992). The traditional statistical calculations to estimate model coefficients and the restrictions imposed by statistical models are totally absent. GP is a univariate modeling technique that typically delivers nonlinear equations. They are difficult to interpret, but forecast rather well. López, Álvarez, and Hernández-García (2000) proposed using a proper empirical orthogonal function decomposition to describe the dynamics of a system, then used GP to extract dynamical rules from the data. They apply GP to obtain one-step-ahead forecast of confined spatiotemporal chaos. Examples of applications of GP to economic forecasting include work by Chen and Yeh (2000), Kaboudan (2000), Neely and Weller (2001), Kaboudan and Liu (2004), and Tsang, Yung, and Li (2004). A thorough review of applications of GP in financial forecasting can also be found in Chen (2002). ANN is a computerized classification technique that delivers forecasts, but (unlike GP) without delivering a

model. ANN architecture is based on the human neural system. It is programmed to go into a training iterative process designed to learn the dynamics of a system. Compared with GP, ANN is more established with superior power in fitting complex dynamics, and has gained attention and acceptance of many forecasters. Gopal and Fischer (1996) used ANN in spatial forecasting. Rossini (2000) used it in forecasting residential housing prices employing hedonic-type specifications. Examples of applications of ANN to economic forecasting include work by Swanson and White (1997a, 1997b), Bennell and Sutcliffe (2004), Clements, Franses, and Swanson (2004), and Huang, Lai, Nakamori, and Wang (2004). Further, Dahl and Hylleberg (2004) gave a recent great review of economic forecasting using artificial neural networks. They compared performance of ANN and three other approaches in forecasting U.S. industrial production and unemployment rate. Attraction to GP and ANN may be attributed to their robustness with respect to many statistical problems that standard econometric or statistical modeling methods face. More specifically, they are robust against problems of multicollinearity, autocorrelation, and nonstationarity.

This investigation implements a specification strategy designed to account for the effects of temporal variations on future neighborhood median single-family residential home prices in the City of Cambridge, Massachusetts. What is presented here is of exploratory nature. The main assumption is that changes in real mortgage rate and in real per capita income have different effects on prices in the different neighborhoods. Although only data available freely via the Web was employed, results reported below seem promising.

The next section contains a description of the data, followed by an explanation of the univariate model specification employed throughout. An introduction to GP and how it can be used in forecasting, as well as a brief review of ANN, are then offered. Forecast results using standard statistical method—*weighted least squares* (WLS), GP, and ANN—are compared, and the final section contains concluding remarks. The results reported below are mixed. Forecasts using GP were significantly more reasonable and more likely to occur than those obtained by ANN or WLS.

INPUT DATA

The data utilized in this study is on housing sales for the City of Cambridge, Massachusetts. It was the only set of data found with complete information to use in developing a housing median price model. The data can be downloaded from a site published by that city's Community Development Department, Community Planning Division (2003). It contains annual median prices of single-family homes for the period 1993-2002 of 12 (out of 13) neighborhoods in Cambridge. The city stopped reporting this type of data after 2002. Figure 1

Figure 1. City of Cambridge: Neighborhood boundaries with major city streets (http://www.ci.cambridge.ma.us/~CDD/commplan/neighplan/ pdfmaps/neighcitymap.html)

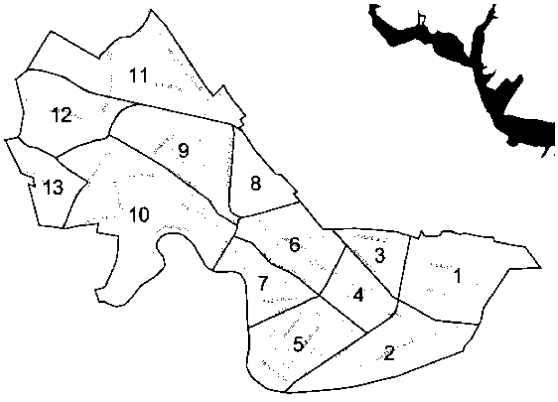
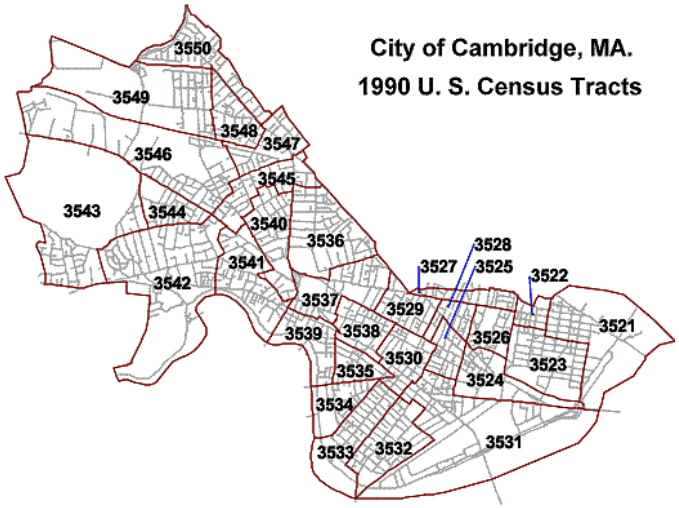


Figure 2. City of Cambridge: Census tracts (http://www.ci.cambridge.ma.us/~CDD/data/maps/1990_census_tract_map.html)



shows the 13 neighborhoods. (Data was available only for 12 of them. There were no data for the strictly commercial area 2.]

To minimize the impact of outliers, a neighborhood annual median price was included in the final data set only if the number of sales in that neighborhood was at least three. Only six years of data (1995-2000 inclusive) were used to find forecasting models. Data of the first two years were lost degrees of freedom due to lags. There were 67 observations to use in the six-year training set. Data available for the last two years (2001 and 2002, consisting of 21 observations) were reserved to evaluate and compare out-of-sample forecasts produced by the three models.

The explanatory variables considered to capture variations in P_{it} follow. The consumer price index for Boston was used to deflate all nominal data. Thus, prices, income, and mortgage rate are in real terms.

- MR_{t-2} = Two years lagged real mortgage rate. This temporal variable is constant across neighborhoods.
- PCI_{it-2} = Neighborhood real per capita income lagged two years. This spatiotemporal variable varies over time as well as among neighborhoods. Per capita income was not available by neighborhood but by census tract. Figure 2 shows these tracts. To obtain representation values of each neighborhood's per capita income, an approximate congruency between maps of neighborhood and tracts in Figures 1 and 2 was used. Neighborhood 1 contains approximately three tracts (3521, 3522, and 3523), for example. Their distribution was visually approximated. Tracts 3521, 3522, and 3523 were subjectively assigned weights of 0.60, 0.30, and 0.10, respectively, to reflect their proportional neighborhood shares. A weighted average per capita income was then computed for that neighborhood. Others were similarly approximated. Using this method may be a source of measurement error. Since there is no alternative, this solution was assumed reasonable given the contiguity of neighborhoods. The results obtained and

presented later seem to confirm that such PCI representation seems adequate.

- $DVPCI(i)_{t-2}$= Twelve spatiotemporal dummy variables designed to capture the impact of changes in real per capita income in neighborhood i at time period t-2 on that neighborhood median price in period t. $DVPCI(i)_{t-2} = PCI_{t-2} * W_{ij}$, where $W_{ij} = 1$ if i = j and zero otherwise for i = 1, .., n neighborhoods and j = 1, …, k neighborhoods where n = k. Use of this type of dummy variable is *atypical*. Goodman and Thibodeau (2003) use dummy variables to account for time of sale. Gençay and Yang (1996) use more standard Boolean dummy variables for (all but one) neighborhoods. Boolean dummy variables capture intercept shifts. $DVPCI(i)_{t-2}$ captures slope changes instead.
- Y_{t-2} = Real median household income for the City of Cambridge, Massachusetts. This temporal variable is constant across neighborhoods.
- $RNPR_i$ = Relative neighborhood price ranking. This ranking was computed by averaging each neighborhood's median price over the period 1993-2000 first, sorting these averages second, and then assigning the numbers 1 through 12 with "1" assigned to the lowest and "12" to the highest average. This integer spatial variable varies by neighborhood only and serves as polygon ID.
- LAT_i = Latitude of the centroid of neighborhood i.
- LON_i = Longitude of the centroid of neighborhood i.
- AP_{t-2} = Average real median price of homes in the City of Cambridge, Massachusetts, lagged two years. This temporal variable remains constant across neighborhoods.
- P_{it-2} = Real median price lagged two periods.

Given that P_{it} and a few of the explanatory variables vary spatially, detecting presence or absence of spatial correlation should be tested one way or another. Measuring spatial autocorrelation using the Moran coefficient or the Geary ratio is not practical here since data is taken over time as well. Instead, spatial autocorrelation was estimated using the following OLS regression model:

$$P_{it} = \alpha + \rho\, P_{i-1t} \qquad (2)$$

where ρ measures the degree of autocorrelation. This equation provides a simple but reasonable approximation of autocorrelation between pairs of contiguous neighbors over time. Autocorrelation is present if the estimated ρ is significantly different from zero. The estimated equation was as follows:

$$P_{it} = 138.9 + 0.452\, P_{i-1t} \qquad (3)$$

The equation was estimated using 1995-2002 data. The intercept and the estimate of ρ's *p-value* = 0.00). Thus, equation 3 confirmed the existence of spatial autocorrelation between pairs of contiguous neighborhoods averaged over time.

BASIC MODEL SPECIFICATION

The basic model is a univariate specification designed to capture variations in P_{it} using mainly economic data while incorporating spatial aspects. Formally:

$$P_{it} = f(S_i, X_t, Z_{it}) \qquad (4)$$

The set of spatial variables S_i that vary only among neighborhoods includes $RNPR_i$, LAT_i, and LON_i. The set of time series variables X_t that vary only over time (t) includes MR_{t-2}, Y_{t-2},

and AP_{t-2}. Z_{it}, or the set of variables that vary over both includes $DVPCI_{i,t-2}$, $PCI_{i,t-2}$, and $P_{i,t-2}$.

To evaluate the efficacy of forecasts from the two nonparametric computational techniques GP and ANN, a forecast using standard statistical methods is obtained. Because cross-sectional and time series data are pooled, and because spatial correlation was detected, the variance of the error *Ordinary least squares* (OLS) delivers will not be constant over observations. When this problem occurs, the model has heteroscedastic error disturbance. OLS parameter estimators are unbiased and consistent, but they are not efficient (Pindyck & Rubinfeld, 1998, p. 147). The *weighted least squares* (WLS) method (which is a special case of *generalized least squares;* Pindyck & Rubinfeld, 1998, p. 148) is used to correct for heteroscedasticity if it is detected.

To test for heteroscedasticity, the Goldfeld-Quandt test applies (Goldfeld & Quandt, 1965). The null hypothesis of homoscedasticity is tested against the alternative of heteroscedasticity. To conduct the test, a few steps are followed. First, a single explanatory variable thought to be responsible for the error variance is identified. Second, the dependent and the explanatory variables' data are sorted in an ascending order according to the dependent variable's values. Third, available observations are divided into three (approximately equal size) groups. A simple regression model is then fit to the first and last groups. The price data were assumed most affected by RNPR (Relative neighborhood price ranking). Two regressions P_{it} = $f(RNPR_i)$ were then estimated using the top and the bottom 25 observations. (The 17 middle observations were ignored.) The two estimated equations yielded two residual sum of squares values: $RSS_1 = 23272.29$ and $RSS_2 = 90508.77$. The ratio of the residual sum of squares (RSS) from the two equations follows an F distribution, and the F-statistic = $RSS_2/RSS_1 = 3.89$. Under the null of homoscedasticity with 23 degrees of freedom in the numerator and the denominator and at the 5% level of significance, the F critical value is 2. Since the F-statistic = 3.89 > critical F = 2, the null is rejected in favor of heteroscedasticity, and equation 4 is estimated using WLS instead of OLS.

Table 1 contains the estimated WLS model obtained to forecast P_{it}. All estimated coefficients are significantly different from zero at the 1% level of significance except for the estimated MR_{t-2} coefficient. It is statistically different from zero at the 10% level of significance. Coefficients of all other explanatory variables considered to include this regression were deleted either because they were statistically equal to zero at more than the 20% level of significance or because their sign was illogical. The equation below succeeded in explaining 76% of median price variations ($R^2 = 0.76$). Further, given that the Durgin-Watson (DW) statistic is very close to 2, there seems to be no autocorrelation problem. The mean squared error or MSE = $[\Sigma(Actual_t - Fitted_t)^2/T]$ is reported at the bottom of the table for later comparisons. Actual vs. fitted values from this model are presented in Figure 3.

The main strength in using WLS lies in the interpretation of the estimated coefficients. Although it is difficult to tell if they are totally meaningful, their signs seem to be at least consistent with expectations. Estimated coefficients in the table suggest the following:

a. If average city price two years earlier (AP_{t-2}) increased by $1,000, current median neighborhood prices (P_{it}) would increase by $959.
b. If mortgage rates (MR) decreased two years earlier by 1%, P_{it} would increase by $27,484 on average.
c. If per capita income in neighborhood 1 (PCI_1) increased two years earlier by $1,000, current prices in that neighbor-

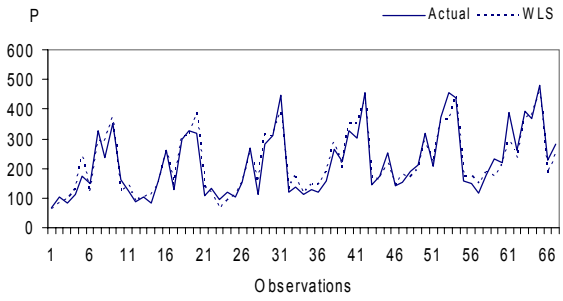
Table 1. Weighted least squares regression estimated coefficients

Variable	Coeff.	Std. Error	t-Stat	*p-value*
AP_{t-2}	0.959	0.406	2.362	0.022
MR_{t-2}	-27.484	13.987	-1.965	0.055
$DVPCI(1)_{t-2}$	23.420	9.407	2.490	0.016
$DVPCI(3)_{t-2}$	31.801	12.181	2.611	0.012
$DVPCI(4)_{t-2}$	21.647	7.971	2.716	0.009
$DVPCI(5)_{t-2}$	11.882	4.079	2.913	0.005
$DVPCI(6)_{t-2}$	10.521	2.869	3.667	0.001
$DVPCI(7)_{t-2}$	27.662	9.568	2.891	0.006
$DVPCI(8)_{t-2}$	19.833	4.748	4.177	0.000
$DVPCI(9)_{t-2}$	9.279	2.371	3.913	0.000
$DVPCI(10)_{t-2}$	6.587	1.608	4.097	0.000
$DVPCI(11)_{t-2}$	6.764	3.114	2.172	0.034
$DVPCI(12)_{t-2}$	13.212	4.775	2.767	0.008
$DVPCI(13)_{t-2}$	15.332	4.821	3.180	0.002
Latlon	-2.650	1.261	-2.102	0.040
$R^2 = 0.76$				
DW = 1.76				
MSE = 1061.23				

hood (P_{1t}) would increase by $23,420. If PCI_2 in neighborhood 2 increased by $1,000 two years ago, P_{2t} would increase by $31,801, and so on.

The last variable in the table *Latlon* was inspired by equation 15 in Clapp (2004): $Latlon_i = (LON_i * LAT_i)^2$. It was selected after a lengthy search for the proper location variable(s) to include among the variables in equation 4. It was the only variable that was not highly collinear with any other variable in the equation. The variable is interesting because the value the estimated coefficient takes is not as important as its sign. If the estimated coefficient is positive, it suggests that prices tend to be higher in the east, north, and north-east of the city. A negative sign suggests that prices tend to be higher in the west, south, and south-west. As the equation suggests neighborhoods of Southern and Western Cambridge are more expensive, which is consistent with actual data.

Figure 3. Actual historical values of median prices and their weighted least squares regression model fitted values



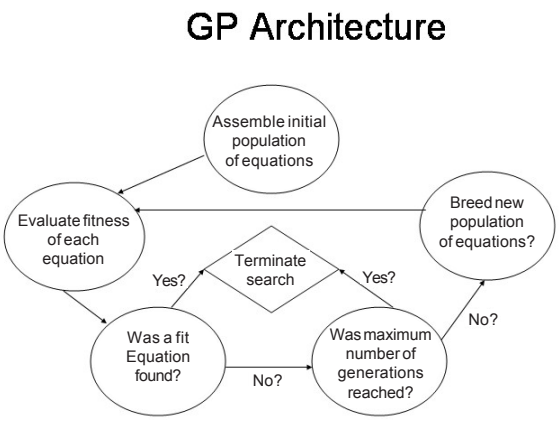
GENETIC PROGRAMMING AND NEURAL NETWORKS

Genetic Programming

Foundations of GP are in Koza (1992). GP is an optimization technique in the form of a computer program designed to emulate Darwin's notion of survival of the fittest. The program is capable of solving diverse optimization problems from many disciplines, but used here to obtain a regression model. Figure 4(a) depicts a basic GP architecture used to search for an optimal equation to use in forecasting.

As Figure 4 shows, GP evolves thousands of model specifications, solves each for fitted values, tests their fitness, and delivers a final best-fit model to use. To obtain a best-fit equation, the computer program starts by randomly assembling an initial population of equations (say 100, 1,000, or even 5,000 of them). The user determines the size of such population. The user also provides variables' data input files. To assemble equation members of a population, the program randomly combines a few variables with randomly selected mathematical operators such as +, -, *, protected /, protected $\sqrt{}$, sine, cosine, among others. Protected division and square root are necessary to prevent

Figure 4(a). Process depicting evolution of equations to select a best-fit one using GP



division by zero and taking the square root of negative numbers. These protections follow standards most GP researchers agree upon to avoid computational problems. More specifically, these protections are programmed such that if in $(x \div y)$, $y = 0$, then $(x \div y) = 1$. And if in $y^{1/2}$, $y < 0$, then $y^{1/2} = -|y|^{1/2}$. Once members of a population of equations are assembled, their respective fitness is computed using one of several choices of fitness measures available. The mean square error (MSE) is most typically used. The equation with the lowest MSE in a population is declared fittest. If at any time an equation accurately replicates values of the dependent variable, the program terminates. Accuracy is determined according to a user-controlled threshold minimum MSE. If GP does not find an equation with MSE = the predetermined Min (MSE), which usually happens, the program breeds a new population. Populations succeeding the initial one are the outcome of a programmed breeding by cloning or self-repro-

Figure 4(b). Example of crossover or breeding between two Individuals (1 & 2); they breed individuals 3 and 4

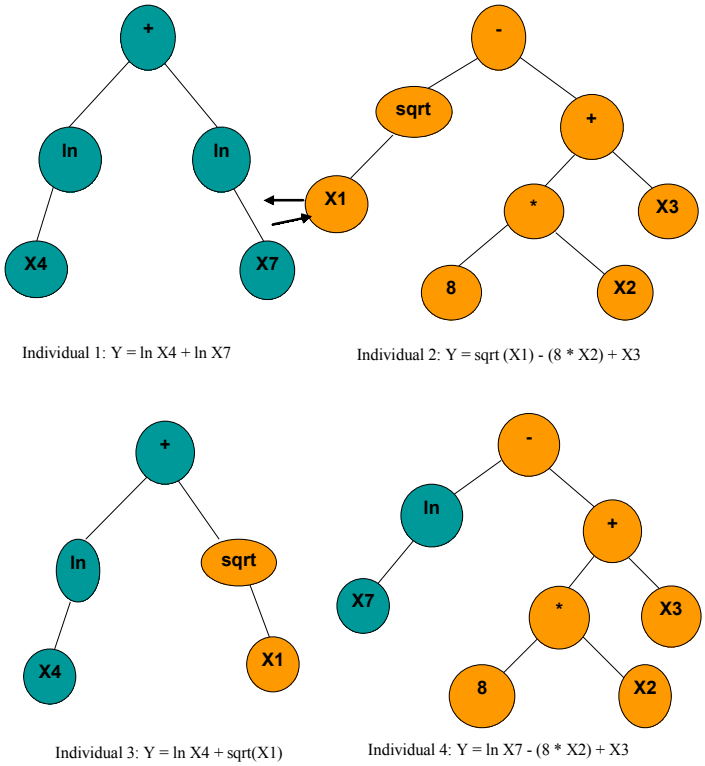
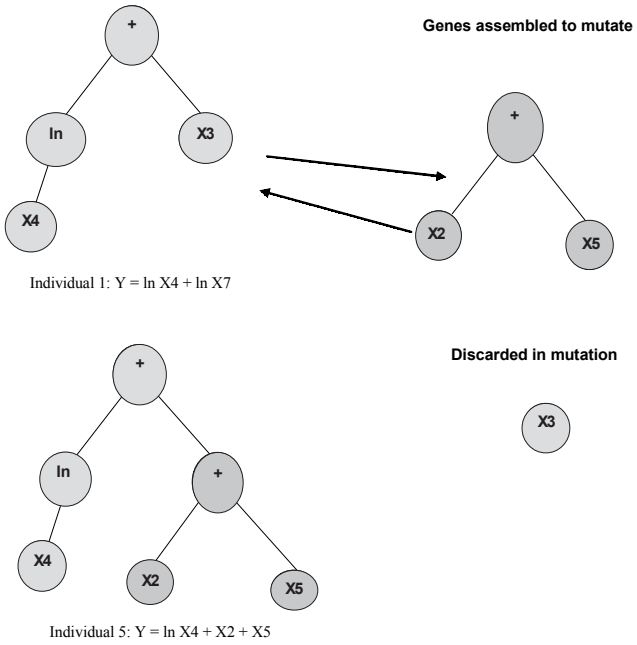


Figure 4(c). Example of mutation breeding using only one Individual (1); the newly bred individual 5 is the outcome



duction, crossover, and mutation. In self-reproduction, the top percentage (say 10 or 20%) best equations in an existing population are simply copied into the new one. In crossover, randomly selected sections from two (usually fitter) equations in an existing population are exchanged to breed two offspring. Figure 4(b) shows an example of crossover between two hypothetical equations. Each equation is represented by a tree structure (known as a parse tree) for easy programming.

In mutation, a randomly selected section from a randomly selected equation in an existing population is replaced by newly assembled part(s) to breed one new member. Figure 4(c) shows an example of mutating the first equation shown in Figure 4(b).

GP continues to breed new generations until an equation with Min (MSE) is found or a preset maximum number of generations is reached. The equation with Min (MSE) in the last population bred is then reported as fittest.

TSGP (time series genetic programming; Kaboudan, 2003) software is used to obtain GP price models here. TSGP is a computer code written for the Windows environment in C++ that is designed specifically to obtain (or evolve in a Darwinian sense) forecasting models. There are other GP software (both commercial and freely available to download) available and can be used to produce similar results. However, whether the same exact software or a different one is used, replicating the exact outcomes is impossible because, by definition, GP is a random search technique. TSGP is presented here only because it is available to download freely and it is rather easy to use. Its user has to prepare two types of input: data input files and a configuration file. Data values of the dependent and each of the independent variables

Table 2. Koza Tableau containing run parameters

Parameter	Value
Population size	1000
Maximum number of generations	100
Mutation rate	0.6
Crossover rate	0.2
Cross-self rate	0.2
Operators	+, -, *, /, sqrt, sin, & cos
Selection method	Roulette wheel
Maximum tree depth	100
Fitness measure	MSE
Terminal set	P_{it-2}, Y_{t-2}, PCI_{it-2}, MR_{t-2}, AP_{t-2}, $DVPCI(i)_{t-2}$, $RNPR_i$, LAT_i, LON_i, *Latlon*
Number of searches completed per run	100

must be supplied in separate ASCII text files. The user does not have to modify much or compile any program. A configuration file containing execution information including name of the dependent variable, number of observations to fit, number of observations to forecast, number of equation specifications to evolve, and other GP-specific parameters TSGP prompts the user with questions upon execution. Before executing TSGP, run parameters must be selected however. TSGP default parameter values are set at population size = 1000, number of generations = 100, self-reproduction rate = 20%, crossover rate = 20%, mutation rate = 60%, and number of best-fit equations to evolve in a single run = 100. The GP literature on selection of these parameters has conflicting opinions and therefore extensive search is used to determine what is best to use under different circumstances. For example, while Chen and Smith (1999) favor higher crossover rate, Chellapilla (1997) favors higher mutation rate, and while Fuchs (1999) and Gathercole and Ross (1997) favor small population sizes over many generations, O'Reilly (1999) argues differently. Discussions on parameter selection options are in Banzhaf, Nordin, Keller, and Francone (1998). With such disagreements, trial and error helps in choosing the appropriate parameters. This situation is aggravated by the fact that assembling equations in GP is random and the fittest equation is one that has global minimum MSE. Unfortunately, GP software typically gets easily trapped at a local minimum while searching for that global MSE. It is therefore necessary to compare a large number of best-fit equations (minimum of 100) to identify or obtain that "best" one. Table 2 contains most of the information used to complete the GP runs in this study. The table is known in the literature as a 'Koza Tableau'.

TSGP produces two types of output files for each run. One has a final model specification. The other contains actual and fitted values as well as performance statistics such as R^2, MSE, and the mean absolute percent error (or MAPE = $[T^{-1}\sum - \text{Fitted}_t| / \text{Actual}_t]$). TSGP also delivers an Excel-formatted file with summary statistics of runs to facilitate identification of the best one.

GP delivers equations that may not reproduce history very well, but they may forecast well. A best-fit model fails to forecast well when the algorithm used delivers outcomes that are too fit. This phenomenon is known as overfitting. (See Lo & MacKinlay, 1999, for more on overfitting.) Generally, if an equation produces accurate out-of-sample *ex post* fore-

casts, confidence in and reliability of its *ex ante* forecast increases. An *ex post* forecast is one produced when dependent variable outcomes are already known but were not used in estimating the model. An *ex ante* forecast is one produced when the dependent variable outcomes are unknown. While evaluating accuracy of the *ex post* forecast, historical fitness of the evolved model cannot be ignored however. If the model failed to reproduce history, most probably it will not deliver a reliable forecast either. The best forecasting model is therefore identified in two steps. First, the final 100 fittest equations evolved are sorted according to lowest fitting MSE. Those equations with the lowest 10 MSE (or 20, arbitrarily set) are then sorted according to *ex post* forecasting ability. Forecasting ability can be measured according to prediction MSE (MSPE) or the mean absolute percent error (MAPE). That equation among the selected 10 with the best forecasting ability is selected as best to use for *ex ante* forecasting. This heuristic approach seems to work well. The idea is that if a model reproduces history reasonably well and if it simultaneously forecasts a few periods ahead fairly accurately, then it has a higher probability of successfully forecasting additional periods into the future.

When TSGP was executed, the fittest among the resulting 100 best-fit equations identified was:

$$P_{it} = PTI_{t-2} * \cos((AP_{t-2} / MR_{t-2}) / PTY_{t-2}) + RNPR_i + PTI_{t-2} * \cos(AP_{t-2} / PTY_{t-2}) + 2 * PTI_{t-2} * \cos(RNPR_i) + 2 * PTI_{t-2} * \cos(PTY_{t-2}) + PTI_{t-2} * \sin(PTY_{t-2}) + \cos(P_{it-2}) + P_{it-2} + PTI_{t-2} * \sin(P_{it-2} + (AP_{t-2} / PTY_{t-2})) + AP_{t-2} / MR_{t-2} \quad (5)$$

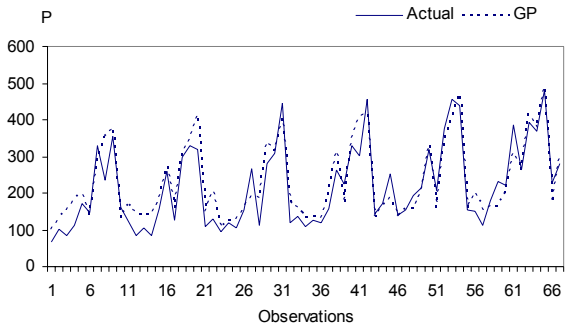
where $PTY_{t-2} = P_{it-2}/Y_{t-2}$ and $PTI_{t-2} = P_{it-2}/PCI_{it-2.}$ Its $R^2 = 0.83$, but the MSE = 1995.98 which is much worse than that WLS delivered. According to the above nonlinear equation, lagged median prices, relative neighborhood price ranking, mortgage rate, ratio of price to city income, and ratio of price to neighborhood per capita income as well as the spatial variable RNPR seem to have succeeded in fitting and forecasting P_{it} best. Actual and GP-fitted median home prices by neighborhoods are in Figure 5.

Neural Networks

Neural networks architecture is a more established computational technique than GP. The literature on ANN is huge. Principe, Euliano, and Lefebvre (2000), among many others, provide a complete description on how ANN can be used in forecasting. Two structures are commonly used in constructing networks: multilayer perceptrons (MLP) and generalized feed-forward networks (GFF). MLP is a layered feed-forward network that learns nonlinear function mappings. It employs nonlinear activation functions. Networks are typically trained with static back-propagation and require differentiable, continuous nonlinear activation functions such as *hyperbolic tangent* or *sigmoid.* A network utilizes input data of all variables to learn how to produce their closest fitted values. Although MLP trains slowly and requires a large number of observations to train, it is easy to use and approximates well. GFF is a generalization of MLP with connections that jump over layers. GFF also trains with static back-propagation. The forecast of neural networks reported below was produced using NeuroSolutions software (2002).

To obtain the best forecast of median neighborhood prices, both MLP and GFF were attempted employing the same set of spatiotemporal data GP and WLS used. First, base MLP and GFF configurations were selected as a starting point to identify the suitable network structure to use. Both are with one hidden

Figure 5. Actual historical values of median prices and their GP model fitted values



layer, use *hyperbolic tangent* transfer function, employ 0.70 learning momentum, and train only 100 epochs. These parameters were then changed one at a time until the best *ex post* forecasting network was identified. Hidden layers tested were set to one, two, and three. Transfer functions tested under each scenario were *hyperbolic tangent* and *sigmoid*. The better networks were then trained using learning rules with momentum set first at 0.7 once, then set at 0.9. Testing of each configuration that started with 100 training epochs was increased by increments of 100 until the best network is identified. The parameter used to terminate training was MSE. (MSE was selected to be consistent with GP.) Unfortunately, neural networks can overfit when training goes on for too long. The final outcome selected is therefore one that minimizes both training MSE as well as the impact of overfitting.

In the final (best-obtained) ANN configuration to use in forecasting, only 10 variables were used after experimenting with all of the available ones. Through trail and error, variables were deleted until the best forecasting configuration was found. For example, the 12 $DVPCI_{it-2}$ produced very poor forecasts and had to be removed from among input variables used. Table 3 contains most of the information used to complete the ANN runs in this study.

The final network structure selected for the purpose of this study was MLP with one hidden layer. It is depicted in Figure 6.

Actual and ANN-fitted values of median prices by neighborhoods over the years are in Figure 7. The best network configuration was a network with a *hyperbolic tangent* transfer function and with learning momentum = 0.90. Only1,700 training epochs were used. The best ANN configuration produced $R^2 = 0.96$ and MSE = 463.51.

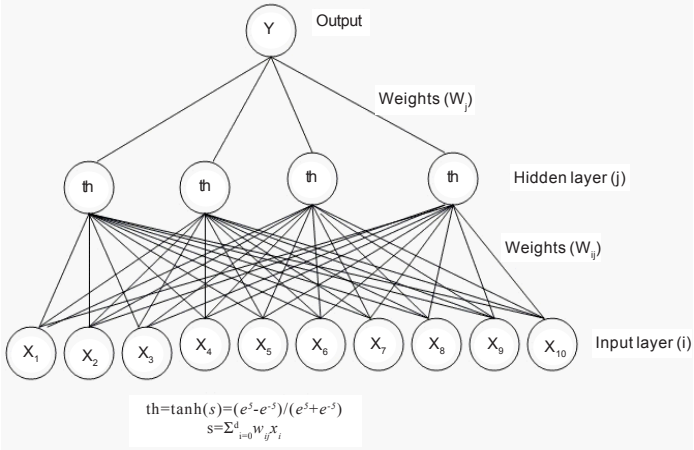
Comparison of Modeling Results

ANN produced better fit of P_{it} training values than the final GP model. It is important to reconfirm here that a technique that succeeds in reproducing history (data used to obtain models or in training) may not necessarily be the one that delivers the best forecast. Table 4 contains comparative statistics on estimation and training results using WLS, GP, and ANN. Based on these statistics, ANN ranked best and GP ranked worst. Figures 5-7 confirm ANN's outstanding ability in reproducing history when compared with WLS and GP. The figure depicts the 67 observations representing the six years (1995-2000), where observations 1-12 belong to 1995, 13-24 belong to 1996, and so on. Determining which of the three produces the best out-of-sample (2001 and 2002) forecasts and not the best fit of the historical data

Table 3. Run parameters used to complete ANN runs

Parameter	Value
Hidden layers	1, 2, & 3
Transfer functions	*hyperbolic tangent* and *sigmoid*
Learning rules momentum	0.7 & 0.9
Training epochs	100 incremented by 100 until best forecast is obtained
Fitness parameter	MSE
Terminal set	P_{it-2},, Y_{t-2}, PTY_{t-2}, PCI_{it-2}, MR_{t-2}, AP_{t-2}, $RNPR_i$, LAT_i, LON_i, *Latlon*.

Figure 6. MLP network used to generate the final forecast



is the main objective however. This is presented next.

FORECASTING

Although ANN delivered best fit in reproducing historical data used in training (1995-2000), forecasts of the WLS model were better than those of ANN, and forecasts of GP were better than those of WLS. Table 5 contains a comparison of the forecast statistics. The Theil's U-statistic reported in the table is a measure of forecast performance. It is known as Theil's inequality coefficient and is defined as:

$$U = \frac{\sqrt{MSE}}{\sqrt{k^{-1}\sum_{j=1}^{k} P_{ik}^2} + \sqrt{k^{-1}\sum_{j=1}^{k} \hat{P}_{ik}^2}} \qquad (6)$$

where j = 1, 2, …, k (with k = 21 forecasted observations representing 2001 and 2002), and $\hat{P}_{ik}$ are forecast values of $\hat{P}_{ik}$. This statistic will always fall between zero and one, where zero indicates a perfect fit (Pindyck & Rubinfeld, 1998, p. 387). NMSPE (=MSPE / variance ($\hat{P}_{ik}$)) is normalized prediction MSE.

Figure 7. Actual historical values of median prices and their ANN-fitted values

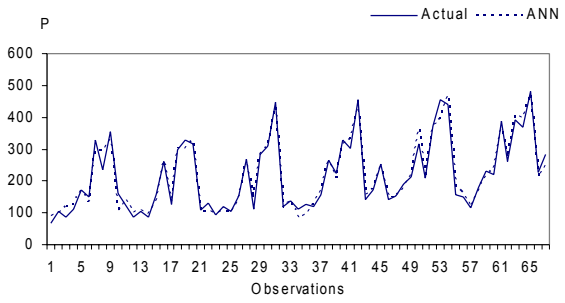


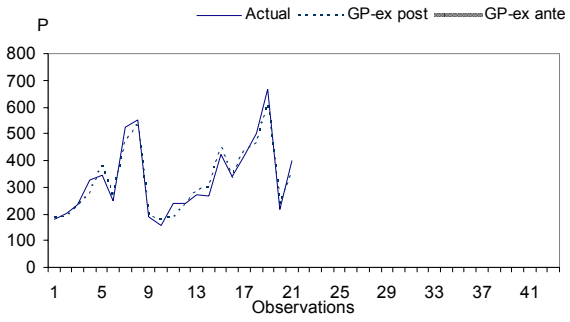
Table 4. Comparative estimation and training statistics

Statistic	WLS	GP	ANN
R^2	0.76	0.83	0.96
MSE	1061.23	1995.98	463.51
MAPE	0.130	0.212	0.091

Table 5. Comparative forecast statistics

	WLS	GP	ANN
Theil's U	0.081	0.044	0.097
MAPE	0.153	0.079	0.156
MSPE	3331.6	981.9	4425.26
NMSPE	0.170	0.079	0.225

Figure 9. Actual values of median prices, and their GP ex post and ex ante forecast values



Forecasts obtained using the three techniques are in Figures 8-10. Each shows actual values reserved for out-of-sample forecasts, their *ex post* forecast, and the *ex ante* forecast.

Two observations are easily deduced from the comparison in Table 5 and Figures 8-10. First, over the *ex post* forecast period, GP clearly delivers the best forecast. Second, and more importantly, the *ex ante* forecast GP delivers seems more logical than that of WLS or ANN. ANN has underestimated prices in areas where homes are relatively more expensive.

CONCLUSION

This chapter explored computational forecasting of neighborhood median residential housing

Figure 8. Actual values of median prices, and their WLS ex post and ex ante forecast values

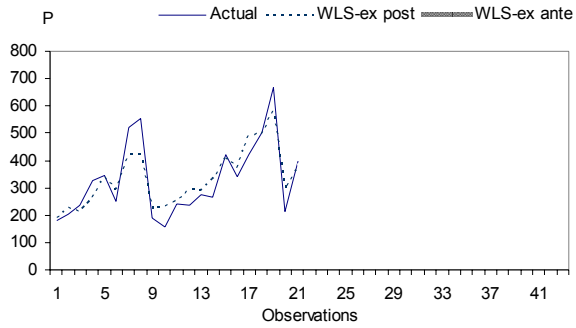
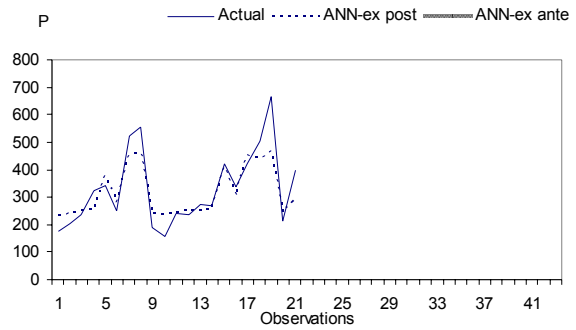


Figure 10. Actual values of median prices, and their ANN ex post and ex ante forecast values



prices utilizing spatiotemporal data. Efficacies of GP and ANN were compared with that of a statistical method: *weighted least squares*. Explanatory variables employed included those that account for the impact of temporal variations in income and mortgage rate, as well as location differences on neighborhood residential housing prices. Such combination of variables prevents use of OLS because some of the temporal variables are highly collinear and location variables are spatially autocorrelated. The homoscedasticity null was rejected when the data was tested. Under these conditions,

the two nonparametric computational techniques GP and ANN seem appropriate to employ in modeling and forecasting prices. These algorithms are robust against many statistical problems. For comparison, WLS specification, applicable under conditions of heteroscedasticity, was also used (instead of OLS) to forecast prices. Over the historical or training period, ANN outperformed the other two techniques. For out-of-sample *ex post* forecast, GP outperformed the other two. GP's superior *ex post* forecast suggests that, among the three, its *ex ante* forecast is possibly the most reliable.

Three ways to account for location variables in models were implemented. The first utilized longitude and latitude (standard in many prior studies) as explanatory variables. The second utilized dummy variables weighted by per capita income—a variable that varied across neighborhoods. The weighted dummy variables helped account for the impact of income change by neighborhood on its median housing prices relative to other neighborhoods. The third utilized relative ranking of neighborhood prices.

When constructing the three models, emphasis was on obtaining predictions useful in decision making. Two-year-ahead forecasts were produced without relying on any forecasted values of explanatory variables. This was possible by setting a lag-length equal to the desired number of periods to predict.

Use of GP, weighted dummy variables, and extended lag-length are all new in modeling and forecasting residential home prices. Further research and experimentation of them is clearly warranted. Linking the type of spatiotemporal model presented here with standard hedonic models also warrants further investigation. While ideas introduced here seem successful, they may not be helpful for areas where clear location definitions of boundaries (similar to Cambridge) are not established or if the data does not exist.

REFERENCES

Anselin, L. (1998). Exploratory spatial data analysis in a geocomputational environment. In M. Fisher, H. Scholten, & D. Unwin (Eds.), *Geocomputation: A primer.* Chichester: John Wiley & Sons.

Anselin, L. (1999). The future of spatial analysis in the social sciences. *Geographic Information Sciences, 5,* 67-76.

Anselin, L., Bera, A., Florax, R., & Yoon, M. (1996). Simple diagnostic tests for spatial dependence. *Regional Science and Urban Economics, 26,* 77-104.

Banzhaf, W., Nordin, P., Keller, R., & Francone, F. (1998). *Genetic programming: An introduction.* San Francisco: Morgan Kaufmann.

Bennell, J., & Sutcliffe, C. (2004). Black–Scholes versus artificial neural networks in pricing FTSE 100 options. *International Journal of Intelligent Systems in Accounting, Finance & Management, 12,* 243-260.

Bin, O. (2004). A prediction comparison of housing sales prices by parametric versus semiparametric regressions. *Journal of Housing Economics, 13,* 68-84.

Cambridge Community Development Department Community Planning Division. (2003). *Cambridge demographic, socioeconomic & real estate market information.* Retrieved from http://www.ci.cambridge.ma.us/ ~CDD/data/index.html

Chellapilla, K. (1997). Evolving computer programs without subtree crossover. *IEEE Transactions on Evolutionary Computation, 1,* 209-216.

Chen, S.-H. (2002). *Genetic algorithms and genetic programming in computational finance.* Dordrecht: Kluwer Academic.

Chen, S., & Smith, S. (1999). Introducing a new advantage of crossover: Commonality-based selection. In W. Banzhaf, J. Daida, A.E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, & R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference*, Orlando, FL (Vol. 1, pp. 13-17). Berlin: Morgan Kaufmann.

Chen, S.-H., & Yeh, C.-H. (2000). Simulating economic transition processes by genetic programming. *Annals of Operations Research, 97,* 265-286.

Clapp, J. (2004). A semiparametric method for estimating local house price indices. *Real Estate Economics, 32,* 127-160.

Clapp, J., Kim, H., & Gelfand, A. (2002). Predicting spatial patterns of house prices using LPR and Bayesian smoothing. *Real Estate Economics, 30,* 505-532.

Clements, M., Franses, P., & Swanson, N. (2004). Forecasting economic and financial time-series with non-linear models. *International Journal of Forecasting, 20,* 169-183.

Dahl, C. M., & Hylleberg, S. (2004). Flexible regression models and relative forecast performance. *International Journal of Forecasting, 20,* 201-217.

Fotheringham, S., Brunsdon, C., & Charlton, M. (2002). *Geographically weighted regression: The analysis of spatially varying relationships*. West Essex, UK: John Wiley & Sons.

Fuchs, M. (1999). Large populations are not always the best choice in genetic programming. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, & R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference* (Vol. 2, pp. 1033-1038). San Francisco: Morgan Kaufmann.

Gathercole, C., & Ross, P. (1997). Small populations over many generations can beat large populations over few generations in genetic programming. In J. Koza, K. Deb, M. Dorigo, D. Fogel, M. Garzon, H. Iba, & R. Riolo (Eds.), *Proceedings of the 2nd Annual Conference on Genetic Programming* (pp. 111-118). San Francisco: Morgan Kaufmann.

Gençay, R., & Yang, X. (1996). A forecast comparison of residential housing prices by parametric versus semiparametric conditional mean estimators. *Economics Letters, 52,* 129-135.

Getis, A., & Ord, K. (1992). The analysis of spatial autocorrelation by use of distance statistics. *Geographical Analysis, 24,* 189-206.

Goldfeld, S., & Quandt, R. (1965). Some tests for homoscedasticity. *Journal of the American Statistical Association, 60,* 539-547.

Goodman, A., & Thibodeau, T. (2003). Housing market segmentation and hedonic prediction accuracy. *Journal of Housing Economics, 12,* 181-201.

Gopal, S., & Fischer, M. (1996). Learning in single hidden layer feed-forward neural network models: Back-propagation in a spatial interaction modeling context. *Geographical Analysis, 28,* 38-55.

Haining, R. (2003). *Spatial data analysis*. Cambridge: Cambridge University Press.

Huang, W., Lai, K., Nakamori, Y., & Wang, S. (2004). Forecasting foreign exchange rates with artificial neural networks: A review. *International Journal of Information Technology & Decision Making, 3,* 145-165.

Kaboudan, M. (2000). Genetic programming prediction of stock prices. *Computational Economics, 16,* 207-236.

Kaboudan, M. (2003). *TSGP: A time series genetic programming software.* Retrieved from http://newton.uor.edu/ facultyfolder/ mahmoud_kaboudan/tsgp

Kaboudan, M., & Liu, Q. (2004). *Forecasting quarterly U.S. demand for natural gas.* Retrieved from http://www.item.woiz.polsl.pl/issue2.1/journal2.1.htm

Koza, J. (1992) *Genetic programming.* Cambridge, MA: The MIT Press.

Lo, A., & MacKinlay, C. (1999). *A non-random walk down Wall Street.* Princeton, NJ: Princeton University Press.

Longley, P., & Batty, M. (1996). *Spatial analysis: Modeling in a GIS environment.* Cambridge, UK: GeoInformation International.

López, C., Álvarez, A., & Hernández-García, E. (2000). Forecasting confined spatiotemporal chaos with genetic algorithms. *Physics Review Letters, 85,* 2300-2303.

Maros-Nikolaus, P., & Martin-González, J. (2002). Spatial forecasting: Detecting determinism from single snapshots. *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering, 12,* 369-376.

Mason, C., & Quigley, J. (1996). Non-parametric hedonic housing prices. *Housing Studies, 11,* 373-385.

Neely, C., & Weller, P. (2001). *Predicting exchange rate volatility: Genetic programming vs. GARCH and RiskMetrics.* Working Paper 2001-009B, Federal Reserve Bank of St. Louis, USA. Retrieved from http://research.stlouisfed.org/wp/2001/2001-009.pdf

NeuroSolutions™. (2002). *The neural network simulation environment. Version 3.* Gainesville, FL: NeuroDimensions.

O'Reilly, U. (1999, April). Foundations of genetic programming: Effective population size, linking and mixing. *Proceedings of the GECCO'99 Workshop.*

Pindyck, R., & Rubinfeld, D. (1998). *Econometric models and economic forecasting.* Boston: Irwin McGraw-Hill.

Principe, J., Euliano, N., & Lefebvre, C. (2000). *Neural and adaptive systems: Fundamentals through simulations.* New York: John Wiley & Sons.

Rossini, P. (2000, January 24-27). Using expert systems and artificial intelligence for real estate forecasting. *Proceedings of the 6th Annual Pacific-Rim Real Estate Society Conference,* Sydney, Australia. Retrieved from http://business2.unisa.edu.au/prres/Proceedings/Proceedings2000/P6A2.pdf

Rubin, D. (1992). Use of forecasting signatures to help distinguish periodicity, randomness, and chaos in ripples and other spatial patterns. *Chaos, 2,* 525-535.

Tsang, E., Yung, P., & Li, J. (2004). EDDIE-Automation, a decision support tool for financial forecasting. *Decision Support Systems, 37,* 559-565.

Chapter LVI
Multiattribute Methodologies in Financial Decision Aid

Mattia Ciprian
University of Trieste, Italy

Massimiliano Kaucic
University of Trieste, Italy

Giulia Nogherotto
University of Trieste, Italy

Valentino Pediroda
University of Trieste, Italy

Danilo DiStefano
AREA Science Park, Italy

ABSTRACT

This chapter introduces the capability of the numerical multi-dimensional approach to solve complex problems in finance. It is well known how, with the growth of computational resource, scientists have developed numerical algorithms for the resolution of complex systems, in order to find the relations between the different components. One important field in this research is focused on the mimic of nature behavior to solve problems. In this chapter two technologies based on these techniques, self-organizing maps and multi-objectives genetic algorithm, have been used to solve two important fields in finance: the country risk assessment and the time series forecasting. The authors, through the examples in the chapter, would like to demonstrate how a multi-dimensional approach based on the mimic of nature could be useful to solve modern complex problems in finance.

INTRODUCTION

The increasing complexity of financial problems over the past few decades has driven analysts to develop and adopt more sophisticated quantitative analysis techniques: furthermore, in the past few years, opinion is growing that the criterion to guide financial decisions has to be multi-dimensional (Zopounidis & Doumpos, 2002a).

This is the reason why mathematical models have replaced the verbal models (Spronk, Steuer,& Zopounidis, 2005). Financial institutions as well as firms acknowledge the multidimensional nature of financial decision problems, and an implication "is that there may be an attempt to redefine the analytical tools used to solve problems; for example, goal programming and multi-criteria decision making," as Bhaskar said as early as 1983 (p. 616). Institutions and firms often use optimization and statistical software packages, even though many of these are not specifically designed for financial decision-making problems. Examples of such programs are the CGX system, the BANKS system, the BANKADVISER system, the INVEX system, the FINEVA system, the FINCLAS system, and the INVESTOR system (Spronk et al., 2005).

The use of mathematics and an operational research approach in finance got its start in the 1950s with the introduction of Markowitz's portfolio theory. Since then many researchers and practitioners gave their contribution. With our work we would illustrate how a multidimensional approach could be useful to solve different financial problems.

In this chapter the attention has been focused on two particular aspects of financial decision making: the country risk assessment and an optimized statistical approach to modeling time series.

The country risk field includes a wide range of different problems and aspects: this is the reason for which many different definitions have been formulated (Bouchet, Clark, & Groslambert, 2003). We will consider it as a additional risk that is involved when business transactions occur across international borders. The assessment of country risk will be realized as a classification problem employing a multi-criteria decision-making (MCDM) approach; further, a self-organizing map (SOM) approach will be used to discover clusters of countries with similar risk patterns and understand the produced rating.

In the second part of the chapter we review the best-known statistical methodology for the time series modeling and forecasting: the ordinary least squares (OLS) model. This model, even though obsolete, is still used by many traders and asset management practitioners. Through an operational research approach and using a multi-objective genetic algorithm (MOGA), we will discuss the regression method, fixing attention on statistical hypothesis testing. It is a well-known fact that, to check the validity of the regression, there are a lot of tests (p-values for t-tests, F-test for testing the overall significance of the model, etc.) which must be satisfied or rejected: when the number of variables is high, this objective could be a very hard task. Consequently, the aim of our work is to make the investigation of feasible solutions an automatic process and to choose the best one.

COUNTRY RISK MULTI-DIMENSIONAL ASSESSMENT

As explained by Meldrum (2000), when business transactions occur across international borders, they carry additional risks not present in domestic transactions. These additional risks,

Table 1. Factors included on "DB1"

Ratio 1	Coface rate 2004
Ratio 2	Birth rate
Ratio 3	Death rate
Ratio 4	Export (% variation)
Ratio 5	Import (% variation)
Ratio 6	GDP (no PPP)
Ratio 7	GDP (PPP)
Ratio 8	Inflation rate
Ratio 9	GDP growth rate
Ratio 10	Public balance/GDP
Ratio 11	Net migration rate
Ratio 12	Infantility mortality rate
Ratio 13	Life expectancy at birth
Ratio 14	Labor force/tot. pop.
Ratio 15	Internet users
Ratio 16	Production growth rate
Ratio 17	Electricity consumption
Ratio 18	Electricity production
Ratio 19	Fertility rate
Ratio 20	Export/GDP
Ratio 21	Import/GDP

Table 2. Ratios used for the country risk prediction of developing countries "DB2"

Ratio 1	Default Standard & Poor's evaluation
Ratio 2	Default S&P's one year before
Ratio 3	Gross domestic product (GDP)
Ratio 4	GDP growth rate
Ratio 5	Rate of inflation
Ratio 6	Exchange rate in purchasing of power parity (PPP)
Ratio 7	Average interest rate
Ratio 8	Average maturity years
Ratio 9	Total U.S. dollar amount of merchandise exports
Ratio 10	Foreign direct investment
Ratio 11	Total U.S. dollar amount of merchandise imports
Ratio 12	Total interest payment
Ratio 13	International reserve
Ratio 14	Total external debt
Ratio 15	Short-term external debt
Ratio 16	Interest on short-term external debt
Ratio 17	Total debt service
Ratio 18	Long-term debt service
Ratio 19	Current account balance

called country risks, typically include risks arising from a variety of national differences in economic structures, policies, socio-political institutions, geography, and currencies. Moreover, if one considers that the liberalization of economies facilitates and encourages goods exchange between companies and that exchanges are up to 70% short-term payable, one can realize how important it is to strictly define and clearly state the country risk.

In our work we propose a method based on the contextual use of SOM and MCDM algorithms that allows us to study these differences and compare them. Our aim is to investigate the dynamics of global economy to gain insight into exchange policies between different countries.

Data

Our study, outlined in the previous section, has been applied to country risk data obtained from many organizations such as World Bank, CIA, and Coface. On the basis of data availability for the more recent period (2004), 52 countries were selected for a variety of factors (21) involving economic, social, demographic, and environment indicators (CIA, 2004; World Bank Group, 2004) (henceforth we will refer to this database as to "DB1"). The nations were selected as follows:

- 22 countries *Europe and CSI* area;
- 8 countries from *America*;
- 9 countries from *Asia and Oceania*;
- 8 countries from *North Africa* and *Middle East*;
- 9 countries from *Sub-Saharan Africa*.

Another database has been organized using data obtained from ISAE (Italian Institute for Studies and Economic Analysis) that provides information about 27 developing countries throughout the period 1983-2000. Henceforth we will call it "DB2."

Figure 1. Energy consumption

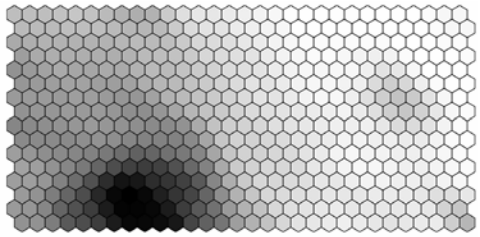


Figure 2. Energy production

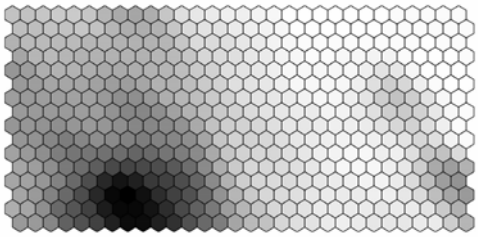


Figure 3. GDP growth rate

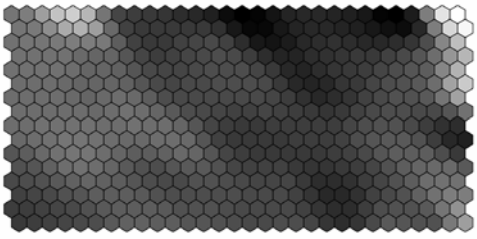
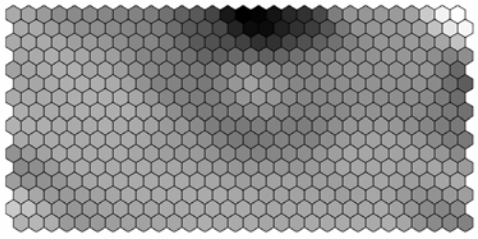


Figure 4. Industrial production growth rate



Self-Organizing Maps

By means of a non-linear ordered regression, self organizing maps provide a topology preserving mapping from the high dimensional space of data to map units, which usually forms a two-dimensional lattice. Thus, SOMs are an efficient way of visualization for highly multidimensional and highly complex datasets. This mapping typology guarantees that nearby points in the space of data are mapped to nearby points in the map (Kohonen, 1998; Kohonen & Somervuo, 2001).

SOMs are based on a vectorial regression on data samples, performed by a set of weight vectors, each one assigned to a map unit, its components being iteratively changed in such a way that they resemble the original data records as best as possible, leading to an ordered map. Choosing a proper map visualization, one can perform two kinds of tasks: inspection of local variable correlations and cluster analysis. (Ultsch, 2003).

Results and Their Interpretation

We have considered performing an analysis of the indicators and general information about the countries because we think that the very complex equilibrium of a country system could be understood only by a careful consideration of these aspects.

Since the 1960s, researchers have tried to build macroeconomic models starting from the comprehension of the assets interactions; but Krugman in 1979 even noted the need for 'not simplified' macroeconomics models that embed a more effective consideration of asset interactions.

We choose to conduct two kinds of analysis on "DB1":

1. **Factor study and individuation of local correlations in order to understand the logics joining the economics and social aspects of countries:** We trained a rectangular map with 20x10 hexagonal nodes during 15000 cycles;

2. **A clustering analysis based on credit risks:** To produce a visual representation and verify the World Bank's classification according to the income level.

Factor Study

We have obtained, as foreseen by us, some obvious relations as between "energy consumption" and "energy production" (Figure 1 and Figure 2)—the correlation describing the degree of relationship between the two variables is 0,963.

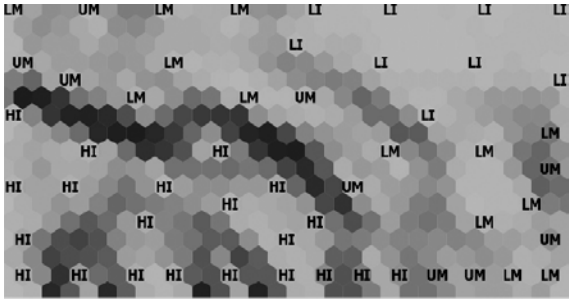
After all, SOMs are very powerful in showing, in certain areas, the non-linear dependencies as in Figure 3 and Figure 4: it appears clearly how local minima and local maxima are displayed in the same nodes, meaning that remarkable variations (positive and negative) of industrial production growth rate heavily influence GDP growth rate. A quite predictable effect, it would not be appreciated using a classical statistic approach. The statistical correlation between the factors is very low: 0,347.

Cluster Analysis

On the basis of these results, it was decided to proceed with the second phase of the analysis: excluding from "DB1" some indicators that had high correlations with others, a reduced set of 14 indicators has been employed. We trained a rectangular map with 20x10 hexagonal nodes through 100,000 learning cycles, and we represented the *U-Matrix*[1] of the weight vectors. Since a model of some observation is associated with each node, it is possible to find the node that matches best with each entity of the observation space and therefore to associate a label to each node.

Following the definition of the groups based on the World Bank's classification of the countries according to the income level (that is: **HI** high income level, **UM** upper middle level, **LM** lower middle level, and **LI** low income level), we produce the map in Figure 5.

Figure 5. U-Matrix with labels associated according to the income level; light grey clusters are separated by dark grey peaks



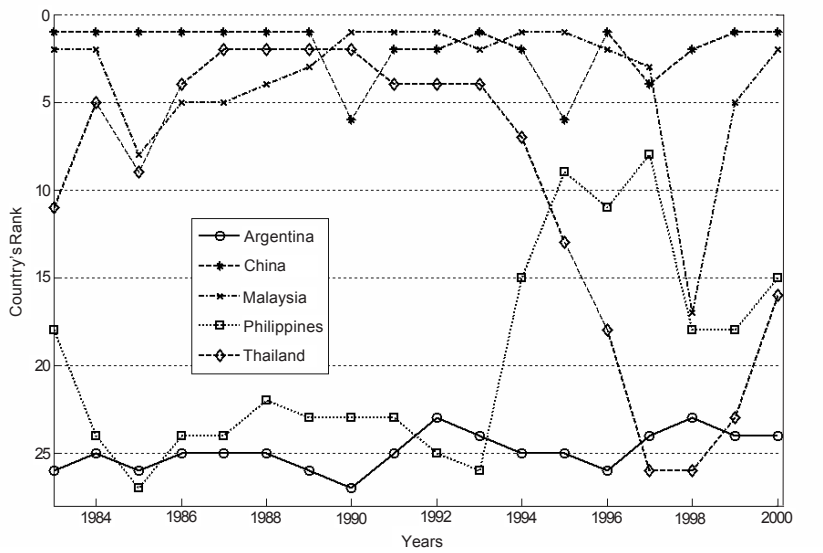
It is worth noting that HI and LI economies are easier to identify than intermediate cases such as LM and UM income economies, as reported in other works (Zopounidis & Doumpos, 2002b; Doumpos & Zopounidis, 2000).

MCDM Approach

Risk assessments are usually expressed as classification of countries into predefined groups according to their performance and level of risk; otherwise they are examined as ranking of the countries from the best to the worst ones according to an appropriate definition of country risk. In our study we will follow the second approach.

We adopted a MCDM algorithm proposed by Sen and Yang (1998, pp. 75-97) called CODASID. This method attempts to generate a clear preference order for alternative designs through a concordance analysis (extending the original analysis of ELECTRE; Roy, 1996) and a discordance analysis by similarity to ideal designs, deriving from the TOPSIS method

Figure 6. Rank's evolution of some developing countries during the period 1983-2000



(Hwang & Yoon, 1981) the basic idea of defining such a distance measure for each feasible design from a given reference design. The basic concept is that the best action should have the shortest distance from an ideal design and the greatest from a negative ideal design. The inputs required by CODASID are a matrix ("Preference Matrix") containing the objects to be explored during the decision procedure, and a vector of weights expressing the relative importance of one attribute with respect to the others.

We performed two different studies:

1. Using "DB2" as "Preference Matrix" and assigning directly the weights to each attribute according to their importance, we could observe the evolution of each developing country during the considered period (1983-2000).
2. Using "DB1" as "Preference Matrix" and the "Entropy Method" to fix the weights, we ranked the countries and compared the results with the World Bank's classification according to the income level.

Evolution of Developing Countries

DB2 contains data on the following developing countries: Argentina, Bolivia, Brazil, Chile, China, Colombia, Costa Rica, Dominican Republic, Ecuador, El Salvador, Guatemala, Hungary, India, Indonesia, Jamaica, Jordan, Malaysia, Mexico, Morocco, Panama, Paraguay, The Philippines, Romania, Thailand, Tunisia, Turkey, and Uruguay.

To understand the capability of our methodology to predict the behavior of the developing countries, we need to compare the ranking evolution by CODASID with the historical events. The ratios used for the countries are summarizing in Table 1, the time series are from 1983 to 2000. The weights for the *multi-criteria decision making* are imposed by financial analyst choice.

From Figure 6, it is possible to note how our methodology recognizes the important fallouts historically occurring in a country; in fact it is possible to visualize a sudden variation in the country rank. From the visualization, we could extract some examples to understand the capa-

Table 3. Ranking obtained from CODASID and compared with the World Bank's classification

Rank	Country	WB's Income Level	Rank	Country	WB's Income Level
1	Singapore	HI	27	Thailand	LM
2	Sweden	HI	28	Poland	UM
3	Canada	HI	29	China	LM
4	Hong Kong	HI	30	Mexico	UM
5	Un. Arab Emirates	HI	31	Tunisia	LM
6	Finland	HI	32	Paraguay	LM
7	United States	HI	33	Argentina	UM
8	Switzerland	HI	34	Bulgaria	LM
9	Ireland	HI	35	Russian Federation	LM
10	Denmark	HI	36	Brazil	LM
11	Belgium	HI	37	Morocco	LM
12	Australia	HI	38	Uruguay	UM
13	Austria	HI	39	Turkey	LM
14	Japan	HI	40	South Africa	LM
15	United Kingdom	HI	41	Egypt	LM
16	Germany	HI	42	Algeria	LM
17	France	HI	43	Madagascar	LI
18	Malaysia	UM	44	India	LI
19	Spain	HI	45	Albania	LM
20	Italy	HI	46	Sudan	LI
21	Portugal	HI	47	Cambodia	LI
22	Saudi Arabia	UM	48	Cote d'Ivoire	LI
23	Slovak Republic	UM	49	Ghana	LI
24	Greece	HI	50	Nigeria	LI
25	Hungary	UM	51	Mozambique	LI
26	Chile	UM	52	Zimbabwe	LI

bility of the methodology. Regarding the time series of the China ranking, it is possible to observe two important events: the Tiananmen massacre in the 1989 (ranking from 1 to 6), and some ranking fluctuations during 1994-1995, mainly correlated with the political repression in Tibet. Another important event that it is possible to observe in the ranking evolution is the Asian economic crisis during 1997-1998; in these two years it is possible to observe in the Figure 6 that the Asian countries (China, Malaysia, Thailand, The Philippines) have a ranking degrading that is visible especially for Malaysia (from 3rd to 16th) and The Philippines (from 8th to 17th). Malaysia quickly improves its ranking after the crisis, probably related to the industrial situation, mainly oil extraction. Also interesting is Argentina's ranking before the 2001 crisis: our model shows how the position of Argentina is not very encouraging (the average rank is 25) in contrast with the idea that Argentina was an economical model for the Latin America countries before 2001.

Risk Comparison

To test the capability of our numerical methodology with other data, we compared the ranking by CODASID with the Word Bank classification. The used data is summarizing in Table 1, where it is possible to note how the ratios cover many economical-social-financial aspects; the total number of countries examined is 52, uniformly distributed around the world. For the weight assignment in the CODASID method, we used the Entropy technique that is particularly efficient to investigate contrasts in discrimination between sets of data (Hwang & Yoon, 1981; Seen & Yang, 1998).

In Table 3 we report the obtained result compared with the World Bank classification; it is possible to note how the ranking by our methodology covers with good accuracy the Word Bank classification, especially for the best countries ("high income") and the worst ("low income"). A small fluctuation is present for the intermediate cases, where it is a mixture between the "upper middle" and "lower middle" countries. This result valorizes the previous use of the SOM, where we found similar relations in the countries' ranking.

OPTIMIZED OLS MODEL

Many traders and asset management practitioners use the traditional ordinary least square (OLS) regression to estimate stock price or indices. This approach has two limits:

1. Even the most expert econometrician experiences difficulties in modeling economic data when the precise formulation of the equation under analysis is unknown a priori.
2. The method is very time expensive since a good model (that is able to predict coming situations) includes a large set of tests (Godfrey, 2005) such as:
   - tests for misspecification of regression mean function and on estimated coefficients;
   - tests for serial correlation and unconditional heteroscedasticity;
   - tests for structural change; and
   - tests for non-normality of the residual distribution.

New directions of research are given by the automatic modeling methods (Hendry & Krolzig, 2001, 2004, 2005), such as *PcGive* and RETINA (RElevant Transformation of the Inputs Network Approach) (Gallo, Perez-Amaral, & White, 2003, 2004; Maindonald, 2002; Faraway, 2002). Their purpose is to give procedures which could be able to detect models in a small amount of time, and also when the researcher has no information or is non-expert.

In our work we propose a new approach based on the OLS regression coupled with a multi-objective genetic algorithm (MOGA) and a multi-criteria decision making (MCDM) method.

Statistical Assumptions and Diagnostic Tests

The regression model to be estimated and subject to diagnostic tests using a sample of size T can be written as

$$y_t = \mathbf{X_t}^{\mathrm{T}} \beta + \varepsilon_t$$

where $\mathbf{X_t}$ is the vector containing observations on exogenous regressors and ε_t represents the error at time t. In particular, it is assumed that the errors ε_t are independently and identically distributed (i.i.d.) with zero mean and variance, with normal cumulative density function.

Table 4. Scheme of diagnostic tests adopted by authors

Global test	
Adjusted R-squared	Close to 1
F-statistic	High value
Specific Test (for each var)	
p-value	Value < 0; 05
Information criterion	
Schwarz criterion	Low value
Homoskedasticity of residuals	
Arch LM Test	High probability
White Heteroskedasticity Test	High probability
Serial correlation of residuals	
Durbin-Watson Stat	Close to 2
Breusch Godfrey Test LM	High probability
Ljung Box Test	High probability
Correlogram - Q statistics	High probability
Normality of residuals	
Jarque Bera Test	High probability
Skewness	Close to 0
Kurtosis	Close to 3
Stability of coefficients	
Chow breakpoint Test	High probability

In order to monitor the statistical model, we used a set of tests to verify the previous properties. In Table 4 we indicate the category of tests we have used and the conditions imposed in order to specify the model correctly.

The variable that is studied through a multiple regression model is represented by the price of the *MSCI AC Europe Index,* which is an industry index used as a benchmark in the European Energy sector.

We choose a set of macro and microeconomic variables to implement our model. At the first step of the analysis, the set was comprehensive of 33 variables. About the macroeconomic variables we considered the information extracted from a study recently implemented by Generali Asset Management (2003). They are classified as follows:

- four variables describing interest rates;
- the price of brent;

- an exchange rate;
- four European and American price markers;
- a set of nine variables relating to global production and particularly to consumption, investments, and balance trade (both EU and U.S.); and
- two variables related to the U.S. labor market.

For the microeconomic predictors we took into consideration a set of variables whose historical series were available on the financial market relative to the Industry Sector Index that we are analyzing; they are classified as follows: four earnings per share, one dividend yield, four kinds of price/earnings, and two kinds of volatility.

The considered historical series contained the data observed over the period of time from February 28, 1995, to December 28, 2003. We have considered monthly observations.

Multi-Objective Optimization

First of all, it is important to understand which kind of variables have been adopted: to each time series we have associated a discrete variable named L that represents the time lag. L could assume every value from zero to six, where zero indicates that the variable is not present in the regression. Every value greater than zero produces a lagged (i.e., shifted) version of the time-series matrix.

Observing Table 4, our problem can be expressed clearly as follows:

$$\begin{cases} \max F_i(\bar{x}) & for\ i = 1, n \\ g_j(\bar{x}) \le 0 & for\ j = 1, m \end{cases}$$

These equations describe a multi-objective problem where, in general, the solution is not unique if the functions are not linearly dependent (as explained in Poloni & Pediroda, 1997).

In our case, the objectives are represented by the maximization (or minimization) of almost all tests listed in Table 4 and only one constraint is fixed: every p-value < 0,05. Since this task is obviously very hard to realize, it is necessary to explore the entire solution space in order to better understand the problem and to redefine the number of objectives needed to obtain the optimal solution with the lowest number of computations required: we have decided to adopt a technique based on the DOE (design of experiment) statistic methodology (Montgomery, 1996; Del Vecchio, 1997; Clarich, Mosetti, Pediroda, & Poloni, 2002). Thus the optimization has been run in three phases: in the first one we have analyzed 49 different configurations on the basis of the "Latin Square" DOE methodology (Weisstein, 2005); in the second phase, through a multi-objective genetic algorithm, we have found the Pareto frontier of the *not dominated* optimal solutions; in the last phase we have chosen from the Pareto frontier a particular configuration through a MCDM algorithm.

First Step: Statistical Analysis

Examining any process, the simplest strategy of experiments would be to use a "full factorial" DOE—that is, taking into account every combination of *control variables* (independent variables) and *levels* (different variable settings). If N is the number of control variables and L_i the number of levels of factor i, the total number of experimental is $\prod_{i=1}^{N} L_i$; in our case it would require more than 7.0e+27 computations. In order to reduce this number, we have used a the *Graeco-Latin Square,* an efficient algorithm derived by the Latin Squares (we used a particular algorithm implemented in modeFRONTIER v.3.1.1 (build 2005.05.09 HJS): the configuration with the minimum vari-

Table 5. Correlation matrix of the objectives: Adjusted R^2, F statistic, Arch LM test probability, LM Breusch Godfrey probability, Ljung-Box probability, White Heteroskedasticity probability, Jarque Bera probability, Chow breakpoint probability, Skewness to 0, Kurtosis to 3, Durbin-Watson to 2, Schwarz criterion value

	AdjR²	F stat	p-ARCH	p-BG	p-LB	p-W	p-JB	p-C	skew	kurto	DW	SC
AdjR²	1	0.9	-0.1	0.4	0.6	-0.3	0.1	0.5	-0.1	-0	-0.7	-0.9
F stat	0.9	1	-0.1	0.3	0.5	-0.3	0.1	0.5	-0.1	-0	-0.5	-1
p-ARCH	-0.1	-0.1	1	-0.1	0	-0	-0.1	-0.1	0.1	0.1	0.1	0.1
p-BG	0.4	0.3	-0.1	1	0.8	-0.2	0.1	0.1	-0.1	-0	-0.7	-0.3
p-LB	0.6	0.5	0	0.8	1	-0.2	0.1	0.2	-0.2	-0	-0.8	-0.5
p-W	-0.3	-0.3	-0	-0.2	-0.2	1	-0.2	-0.1	0.2	0.2	0.4	0.3
p-JB	0.1	0.1	-0.1	0.1	0.1	-0.2	1	0.2	-0.6	-0.8	-0.2	-0.1
p-C	0.5	0.5	-0.1	0.1	0.2	-0.1	0.2	1	-0.1	-0.1	-0.3	-0.5
skew	-0.1	-0.1	0.1	-0.1	-0.2	0.2	-0.6	-0.1	1	0.3	0.1	0.1
kurto	-0	-0	0.1	-0	-0	0.2	-0.8	-0.1	0.3	1	0.2	0.1
DW	-0.7	-0.5	0.1	-0.7	-0.8	0.4	-0.2	-0.3	0.1	0.2	1	0.5
SC	-0.9	-1	0.1	-0.3	-0.5	0.3	-0.1	-0.5	0.1	0.1	0.5	1

ables correlation is chosen). This algorithm requires only the definition of the number of levels ($L_i = 7$), and the number of generated designs is $L_i^2 = 49$; by both definition and implementation, a Graeco-Latin Square represents a balanced design.

Analyzing the solutions space, it is possible to understand if any relation exists between the objectives; as reported in Table 5, many objectives are quite correlated: this result allows the reduction of their number from 12 to 6 (eliminating DW, SC, kurto, skew, p-LB, and Adj R^2); more precisely:

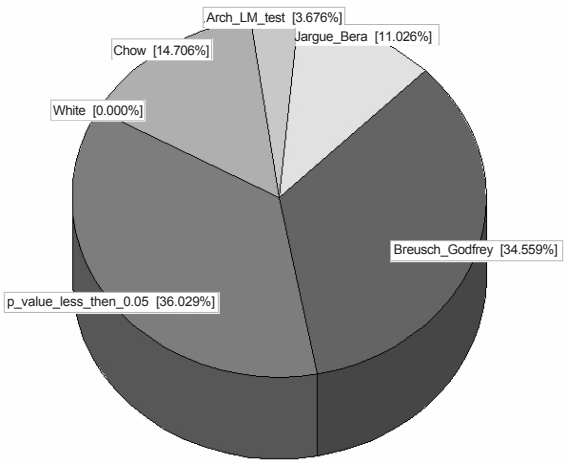
1. Maximizing *F statistic*, we obtain the contextual maximization of *Adjusted* R^2 and the minimization of *Schwarz criterion value.*
2. Maximizing *Breusch Godfrey probability*, we obtain an high *Ljung Box probability* and a minimization of |*Durbin Watson* - 2|.
3. Maximizing *Jarque Bera probability*, we obtain a reduction of |*skewness*| and |*kurtosis* - 3|.

Among the objectives, there are some probability estimations that can be used to state the truth of the null hypothesis, comparing the p-value to an acceptable significance value α; this particular feature allows us to impose some constraints instead of objectives. Creating a Broken Constraints pie chart, which shows the influence of the different constraints on the unfeasible designs, we obtained as indication (see Figure 7) that the following constraints are hardly satisfied simultaneously:

1. p-value < 0.05 for each variable;
2. Breusch Godfrey p-value > α.

To achieve every goal and satisfy every constraint, we decided to drive the optimization as follows:

Figure 7. Broken Constraints pie chart



- Objectives:
  1. minimize the number of times the p-value is greater then 0.05 in each regression;
  2. maximize F statistic;
  3. maximize Breusch Godfrey probability;
  4. maximize Jarque Bera probability;
- Constraints:
  1. Arch LM test probability > *a*;
  2. White Heteroskedasticity probability > *a*
  3. Chow breakpoint probability > *a*;
  4. Breusch Godfrey probability > *a*;
  5. Jarque Bera probability > *a*;
  6. Ljung Box probability > *a*

with $\alpha = 0.30$ (for each autocorrelation test, a number of lags equal to the maximum value of L_i have been adopted).

Second Step: MOGA

In order to solve the multi-objective problem, we have used a multi-objective genetic algorithm (Deb, 1999) that uses a smart multi-search elitism able to preserve some excellent solutions without bringing premature convergence into local optimal fronts (Poles, 2003; ESTECO, 2005). The initial population (350 entries) has been created using the Sobol method.

After about 20 generations, the optimization has been deemed concluded; Figures 8-11 show plots of the moving average of objectives values as a function of the Design ID (which identifies the sequence of design generated). It appears clearly that each goal has evolved simultaneously towards better values.

The total time required for the whole optimization was about 2.5 hours using a single PC

Figure 8. Number of times the p-value constraint was broken

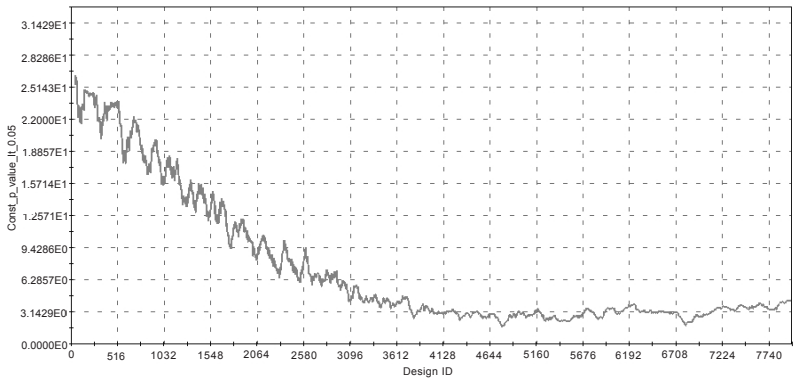


Figure 9. Jarque Bera probability

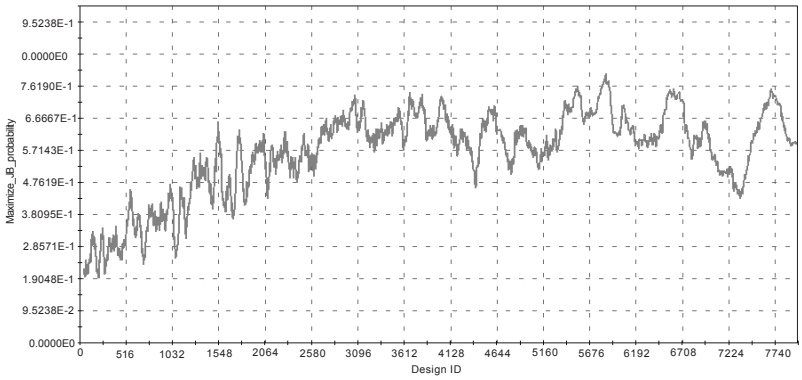


Figure 10. F statistic

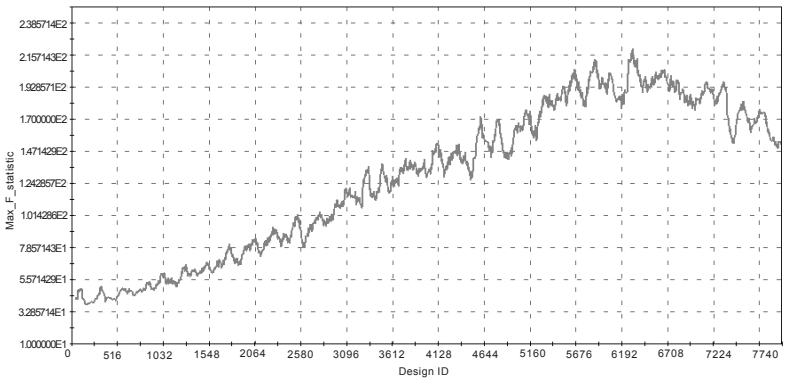


Figure 11. Breusch Godfrey probability

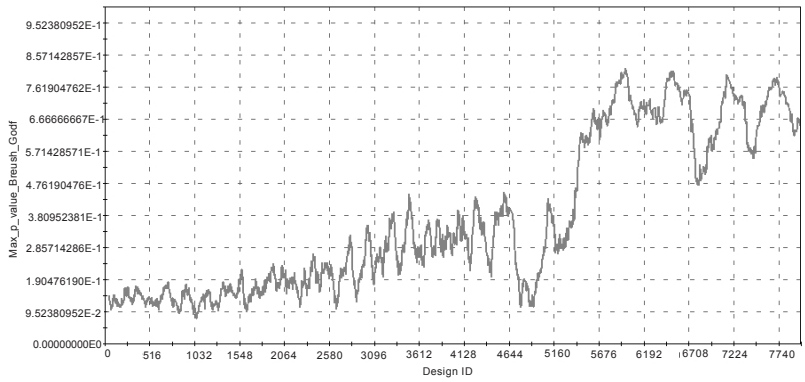


Table 6. Objectives values of the first five configurations chosen from the Pareto front

Adj R^2	DW	SC	F stat	p-ARCH	p-BG	p-LB	p-W	p-JB	p-C	skew	kurto
0,93	1,98	6,29	260,69	0,59	0,74	0,73	0,30	0,68	0,58	0,03	3,44
0,93	1,98	6,35	234,04	0,50	0,69	0,67	0,34	0,66	0,54	0,07	3,44
0,93	2,05	6,31	244,68	0,47	0,66	0,66	0,32	0,56	0,59	0,10	3,51
0,93	1,76	6,40	232,91	0,92	0,60	0,61	0,35	0,52	0,66	0,01	3,58
0,92	1,95	6,46	181,72	0,79	0,91	0,93	0,36	0,37	0,50	0,24	3,52

CPU (1.4 GHz, 2 GB Ram)—an important saving of time compared to the traditional manual procedure.

Third Step: MCDM

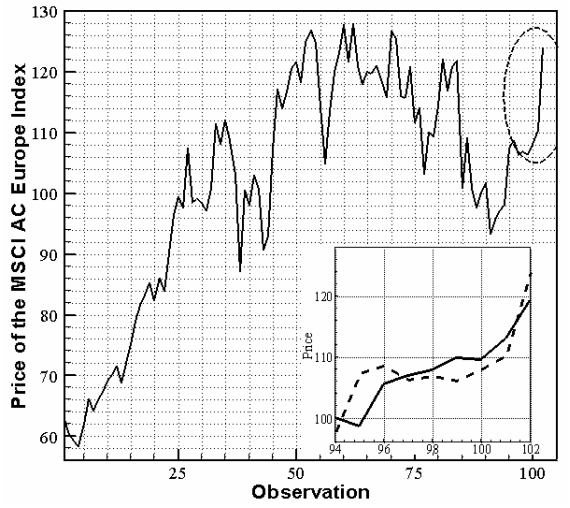
After the optimization we needed a tool that would help to choose one particular configuration from the Pareto-optimal set. We adopted a MCDM algorithm proposed by Yang and Sen (1998): CODASID. This method has been briefly described in a previous section. The inputs required by CODASID are:

- a decision matrix which contains information for multi-attribute evaluations: we used the entire set of values of configurations chosen in the Pareto front;
- a vector of weights expressing the relative importance of one attribute over others: we used a vector of ones.

In Table 6 we report the objective values of the first five configurations chosen from the Pareto front; the ranking is based on the magnitude of "goodness" of every solution.

Having chosen the best alternative, we have obtained the regression and tested the solution eight months later than the last data used, as shown in Figure 12. The short and long trend of the market are clearly identified. Any inaccuracy could be due only to the OLS method (whose limits are known).

Figure 12. Regression obtained choosing the best configuration; comparison between the profile of the original data (dotted) and the forecasted (solid line)



CONCLUSION

In this chapter we have discussed two particular aspects of financial decision making: the country risk assessment and an optimized statistical approach to model the time series. We proposed some useful tools, developed and frequently used in modern engineering contexts, to study this problem: multi-objective genetic algorithm, self-organizing maps, and multi-criteria decision making. We could derive the following conclusions:

- As Meldrum (2000) said in a recent work, "A company needs to examine the relationship between risk and its businesses to make sure risk measures actually help the

company improve its business decisions"; an additional risk is represented by *country risk*. In this chapter we show how SOM and MCDM could be used as tools to analyze this risk: both proved efficient in handling many data together and producing a consistent classification of countries according to their risk level. The results have been compared with other recent works.

- The OLS regression is still considered, thanks to its simplicity and accuracy, the best method to model time series. Every regression is characterized by many tests, probability values, and descriptive measures; they have to be satisfied or have adequate values: reaching this objective could be hard and time expensive. We proposed an operational research approach, coupling statistical notions to an multi-objective genetic algorithm, able to study, search, and find the best regression model. This approach appears to be practical and profitable.

ACKNOWLEDGMENTS

This work has been partially financed by MIUR (Italian Ministry of Education, University and Research) under the research project "Complex Systems in Economics" (decree 1259/Ric at June 30, 2003). We would like to thank ESTECO for the use of *modeFRONTIER*. We wish to express our thanks to Professor Carlo Poloni and Professor Maurizio Fanni for their contributions and constructive comments.

NOTE

U-matrix reports on each unit its mean distance from its six neighboring units; low values indicate units that are very similar to their neighborhood, forming an emerging cluster structure (Ultsch, 2003).

REFERENCES

Bouchet, M. H., Clark, E., & Groslambert, B. (2003). *Country risk assessment.* Chichester, UK: Wiley Finance.

Bhaskar, K., & McNamee, P. (1983). Multiple objectives in accounting and finance. *Journal of Business Finance & Accounting, 10*(4), 595-621.

Central Intelligence Agency. (2004). *The world factbook 2004*. Retrieved from http://www.cia.gov/cia/publications/factbook/

Clarich, A., Mosetti, G., Pediroda, V., & Poloni, C. (2002, February). Application of evolutive algorithms and statistical analysis in the numerical optimization of an axial compressor. *Proceedings of the 9th of International Symposium on Transport Phenomena and Dynamics of Rotating Machinery,* Honolulu, Hawaii.

Deb, K. (1999). Multi-objective genetic algorithm: Problems and construction of test problem. *Evolutionary Computer Journal, 7*(3), 205-230.

Del Vecchio, R. J. (1997). *Design of experiments: A primer for technologists.* Munich: Hanser/Gardner.

Doumpos, M., & Zopounidis, C. (2000). Assessing financial risks using a multicriteria sorting procedure: The case of country risk assessment. *OMEGA, 29*(1), 97-109.

ESTECO. (2005). *modeFRONTIER user manual.* Trieste, Italy.

Faraway, J.J. (2002, July). *Practical regression and Anova using R*. Retrieved Novem-

ber 10, 2005, from http://www.stat.lsa.umich.edu/~faraway/book/pra.pdf

Gallo, G. M., Perez-Amaral, T., & White, H. (2003). A flexible tool for model building: The relevant transformation of the inputs network approach (retina). *Oxford Bulletin of Economics and Statistics, 65,* 92-102.

Gallo, G. M., Perez-Amaral, T., & White, H. (2004). A comparison of complementary automatic modeling methods: Retina and pcgets. *Econometric Theory, 20*.

Godfrey, L. G. (2005). Controlling the overall significance level of a battery of least squares diagnostics tests. *Oxford Bulletin of Economics and Statistics, 67,* 263-279.

Hendry, D. F., & Krolzig, H. M. (2001). Computer automation of general to specific model selection procedures. *Journal of Economic Dynamics and Control, 25,* 831-866.

Hendry, D. F., & Krolzig, H. M. (2004). We ran one regression. *Oxford Bulletin of Economics and Statistics, 66*(5), 799-810.

Hendry, D. F., & Krolzig, H. M. (2005). The properties of automatic Gets modeling. *The Economic Journal, 115,* C31-C61.

Hwang, C. L., & Yoon, K. (1981). *Multiple attribute decision making: Methods and applications.* Berlin: Springer-Verlag.

Kohonen, T., & Somervuo, P. (2001). *The self-organizing map* (3rd ed.). Heidelberg: Springer.

Kohonen, T. (1998). The self-organizing map. *Neurocomputing, 21,* 1-6.

Krugman, P. (1979). A model of balance-of-payments crises. *Journal of Money, Credit and Banking, 11*(3), 311-325.

Maindonald, J. H. (2004, October). *Using R for data analysis and graphics. Introduction, code and commentary.* Retrieved November 10, 2005, from http://cran.r-project.org/doc/contrib/usingR.pdf

Markowitz, H. (1950). Theories of uncertainty and financial behavior. *Econometrica*, (19), 325-326.

Meldrum, D. H. (2000). Country risk and foreign direct investment. *Business Economics, 35*(1), 33.

Montgomery, D. C. (1996). *Design and analysis of experiments* (4th ed.). New York: John Wiley & Sons.

Poles, S. (2003, December). *An improved multi-objective genetic algorithm*. ESTECO Technical Report 2003-006, Italy.

Poloni, C., & Pediroda, V. (1997). GA coupled with computationally expensive simulations: Tools to improve efficiency. In D. Quagliarella, J. Periaux, C. Poloni, & G. Winter (Eds.), *Genetic algorithm and evolution strategies in engineering and computer science* (pp. 267-288). New York: John Wiley & Sons.

Roy, B. (1996). *Multicriteria methodology for decision aiding.* Dordrecht: Kluwer Academic.

Sen, P., & Yang, J. B. (1998). *Multiple criteria decision support in engineering design.* Springer.

Spronk, J., Steuer, R. E., & Zopounidis, C. (2005). Multicriteria decision aid/analysis in finance. In J. Figueira, S. Greco, & M. Ehrgott (Eds.), *Multiple criteria decision analysis: State-of-the-art surveys* (pp. 799-858). Boston/Dordrecht/London: Springer-Verlag.

World Bank Group. (2004). *WDI data query.* Retrieved November 10, 2005, from http://devdata.worldbank.org/data-query/

Ultsch, A. (2003). Maps for the visualization of high-dimensional data spaces. *Proceedings of the Conference on WSOM*, Japan, Kyushu (pp. 225-230).

Weisstein, E. W. (2005). *Latin square.* Retrieved November 10, 2005, from http://www.mathworld.com

Zopounidis, C., & Doumpos, M. (2002a). Multicriteria decision aid in financial decision making: Methodologies and literature review. *Journal of Multi-Criteria Decision Analysis, 11,* 167-186.

Zopounidis, C., & Doumpos, M. (2002b). On the use of a multi-criteria hierarchical discrimination approach for country risk assessment. *Journal of Multi-Criteria Decision Analysis, 11,* 279-289.

Chapter LVII
Multi-Objective Optimization Evolutionary Algorithms in Insurance-Linked Derivatives

M. J. Pérez
Universidad Carlos III de Madrid, Spain

J. García
Universidad Carlos III de Madrid, Spain

L. Martí
Universidad Carlos III de Madrid, Spain

J. M. Molina
Universidad Carlos III de Madrid, Spain

ABSTRACT

This work addresses a real-world adjustment of economic models where the application of robust and global optimization techniques is required. The problem dealt with is the search for a set of parameters to calculate the reported claim amount. Several functions are proposed to obtain the reported claim amount, and a multi-objective optimization procedure is used to obtain parameters using real data and to decide the best function to approximate the reported claim amount. Using this function, insurance companies negotiate the underlying contract—that is, the catastrophic loss ratio defined from the total reported claim amount. They are associated with catastrophes that occurred during the loss period and declared until the development period expired. The suitability of different techniques coming from evolutionary computation (EC) to solve this problem is explored, contrasting the performance achieved with recent proposals of multi-objective evolutionary algorithms (MOEAs). Results show the advantages of MOEAs in the proposal in terms of effectiveness and completeness in searching for solutions, compared with particular solutions of classical EC approaches (using an aggregation operator) in problems with real data.

INTRODUCTION

The search for models explaining a database representing different real situations is common in many practical applications. Usually, real data represent opposite behaviors so that the acceptable model should achieve a certain level of performance with each situation. This problem can be generally transformed into a multi-objective optimization, attaining an appropriate trade-off to cover all cases. This field has been extensively studied in past years by evolutionary computation (ECs) researchers (Deb, 2001), defining the properties and advantages for different families of algorithms. Two basic types of approaches are distinguished: on the one hand the aggregation of all objectives into a single scalar function to apply standard optimization methods, and on the other hand the use of specific multi-objective evolutionary algorithms (MOEAs), exploiting the concept of multi-objective domination. In this second case, the algorithms preserving dominance are intended to maintain diverse population and provide a representation of the whole Pareto-optimal front after a single optimization run. In this chapter we propose the application of these techniques to search for the most appropriate model for the underlying loss ratio of catastrophic contract of capital markets, considering at the same time different cases covered in the available data set.

The frequency and intensity of catastrophe damages have grown severely in past years, and this tendency is expected to be continued in the future (Sigma, 2003, 2005). Insurance companies increase costs to cover these events, and they need new methods to guarantee the viability of their business activity with reasonable economic yield. The expert opinion about catastrophes shows a high probability of hurricanes like Andrew and earthquakes on the West Coast. Concurrently with these phenomena, zones with a higher risk of hurricanes and earthquakes experience great demographic growth. Following *NPA Data Services,* the population growth, until 2025, in these areas (California, Florida, and Texas) will be higher (up 36.6%) than other areas.

Some insurance companies and financial engineers have designed new ways to assume catastrophic risk through the interchange in financial markets, known as securitization (Sigma, 2001). Securitization means to trade with derivatives whose underlying assets are usually non-negotiated. The securitization allows insurance companies risk reduction and diversification.

Until today, principal catastrophe risk securitizations are catastrophe options and bonds. In this way, the Chicago Board of Trade (CBOT) begins to trade the first derivative specifically designed to insurers in December 1992: CAT-futures and CAT-options. Limitations in these products carried that, in September 1995, these contracts were changed by PCS options. Around the same time (1993), the first catastrophe bonds, CAT-bonds (Mueller, 2002; Cox, 1997), were issued, presenting a high-yield restricted loss to the principal and interest if a catastrophe occurs during the bond life. Actually, the usual catastrophe bonds and options are based on indexes to trigger the contract payment (Loubergé, Kellezi, & Gilli 1999; Muermann, 2003).

The underlying contract is the catastrophic loss ratio $LR = \frac{L(T_2)}{cte}$, defined from the total reported claim amount $L(T_2) = \sum S(t)$, associated to occurred catastrophes during the loss period, and declared until the development period expired.

LR is a random variable because the incurred total loss amount $L(T_2)$ is an unknown value during the contract life; the number of catastrophes, their severity, and the occurrence moment, as well as their reported claim intensity, are all unknown.

In this theoretical approach, one of the main parts of the analysis of derivatives is its value during a certain time interval. Then, a model ought to be developed to evaluate the loss time evolution and thus the underlying loss ratio of both types of contracts $L(T_2)$.

Several authors propose the determination of the underlying loss ratio with the objective of evaluating these derivatives. Usually, the proposed method uses the continuous models based on the hypothesis of a geometric Brownian motion that systematizes the evolution of instantaneous claims. The proposed methods systematize the instantaneous claim evolution and include, through Poisson process, the possibility of appearing severity catastrophes (Cummins, 1995; Geman, 1997; Alegre, Devolder, & Pérez, 2003) proposes a mathematical discrete model for loss ratio that converges, in average sense, to the continuous model assumed in this work.

In this chapter, we formalize a continuous model to determine the underlying loss ratio of the catastrophic contract of capital markets. This model calculates the total amount of a certain catastrophe by aggregating two variables: the reported claim amount and the incurred but not reported claim amount.

The model assumes that catastrophes could be classified in three categories based on its intensity. The first category supposes an instantaneous declaration (without producing IBNR, incurred but not reported claims), and second and third categories represent more severity catastrophes with a temporal evolution.

In order to fit the available data sets to the aforesaid models, a multi-objective optimization approach has been considered. Several methods have been implemented, from the standard evolution strategies (Schwefel, 1981; Ohkura, 2001) to the advanced MOEAs, also using different strategies for the aggregation of partial fitness functions.

The MOEAs have the double goal of guiding the search towards the global Pareto-optimal set and at the same time cover as many solutions as possible. This implies the enlargement of population size and specific procedures in the genetic operators to guarantee a well-distributed sample of the front. Deb (2001) presents a wide analysis of the most outstanding multi-objective evolutionary algorithms. There are several different approaches, depending on the types of operators for selection, mutation, and use of domination. Basically, they can be divided into two groups, non-elitist and elitist algorithms. The non-elitist group includes first approaches, such as VEGA (vector evaluated GA; Schaffer, 1985), which divides population in subsets assigned to different objectives or most recent MOGA (multi-objective GA; Fonseca, 1993), NSGA (non-dominated sorting GA; Srnivas, 1994) and NPGA (niched-Pareto GA; Horn, Nafpliotsis, & Goldberg, 1994), and PESA (Pareto envelope-based selection algorithm; Corne, Knowles, & Oates, 2000). The last algorithms apply non-dominated classification in the GA population, with variations in the type of selection operators. The elitist approaches to MOEA are the most recent techniques; they explicitly use operators to preserve the elites of population in the offspring generation to assure that fitness does not decrease. Among them, we can mention the NSGA-II (non-dominated sorting GA-II; Deb, Pratap, Agrawal, & Meyarivan, 2002), SPEA (strength Pareto EA; Zitzler & Thiele, 1998), SPEA2 (Zitzler, Laumanns, & Thiele, 2001), PAES (Pareto-archived evolution strategy; Knowles, 1999), PESA-II (Corne, Jerram, Knowles, & Oates, 2001), CAEP (cultural algorithms with evolutionary programming; Becerra, 2005), and so forth. These algorithms exploit elitism by explicitly keeping a population with non-dominated solutions until now, so the selection operators are based on comparisons with them. A thorough survey regarding the above models and many others can be found in Tan, Khor, and Lee (2005) and elsewhere.

MATHEMATICAL MODELS

Model Definition

Catastrophes are classified into three groups depending on their severity:

$$k_j^i \quad \text{with} \quad \begin{matrix} i = 1,2,3 \\ j = 1,2,\ldots,N^i(t) \end{matrix} \tag{1}$$

where:

- i = 1, 2, 3 represents small-scale catastrophes, medium-scale catastrophes, and major catastrophes;
- k_j^i are independent random variables identically distributed inside each group; and
- $N^i(t)$ represents the random number of catastrophes in $(0, T_1)$. It is modeled by independent Poisson distributions for each *i-th* type of catastrophe, with intensity $\lambda^i T_1$ with $i = 1,2,3$, where T_1 represents the time interval in which an event of a catastrophe is included in the study.

Finally, the t_j^i with $i = 1,2,3$ and $j = 1,2,\ldots,N^i(t)$ variable is defined as the moment which a catastrophe occurs, with $t_j^i \le T_1$.

By modeling the number of catastrophes occurring with a Poisson distribution, the inter-occurrence time obtained $t_j^i - t_{j-1}^i$ follows an exponential distribution of parameter λ^i.

We can define catastrophe severity as follows:

$$k_j^i = S_j^i(t) + R_j^i(t) \tag{2}$$

where:

- $R_j^i(t)$ is a continuous random variable which determines the incurred but not reported claim amount at the moment t associated to the catastrophe occurred in t_j^i; and
- $S_j^i(t)$ is the continuous random variable total reported claim amount at time t related to a single catastrophe occurred in .

Once a catastrophe of severity has occurred in t_j^i, the claim reporting process associated with this single catastrophe is initiated lasting until the end of a certain period T_2 (in the PCS options, development period). The intensity of claim reporting would seem to be greatest just after the occurrence of the catastrophe and decreases with time.

Then, instantaneous claim process is represented by a differential equation that describes the increase of reported claim amount proportional to the incurred but not reported claim amount, the main variable of the formal model:

$$dS_j^i(t) = \alpha^i\left(t - t_j^i\right) R_j^i(t) dt \tag{3}$$

where $\alpha^i\left(t - t_j^i\right)$ is a real variable function.

By differentiating the equation 2 results:

$$dS_j^i(t) = -dR_j^i(t) \tag{4}$$

and by substituting in equation 3 $dS_j^i(t)$ for equation 4, the differential equation that represent the evolution of is obtained:

$$dR_j^i(t) = -\alpha^i\left(t - t_j^i\right) R_j^i(t) dt \tag{5}$$

Equation 5 shows that the incurred but not reported claim amount in *t* decreases with time according to the function $\alpha^i\left(t - t_j^i\right)$, named claim rate.

In order to determine this function, empirical data is analyzed considering the hypothesis that claims associated to a medium-scale catastrophe (i=2) are declared faster than claims associated to a major catastrophe (i=3), therefore, $\alpha^2\left(t - t_j^i\right) > \alpha^3\left(t - t_j^i\right)$. We suppose the small catastrophes i=1 are reported integrally at the time of occurrence t_j^i, becoming a part of the index instantaneously $\alpha^1\left(t - t_j^i\right) \to \infty$.

In the general deterministic model (equation 5), randomness is incorporated into the incurred but not reported claim process, applying a

Wiener Process to perturb claim reporting rate $\alpha^i(t-t_j^i)$. Therefore, the following stochastic differential equation is derived:

$$dR_j^i(t) = -\alpha^i(t-t_j^i)R_j^i(t)dt + \sigma^i R_j^i(t)dw_j^i(t) \quad (6)$$

with $t_j^i \in [0,T_1]$ y $t \in [t_j^i, T_2]$ and where:

- $\alpha^i(t-t_j^i)$ is the claim reporting rate function that represents the process drift;
- σ^i is a constant value that represents the process volatility; and
- $w_j^i(t-t_j^i)$ is a Wiener process related to each different catastrophe.

Wiener process randomizes each catastrophe's claim reporting rate. Then each catastrophe follows its own behavior expressed by independent Wiener processes with different volatility parameters. In order to preserve the decreased behavior of incurred but not reported claim amount, a small volatility value is required. This value assures a quasi-null probability of considering an increase in $R_j^i(t)$.

General Resolution for Proposed Model

Applying Itô's Lemma in equation 6, we obtain $R_j^i(t)$:

$$R_j^i(t) = k_j^i e^{-\int_0^{t-t_j^i}\alpha^i(s)ds - \frac{(\sigma^i)^2}{2}(t-t_j^i)+\sigma^i w_j^i(t-t_j^i)} \quad (7)$$

with the following bound constrains:

- If $t = t_j^i$ then $R_j^i(t_j^i) = k_j^i$, the incurred but not reported claim amount is the total amount of catastrophe.
- If $t \to \infty$ then $R_j^i(\infty) = 0$, the incurred but not reported claim amount is 0.

From the relation defined between and defined in equation 2, we obtain :

$$S_j^i(t) = k_j^i\left[1 - e^{-\int_0^{t-t_j^i}\alpha^i(s)ds - \frac{(\sigma^i)^2}{2}(t-t_j^i)+\sigma^i w_j^i(t-t_j^i)}\right] \quad (8)$$

Symmetrically to the incurred but not reported claim amount, bound constrains for the reported claims amount are:

- If $t = t_j^i$ then $S_j^i(t_j^i) = 0$, the reported claims amount is 0.
- If $t \to \infty$ then $S_j^i(t_j^i) = k_j^i$, the reported claim amount is the total amount of catastrophe.

Solution for Constant Claim Reported Rate

When claim rate is considered constant $\alpha^i(s) = \alpha^i$, we can calculate the integral of equation 7 as:

$$\int_0^{t-t_j^i}\alpha^i(s)ds = \int_0^{t-t_j^i}\alpha^i ds = \alpha^i \cdot (t-t_j^i) \quad (9)$$

By substituting this result in equation 7, the incurred but not reported claim amount at *t* is[1]:

$$R_j^i(t) = \begin{cases} 0 & i = 1 \\ k_j^i e^{-\left(\alpha^i + \frac{(\sigma^i)^2}{2}\right)(t-t_j^i)+\sigma^i w_j^i(t-t_j^i)} & i = 2,3 \end{cases} \quad (10)$$

and the reported claim amount until *t* is:

$$S_j^i(t) = \begin{cases} k_j^i & i = 1 \\ k_j^i\left[1 - e^{-\left(\alpha^i + \frac{(\sigma^i)^2}{2}\right)(t-t_j^i)+\sigma^i w_j^i(t-t_j^i)}\right] & i = 2,3 \end{cases} \quad (11)$$

$R_j^i(t)$ is a random variable whose distribution depends on total catastrophe amount distribution k_j^i. With the hypothesis that k_j^i is a constant value, $R_j^i(t)$ will follow a lognormal distribution,

where normal distribution parameters associated are:

$$N\left(\ln k_j^i - \left(\alpha^i + \frac{\sigma^2}{2}\right)\left(t - t_j^i\right), \sigma^i \sqrt{t - t_j^i}\right) \tag{12}$$

This means, in average, that the incurred but not reported claim amount shows an exponential decreasing asymptotic to abscises axis, and then the reported claim amount increases asymptotically to k_j^i,

$$E\left[R_j^i(t)\right] = k_j^i e^{-\alpha^i\left(t - t_j^i\right)} \quad E\left[S_j^i(t)\right] = k_j^i\left[1 - e^{-\alpha^i\left(t - t_j^i\right)}\right] \tag{13}$$

Solution for Asymptotic Claim Reported Rate

When claim rate is considered asymptotic $\alpha^i(s) = \alpha^i \cdot \left(1 - e^{-\beta^i s}\right)$, we can calculate the integral of equation 7 as:

$$\int_0^{t-t_j^i} \alpha^i(s)ds = \int_0^{t-t_j^i} \alpha^i \cdot \left(1 - e^{-\beta^i s}\right)ds = \alpha^i \cdot \left(t - t_j^i\right) - \frac{\alpha^i}{\beta^i} \cdot \left(1 - e^{-\beta^i\left(t - t_j^i\right)}\right) \tag{14}$$

By substituting this result in equation 7, the incurred but not reported claim amount at *t* is:

$$R_j^i(t) = \begin{cases} 0 & i = 1 \\ k_j^i e^{-\left(\alpha^i + \frac{\left(\sigma^i\right)^2}{2}\right)\left(t - t_j^i\right) + \sigma^i w_j^i\left(t - t_j^i\right)} e^{\frac{\alpha^i}{\beta^i}\left(1 - e^{\beta^i\left(t - t_j^i\right)}\right)} & i = 2,3 \end{cases} \tag{15}$$

and the reported claim amount until *t* is:

$$S_j^i(t) = \begin{cases} k_j^i & i = 1 \\ k_j^i\left[1 - e^{-\left(\alpha^i + \frac{\left(\sigma^i\right)^2}{2}\right)\left(t - t_j^i\right) + \sigma^i w_j^i\left(t - t_j^i\right)} e^{\frac{\alpha^i}{\beta^i}\left(1 - e^{-\beta^i\left(t - t_j^i\right)}\right)}\right] & i = 2,3 \end{cases} \tag{16}$$

In the case of , when volatility is null, equations 15 and 16 are:

$$R_j^i(t) = \begin{cases} 0 & i = 1 \\ k_j^i e^{-\left(\alpha^i\right)\left(t - t_j^i\right)} e^{\frac{\alpha^i}{\beta^i}\left(1 - e^{\beta^i\left(t - t_j^i\right)}\right)} & i = 2,3 \end{cases} \tag{17}$$

$$S_j^i(t) = \begin{cases} 0 & i = 1 \\ k_j^i\left[1 - e^{-\left(\alpha^i\right)\left(t - t_j^i\right)} e^{\frac{\alpha^i}{\beta^i}\left(1 - e^{\beta^i\left(t - t_j^i\right)}\right)}\right] & i = 2,3 \end{cases} \tag{18}$$

Solution for Hybrid (Constant and Asymptotic) Claim Reported Rate

When claim rate is considered hybrid (increasing linearly until moment s_m^i and constant $\alpha^i(s) = \alpha^i$ from this moment),

$$\alpha^i(s) = \begin{cases} \frac{\alpha^i}{s_m^i} s & 0 \le s \le s_m^i \\ \alpha^i & s > s_m^i \end{cases} \tag{19}$$

we can calculate the integral of the equation in two different situations:

- If evaluation moment t is previous to the moment of claim rate change $t_j^i \le t \le t_j^i + s_m^i$, then:

$$\int_0^{t-t_j^i} \alpha^i(s)ds = \int_0^{t-t_j^i} \frac{\alpha^i}{s_m^i} sds = \frac{\alpha^i \cdot \left(t - t_j^i\right)^2}{2s_m^i} \tag{20}$$

By substituting this result in equation 7, the incurred but not reported claim amount at *t* is:

$$R_j^i(t)=\begin{cases}0 & i=1\\ k_j^i e^{-\left(\frac{\alpha^i}{2s_m^i}\left(t-t_j^i\right)+\frac{\left(\sigma^i\right)^2}{2}\right)\cdot\left(t-t_j^i\right)+\sigma^i w_j^i\left(t-t_j^i\right)} & i=2,3\end{cases} \tag{21}$$

and the reported claim amount until *t* is:

$$S_j^i(t)=\begin{cases}k_j^i & i=1\\ k_j^i\left[1-e^{-\left(\frac{\alpha^i}{2s_m^i}\left(t-t_j^i\right)+\frac{\left(\sigma^i\right)^2}{2}\right)\cdot\left(t-t_j^i\right)+\sigma^i w_j^i\left(t-t_j^i\right)}\right] & i=2,3\end{cases} \tag{22}$$

- If evaluation moment *t* is later than the moment of claim rate change $t>t_j^i+s_m^i$, then,

$$\int_0^{t-t_j^i}\alpha^i(s)ds=\int_0^{s_m^i}\frac{\alpha^i}{s_m^i}sds\int_{s_m^i}^{t-t_j^i}\alpha^i ds=\alpha^i\cdot\left(t-t_j^i\right)-\frac{\alpha^i s_m^i}{2} \tag{23}$$

By substituting this result in equation 7, the incurred but not reported claim amount at *t* is:

$$R_j^i(t)=\begin{cases}0 & i=1\\ k_j^i e^{-\left(\alpha^i+\frac{\left(\sigma^i\right)^2}{2}\right)\cdot\left(t-t_j^i\right)+\sigma^i w_j^i\left(t-t_j^i\right)}e^{\frac{\alpha^i s_m^i}{2}} & i=2,3\end{cases} \tag{24}$$

and the reported claim amount until *t* is:

$$S_j^i(t)=\begin{cases}k_j^i & i=1\\ k_j^i\left[1-e^{-\left(\alpha^i+\frac{\left(\sigma^i\right)^2}{2}\right)\cdot\left(t-t_j^i\right)+\sigma^i w_j^i\left(t-t_j^i\right)}e^{\frac{\alpha^i s_m^i}{2}}\right] & i=2,3\end{cases} \tag{25}$$

In the case of $\sigma^i=0$, when volatility is null, equations 24 and 25 are:

$$R_j^i(t)=\begin{cases}0 & i=1\\ k_j^i e^{-\alpha^i\left(t-t_j^i\right)}e^{\frac{\alpha^i s_m^i}{2}} & i=2,3\end{cases} \tag{26}$$

$$S_j^i(t)=\begin{cases}0 & i=1\\ k_j^i\left[1-e^{-\alpha^i\left(t-t_j^i\right)}e^{\frac{\alpha^i s_m^i}{2}}\right] & i=2,3\end{cases} \tag{27}$$

Evaluation of $L(T_2)$

Loss ratio of catastrophic insurance derivatives is defined as:

$$LR=\frac{L(T_2)}{\text{cte}} \tag{28}$$

where $L(T_2)$ is the total claim reporting amount at T_2 associated to the catastrophe occurrence in the $[0,T_1]$ period.

$L(T_2)$ is defined by aggregation of three components:

$$L(T_2)=L^1(T_2)+L^2(T_2)+L^3(T_2)=\sum_{i=1}^{3}L^i(T_2) \tag{29}$$

where $L^i(T_2)$ is the total claim reporting amount at T_2 associated to type i catastrophe occurred in $[0,T_1]$

$$L^i(T_2)=\sum_{j=1}^{N^i(T_1)}k_j^i\left[1-e^{-\int_0^{T_2-t_j^i}\alpha^i(s)ds-\frac{\left(\sigma^i\right)^2}{2}\left(T_2-t_j^i\right)+\sigma^i w_j^i\left(T_2-t_j^i\right)}\right] \tag{30}$$

Then $L(T_2)$ could be expressed as:

$$L(T_2)=\sum_{i=1}^{3}L^i(T_2)=\sum_{j=1}^{N^1(T_1)}k_j^1+\sum_{i=2}^{3}\sum_{j=1}^{N^i(T_1)}k_j^i$$

$$\left[1 - e^{-\int_0^{T_2 - t_j^l} \alpha^l(s)ds - \frac{\left(\sigma^l\right)^2}{2}\left(T_2 - t_j^l\right) + \sigma^l w_j^l \left(T_2 - t_j^l\right)}\right] \tag{31}$$

MULTI-OBJECTIVE OPTIMIZATION WITH EVOLUTIONARY ALGORITHMS

The concept of optimization refers to the process of finding one or more feasible solutions of a problem which corresponds to the extreme values (either maximum or minimum) of one or more objective functions. Initial approaches to optimization were focused on the case of solving problems involving only one objective. However, as most real-world optimization problems involve many objectives, the research on this area has rapidly broaden this attention to encompass what has been called multi-objective optimization.

When moving from the single-objective approach to the multi-objective one, the question of objective trade-offs arises. This is caused by the fact that a solution might be optimal with respect to one objective, but would be less-than-optimal for the rest. Therefore the globally optimal solution to a given problem would be a compromise between feasible ones of the different objectives.

This leads us to the dilemma that hovers over all multi-objective optimization algorithms: how to select a solution from the set of feasible ones. This selection implies the use of high-level pondering criteria. An ideal strategy for solving this issue can be devised as:

1. Find a set of partially optimal solutions using different selection criteria, and then;
2. choose among the elements of the set using a high-level knowledge.

Classical approaches to this class of problems are based on preference-based search procedures. In these procedures, fitness values produced by each objective are scaled via a relative preference vector. This kind of process is repeated on a point-by-point basis generating a (hopefully) better solution after each iteration.

As a result of the need for better optimization methods, a number of novel optimization algorithms of stochastic nature have been proposed. Among these new models are *evolutionary algorithms (EA)*. EAs mimic nature's evolutionary principles to control the progress of the search toward an optimal solution. The fundamental difference of this type of algorithm with regard to the classic ones is that instead of focusing on a single solution, EAs maintain a population of solution. This population is refined in each iteration using operators related to the processes of natural selection, crossover, mutation, and so on.

In this work we discuss the application of an evolutionary algorithm known as strength Pareto evolutionary algorithm (SPEA; Zitzler & Thiele, 1998, 1999). In particular we apply an improvement of this method known as SPEA2 (Zitzler, Laumanns, & Thiele, 2002). However, in order to properly present the details of these models, we feel that we first need to formally sketch the theoretical foundation on what they are built upon.

Multi-Objective Optimization

In general terms, the multi-objective optimization problem can be formally stated as

$$\begin{aligned} \text{minimize or maximize} \quad & f_m(\mathbf{x}), && m = 1, \ldots, M; \\ \text{suject to} \quad & g_j(\mathbf{x}) \geq 0, && j = 1, \ldots, J; \\ & h_k(\mathbf{x}) = 0, && k = 1, \ldots K; \\ & x_i^l \leq x_i \leq x_i^u, && i = 1, \ldots, n. \end{aligned} \tag{32}$$

where $\mathbf{x} \in \mathbf{R}^n$ is the vector of variables to be optimized, each bounded by $\mathbf{x}_i^l$ and $\mathbf{x}_i^u$; $f_m(...)$ are the objective functions being minimized or maximized; and $g_j(...)$ and $h_k(...)$ are the inequality and equality constraints, respectively.

Following the duality principle 0, we can rephrase any maximization objective as a minimization one (and vice versa). Thanks to this and because of the fact that we are not going to deal with constraints, we can rewrite equation 33 as:

$$\text{minimize} \quad \mathbf{z}_m = f_m(\mathbf{x}), \quad m = 1, \ldots, M; \qquad (33)$$

Here the vector $\mathbf{z}_m$ designates the quality of a given solution and is known as the objective vector.

In order to rank the quality of a solution with respect to others, the concept of dominance has been introduced.

Evolutionary Algorithms

An evolutionary algorithm (also EA, evolutionary computation, artificial evolution) is a generic term used to indicate any population-based metaheuristic optimization algorithm that uses mechanisms inspired by biological evolution, such as reproduction, mutation and recombination, and so forth. Candidate solutions to the optimization problem play the role of individuals in a population, and the cost function determines the environment within which the solutions "live." Evolution of the population then takes place after the repeated application of the above operators.

Because they do not make any assumption about the underlying fitness landscape, it is generally believed that evolutionary algorithms perform consistently well across all types of problems (see, however, the no-free-lunch theorem; Wolpert, 1997).

Genetic algorithms are a particular class of evolutionary algorithm. Genetic algorithms are typically implemented as a computer simulation in which a population of abstract representations (called chromosomes) of candidate solutions (called individuals) to an optimization problem evolves toward better solutions. GA theory is elegant, fascinating, and full of subtle variations that may—or may not—be useful in certain applications. Since Holland's (1975) seminal work, numerous variations of the conventional GA have been proposed. Most of the variations are well beyond the scope of this work.

Strength Pareto Evolutionary Algorithm

The SPEA was proposed by Zitzler and Thiele (1998, 1999). This algorithm implements elitism by preserving an external population $\overline{P}$. This external population stores a fixed amount of non-dominated solutions that were discovered since the beginning of the simulation. Then, after every iteration of the algorithm, if a new non-dominated solution is found, it is compared with those present in $\overline{P}$ preserving the best solutions. However, SPEA goes beyond just keeping an elite set of solutions. It uses the solutions stored in $\overline{P}$ along with the plebeian solutions in all genetic operations with the hope of inducing a better performance of the search in the solution space.

Although SPEA has produced a number of relevant results, it has been pointed out that it has some potential weaknesses (Zitzler et al., 2002).

SPEA2 was proposed as an attempt to overcome the limitations of SPEA. It keeps the overall scheme of its predecessor, but in contrast to SPEA, SPEA2 uses a fine-grained fitness assignment strategy that incorporates density information. Furthermore, the external population has a fixed size; therefore, whenever the number of non-dominated solutions is less than the predefined archive size, the archive

is filled up by dominated individuals. Finally, the clustering technique used to prune the external population has been replaced by an alternative truncation method, which has similar features but does not miss boundary points.

PROBLEM STATEMENT

Multi-Objective Problem Definition Over Empirical Data

The basic problem addressed in this work is the adjustment of mathematical models described earlier to fit data representing real disasters. This process is formulated as an optimization over the parameters to maximize the models fitting with respect to real data.

Data are related with seven documented disasters (floods) that occurred in different Spanish regions: Alcira (October 1, 1991), Donosti (June 23, 1992), Barcelona 1 (September 14, 1999), Barcelona 2 (October 20, 2000), Zaragoza (October 20, 2000), Valencia (October 20, 2000), and Murcia (October 20, 2000). The data, obtained from the "Consorcio de Compensación de Seguros",[2] contains the details of amounts and dates for claims raised after each disaster. Data have been filtered and aggregated to obtain temporal series (evenly distributed per week, with time origin in zero) with the variable $R_j^i(t)$, as defined earlier with an incurred but not reported claim amount (IBNR). The resulting time series with these values, as a relative percentage of total amount for each disaster, are presented in Table 1.

Given these data, the optimization problem is the search for parameters of three certain distribution functions in order to minimize the prediction error for all data samples (total sum per series). For a model $R(t_i)$, with data y_{ij} for j-th disaster (with length N_j), the performance value to be minimized in the adjustment of R(.) is:

Table 1. ES-optimization of model 1, specific adjustment

	NON REPORTED CLAIM AMOUNT (%)						
	Alcira 01/10/1991	San Sebastián 23/06/1992	Barcelona 14/09/1999	Barcelona2 20/10/2000	Zaragoza 20/10/2000	Valencia2 20/10/2000	Murcia 20/10/2000
WEEK							
0	100.00	100.00	100.00	100.00	100.00	100.00	100.00
1	84.94	88.08	90.68	85.95	60.11	97.54	88.46
2	53.65	36.04	68.38	61.73	56.91	80.18	75.55
3	34.96	23.67	50.68	34.98	38.30	60.15	48.70
4	24.05	16.68	41.42	24.07	29.79	43.16	31.13
5	18.86	12.29	31.58	14.05	23.40	31.96	21.41
6	13.36	9.94	25.43	12.41	19.15	27.55	15.78
7	10.53	8.72	19.56	8.52	18.09	19.54	11.27
8	8.04	7.76	16.68	5.23	15.43	15.29	8.71
9	6.94	6.80	13.28	5.23	15.43	14.76	8.24
10	5.23	5.78	10.54	5.08	15.43	14.70	6.57
11	4.08	5.18	8.15	3.44	10.64	11.06	5.36
12	3.71	4.33	6.80	3.59	6.91	8.46	4.00
13	3.56	3.45	6.13	2.24	3.72	6.98	3.38
14	2.60	2.69	3.41	1.20	3.72	6.21	2.60
15	1.75	1.81	3.41	0.60	3.72	5.17	2.80
16	1.30	1.59	2.61	0.60	2.66	4.22	2.29
17	0.77	1.39	1.81	0.60	2.66	3.50	2.25
18	0.29	1.16	1.26	0.45	2.13	2.72	2.14
19	0.00	0.96	0.56	0.45	1.60	2.26	1.67
20	0.00	0.76	0.00	0.45	1.60	1.88	1.28
21	0.00	0.45	0.00	0.45	1.60	1.69	1.09
22	0.00	0.28	0.00	0.00	1.60	1.61	0.93
23	0.00	0.20	0.00	0.00	1.60	0.90	0.66
24	0.00	0.17	0.00	0.00	1.60	0.54	0.62
25	0.00	0.11	0.00	0.00	1.60	0.36	0.66
26	0.00	0.06	0.00	0.00	1.60	0.19	0.16
27	0.00	0.00	0.00	0.00	0.00	0.00	0.00

$$\varepsilon_j = \sum_{i=1}^{N_j} \left(R(t_i) - y_{ij} \right)^2 \tag{34}$$

The three models analyzed and their parameters, under the particular conditions mentioned above, are:

- **Model 1:** Constant claim reported model: an exponential distribution (from equation 13) where a single shape parameter, a (positive values), must be adjusted to the real data distribution:

$$R_1(t_i) = 100 \exp(-\alpha t_i)$$

- **Model 2:** Asymptotic claim reported model: following equation 17, two shape

parameters (a and b, both positive) must be adjusted to obtain the maximum fitting of data:

$$R_2(t_i) = 100\exp(-\alpha t_i) * \exp((1.0 - \exp(-\beta * t_i)) * \alpha/\beta)$$

- **Model 3:** Hybrid model: following equations 19 and 26, two parameters must be fitted: the time when claim rate changes from linearly increasing to constant, s_m (a positive discrete parameter since weeks are represented with integers); and the shape parameter α:

$$R_3(t_i) = \begin{cases} 100.0 * \exp(-0.5\alpha\, t_i^2/s_m), & t_i \le s_m \\ 100.0 * \exp(-0.5\alpha\, (2t_i - s_m)), & t_i > s_m \end{cases} \quad (35)$$

Finally, this optimization process has been carried out in two ways: as a local adjustment for each disaster ε_j, and a global optimization trying to fit all disasters at the same time (optimization of total sum of errors $\sum_{j=1}^{7} \varepsilon_j$). In the case of using a scalar evolution strategies algorithm, a scalar fitness function was used for each disaster, and each adjustment was a different optimization run, including the additional optimization of total sum. However, the use of a multi-objective algorithm allowed us to perform optimizations searching in a single run all parameter sets suitable for the seven situations and for the total sum (all included in the Pareto front) at the same time.

RESULTS

The general structure of this section with results for the optimization problem presented above is as follows. In the first place, for each of the three models, the particular optimizations are obtained for each individual data set (fitting the models for each region) and the values if the sum of error of all series is minimized. So, this first phase presents results for seven independent optimizations, all of them carried out with a evolution strategies algorithm (optimization of a scalar function depending of one parameter for first model, and of two parameters for the second and third ones). In a second phase, the optimizations with SPEA2 algorithm are presented. In this case, each model has been optimized considering the six data sets as different objectives to be fitted at the same time with a multi-objective optimization. In this case, 2D and 3D views of the final population of solutions (with seven dimensions) after convergence are depicted. From this set of solutions (50 individuals in populations) representing the whole Pareto-optimal front, the particular points with optimal values for each individual region and for the total sum are selected, in order to be compared with those generated with a scalar optimization for particular functions.

Model 1 (Exponential Distribution, One Parameter)

For each individual disaster, the optimization procedure with evolution strategies has been applied, with the results indicated in Table 2. There, the optimum parameter α for each data subset is indicated, together with the optimum value achieved for that case. Table 3 shows the results after a global optimization considering the sum of errors if a single parameter is used to fit all series. Obviously, the global strategy presents worse results for each disaster with respect to the optimum value obtained specifically for that case. This adjustment of model is depicted in Figure 1 with all data series. There can be seen the difficulty to obtain a "global" value of parameter fitting all series and the necessity to derive the appropriate value for each particular series.

Table 2. ES-optimization of model 1, specific adjustment

Series	Alfa	Squared Error Sum
Alcira	0,317	195,21
Donosti	0,394	680,02
Barcelona1	0,220	138,59
Barcelona2	0,319	341,54
Zaragoza	0,277	533,15
Valencia	0,199	534,69
Murcia	0,259	523,72
TOTAL		2946,93

Table 3. ES-optimization of model 1, adjustment of sum

Series	Alfa	Squared Error Sum
Alcira		399,68
Donosti		1673,43
Barcelona1		605,15
Barcelona2		575,36
Zaragoza		535,77
Valencia		1654,44
Murcia		545,16
TOTAL	0,271609	5988,99

The SPEA2 algorithm results are depicted in Figures 2 and 3. Since the results comprise the populations with seven-dimension individuals, some views have been selected for graphical illustration, presenting some 3D and 2D groups of objectives (model fitting to combinations of different regions). In the figures, the success to obtain a well-distributed sample of Pareto optimal front (non-dominated solutions) can be clearly appreciated.

Finally, from the whole population after SPEA2 algorithm's convergence (50 individuals), some particular points have been selected: those corresponding to a minimum value for each individual dimension (each one of the seven regions) and for the total sum. Their

Figure 1. Curves obtained with model 1 (specific and global optimizations)

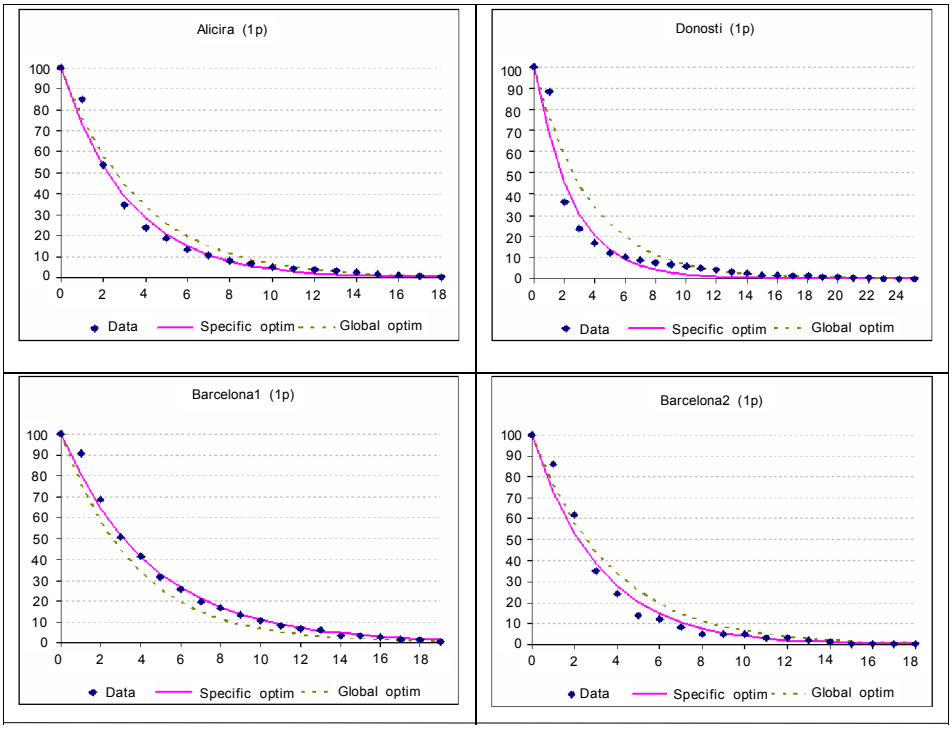
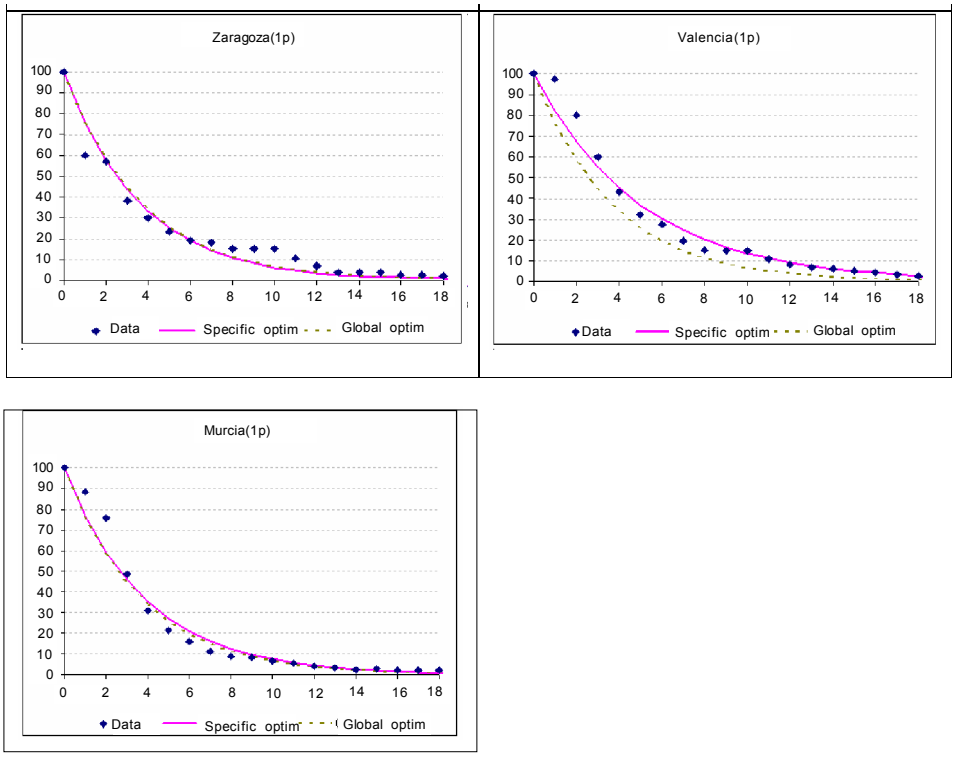


Figure 1. Curves obtained with model 1 (specific and global optimizations) (cont.)



values (parameters and corresponding errors) are displayed in Tables 3 and 4. If we compare these values with those in Tables 1 and 2, we can check their closeness. The conclusion is that the multi-objective optimization algorithm achieves, in a single run, a set of solutions containing all Pareto optima, including as particular individuals the values generated with seven (conventional) scalar optimizations for the specific objectives defined.

Model 2 (Exponential Distribution, Two Parameters)

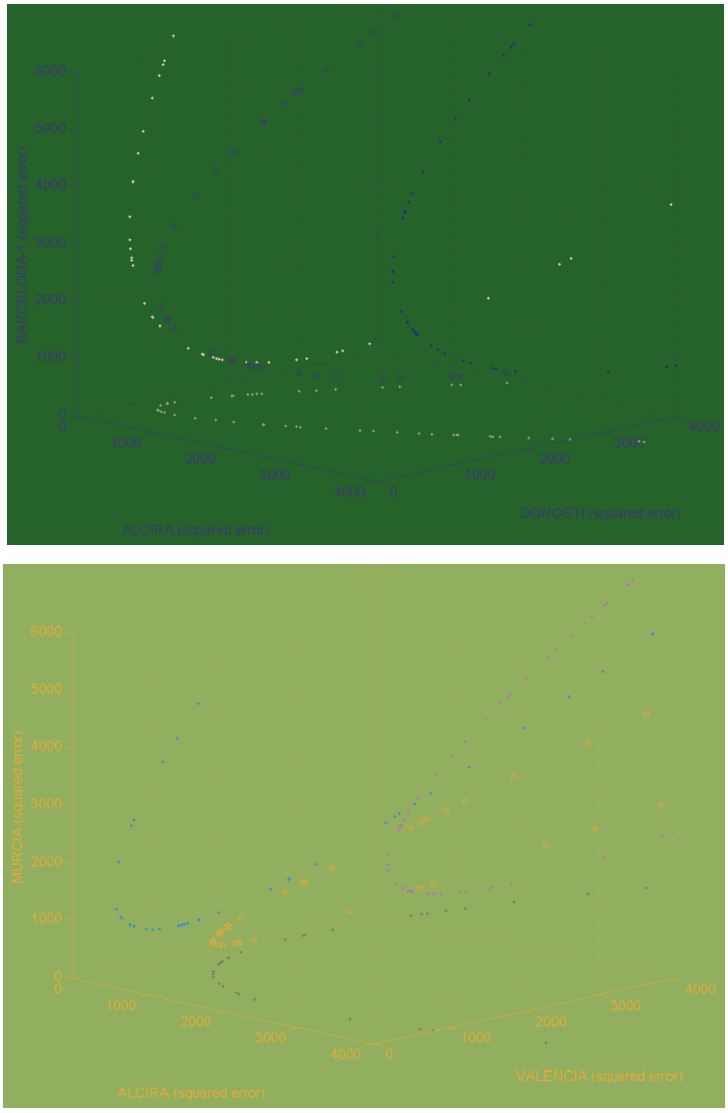
A similar analysis can be done for this second model with two parameters. In the first phase, the optima found with particular optimization runs are presented in Tables 5, 6, and Figure 4. Compared with the previous model (Tables 1 and 2), we can see that the capability of this second model to fit the data samples is higher, with an error reduction for all cases.

Regarding the SPEA2 results for this model, the results are presented in Figures 5 and 6, and selected values in the front appear in Tables 7 and 8. The same comments can be said about the capability to find the whole set of non-dominated solutions in a single run, including the previous values (or points very close to them) as particular individuals in the final population.

Model 3 (Hybrid Model, Two Parameters)

Finally, a similar analysis of results can be carried out for this third model, corresponding to a hybrid combination of exponential distributions. Tables 9 and 10, and Figure 7 represent

Figure 2. 3D views of Pareto front for model 1 (triads of different regions)



the values for the first phase, with seven specific optimizations, while Figures 8 and 9 with Tables 11 and 12 show the results regarding the multi-objective optimization run. Compared with the values obtained in the same situation for the previous models, we can conclude the highest capability of this model to fit the available data.

CONCLUSION

The proposed model for the total reported claim amount probability distribution allows the classification of catastrophes and stochastic estimation of model parameters. The core of the model is the definition of the reporting dynamics based on an exponential model with a claim rate function which can be applied to the incurred but not reported claim amount. Initially, we have considered a deterministic claim reporting rate for each type of catastrophe. The relative simplicity of the presented model eases parameter estimation and simulation.

In this work, the application of machine learning techniques allows estimation of the parameters of the model by the optimization of

Figure 3. 2D views of Pareto front for model 1 (pairs of first region with the rest)

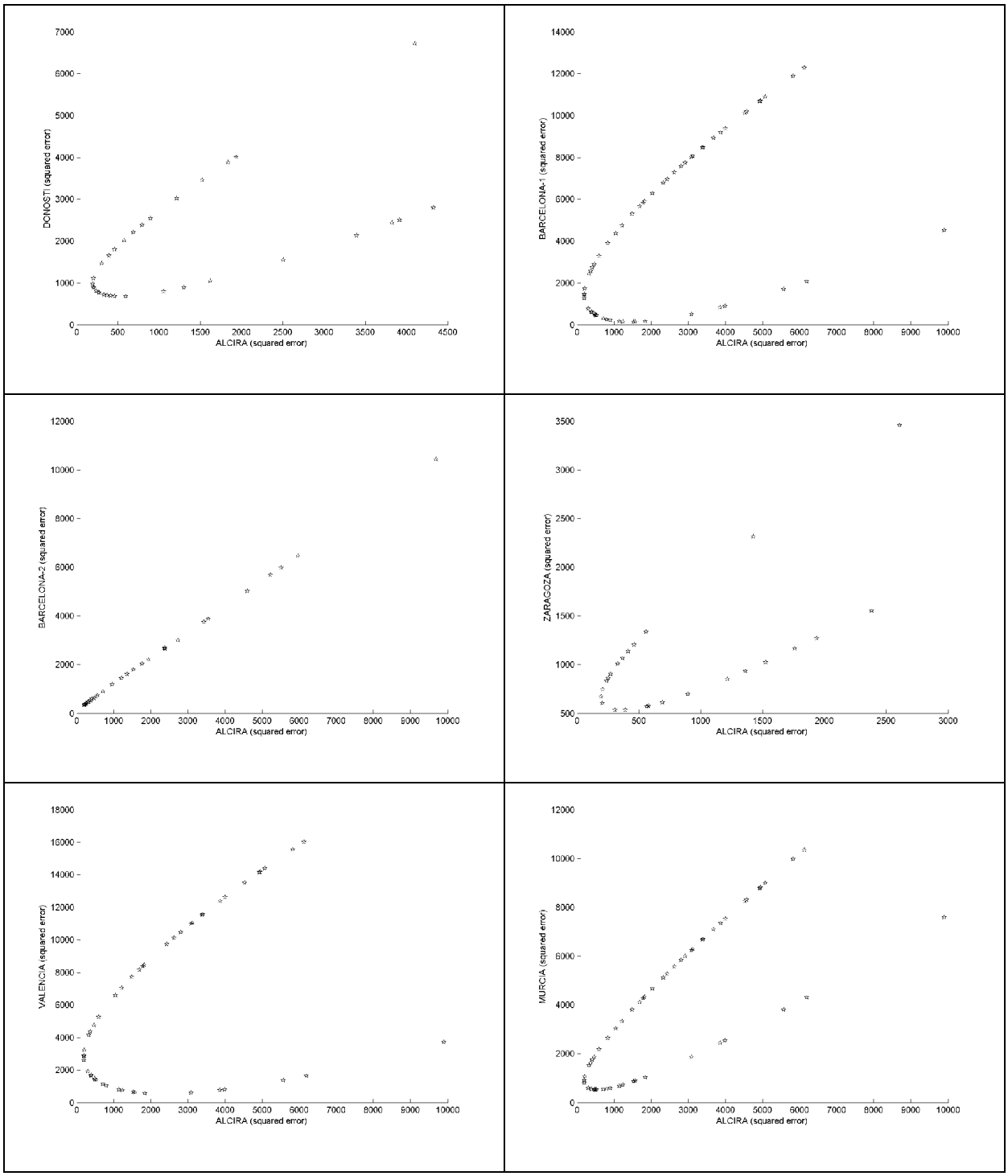


Table 4. SPEA-2 solutions for model 1, optima for specific regions

Series	Alfa	Squared Error Sum
Alcira	0,319	195,639
Donosti	0,395	680,119
Barcelona1	0,218	141,626
Barcelona2	0,319	341,539
Zaragoza	0,279	533,663
Valencia	0,198	534,837
Murcia	0,257	524,721
TOTAL		2952,145

Table 5. SPEA-2 solutions for model 1, optimum for total sum

Series	Alfa	Squared Error Sum
Alcira		413,86
Donosti		1.702,41
Barcelona1		584,66
Barcelona2		590,55
Zaragoza		537,38
Valencia		1.621,58
Murcia		540,79
TOTAL	0,27026	5.991,22

Table 6. ES-optimization of model 2, specific adjustment

Series	Alfa	Beta	Squared Error Sum
Alcira	0,371	2,252	95,95
Donosti	0,521	1,750	493,84
Barcelona1	0,248	2,006	35,01
Barcelona2	0,436	1,132	77,47
Zaragoza	0,277	262,232	535,98
Valencia	0,252	0,962	172,37
Murcia	0,383	0,750	132,51
TOTAL			1543,12

Table 7. ES-optimization of model 1, adjustment of sum

Series	Alfa	Beta	Squared Error Sum
Alcira			310,05
Donosti			1622,69
Barcelona1			554,14
Barcelona2			385,26
Zaragoza			876,14
Valencia			1489,53
Murcia			319,26
TOTAL	0,30458	2,68828	5557,07

Figure 4. Curves obtained with model 2 (specific and global optimizations)

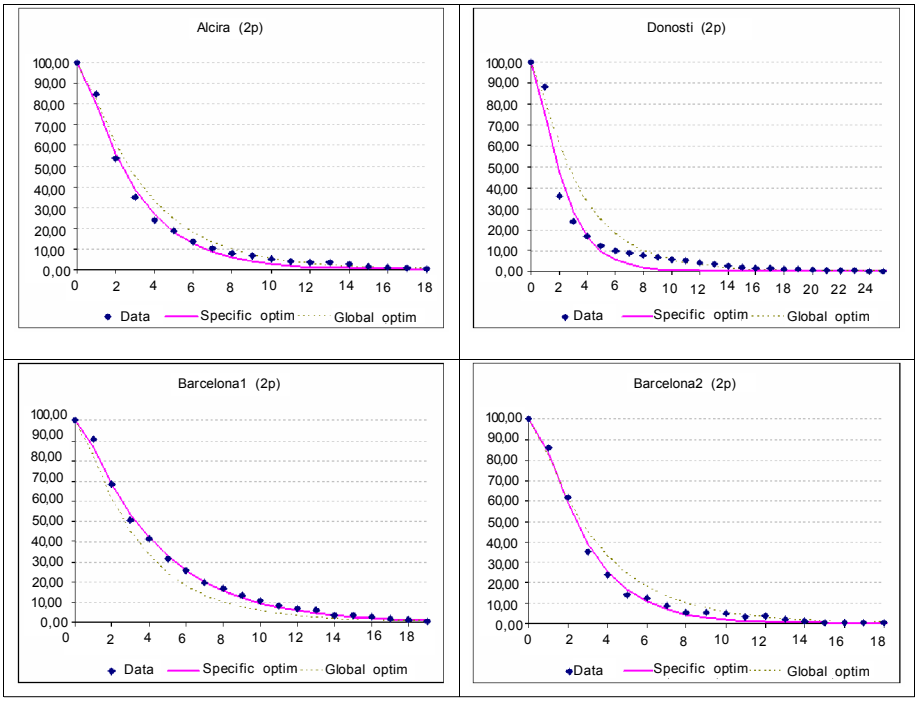


Figure 4. Curves obtained with model 2 (specific and global optimizations) (cont.)

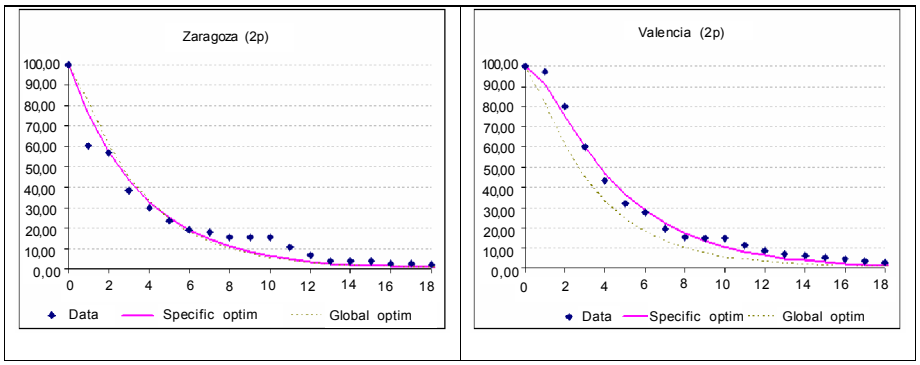
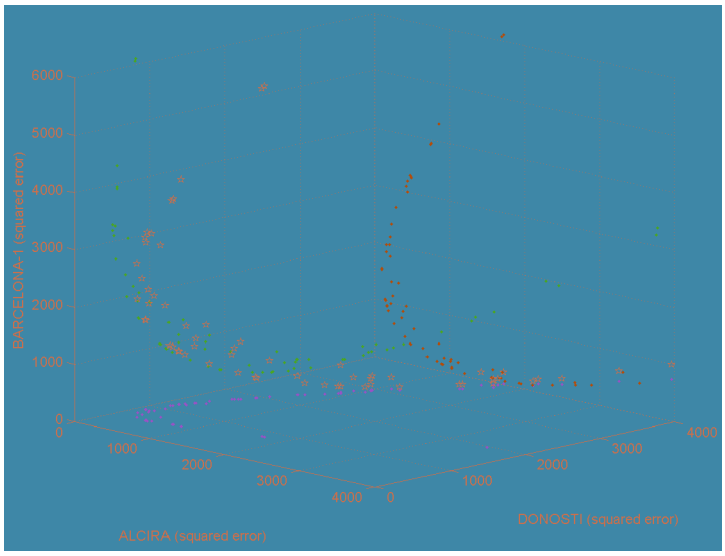


Figure 5. 3D views of Pareto front for model 2 (triads of different regions)



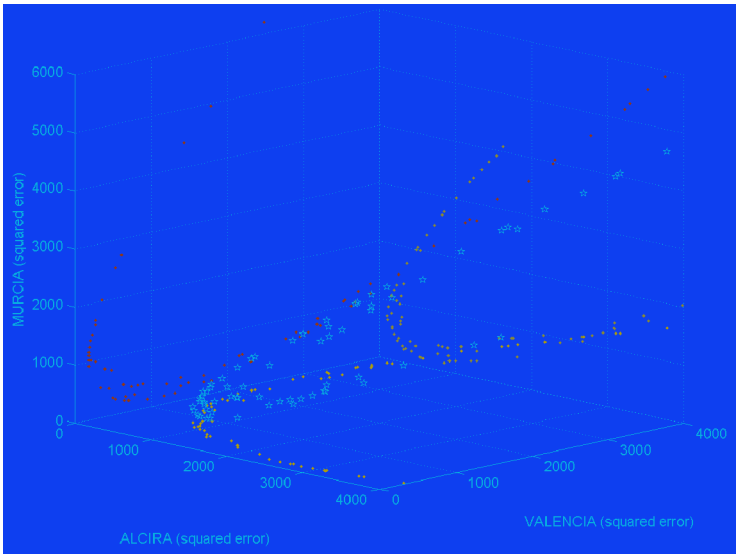


Figure 6. 2D views of Pareto front for model 2 (pairs of first region with the rest)

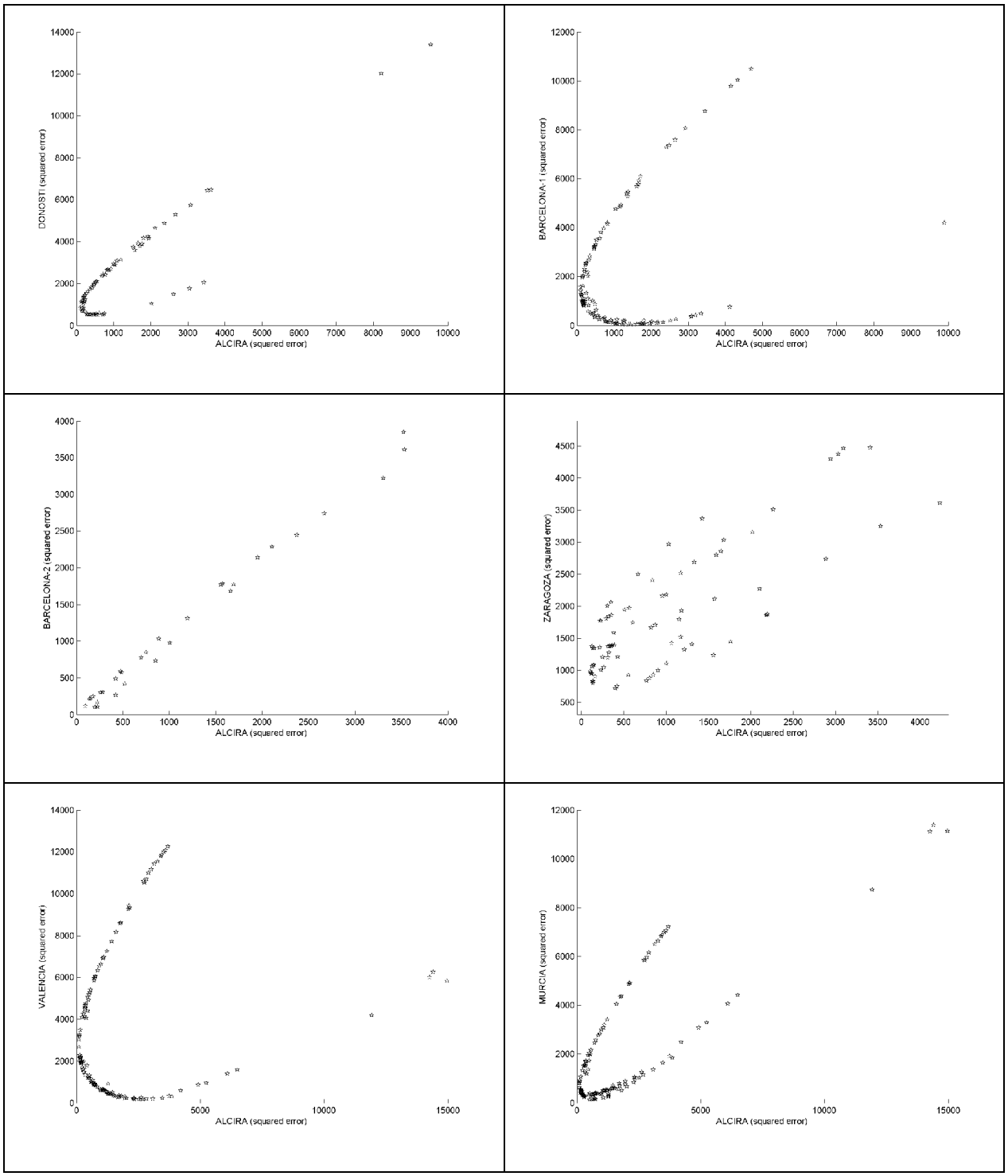


Table 8. SPEA-2 solutions for model 2, optima for specific regions

Series	Alfa	Beta	Squared Error Sum
Alcira	0,385	1,836	100,023
Donosti	0,515	1,836	494
Barcelona1	0,245	2,368	38,439
Barcelona2	0,385	1,836	103,019
Zaragoza	0,282	4,726	726,580
Valencia	0,257	0,805	183,876
Murcia	0,385	0,805	140,528
TOTAL			1786,614

Table 9. SPEA-2 solutions for model 2, optimum for total sum

Series	Alfa	Beta	Squared Error Sum
Alcira			261,74
Donosti			1.536,25
Barcelona1			666,76
Barcelona2			284,76
Zaragoza			1.043,82
Valencia			1.599,72
Murcia			279,64
TOTAL	0,32937	1,83620	5.672,69

Table 10. ES-optimization of model 3, specific adjustment

Series	Alfa	s	Squared Error Sum
Alcira	0,378	0,936	83,92
Donosti	0,545	1,196	424,13
Barcelona1	0,248	0,961	26,54
Barcelona2	0,423	1,470	65,25
Zaragoza	0,274	0,002	534,56
Valencia	0,250	1,982	129,31
Murcia	0,370	2,212	104,94
TOTAL			1368,65

Table 11. ES-optimization of model 1, adjustment of sum

Series	Alfa	s	Squared Error Sum
Alcira	0,378	0,936	83,92
Donosti	0,545	1,196	424,13
Barcelona1	0,248	0,961	26,54
Barcelona2	0,423	1,470	65,25
Zaragoza	0,274	0,002	534,56
Valencia	0,250	1,982	129,31
Murcia	0,370	2,212	104,94
TOTAL			1368,65

Figure 7. Curves obtained with model 3 (specific and global optimizations)

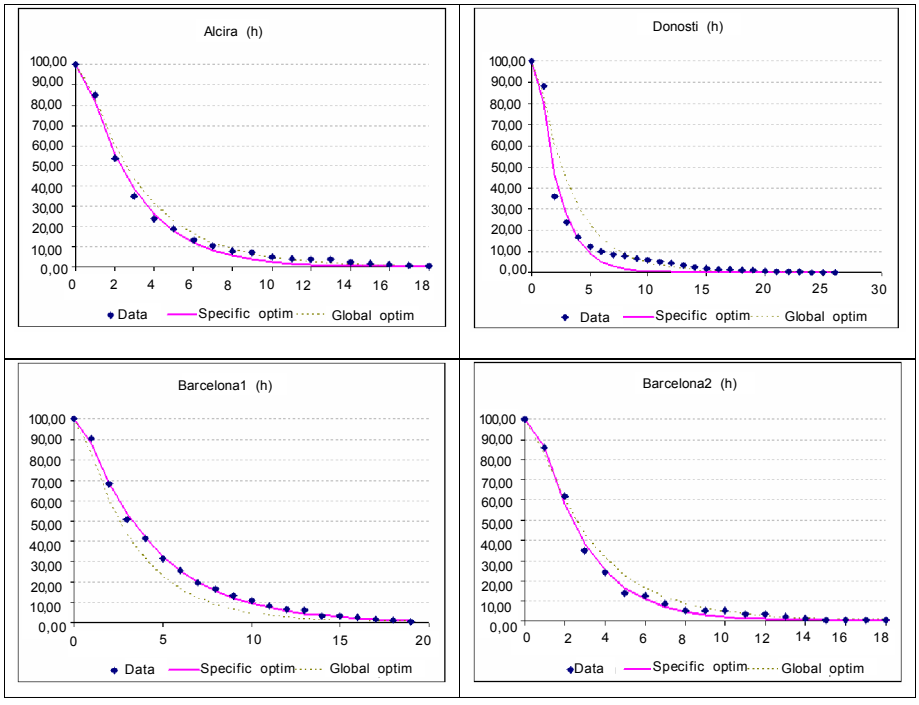


Figure 7. Curves obtained with model 3 (specific and global optimizations) (cont.)

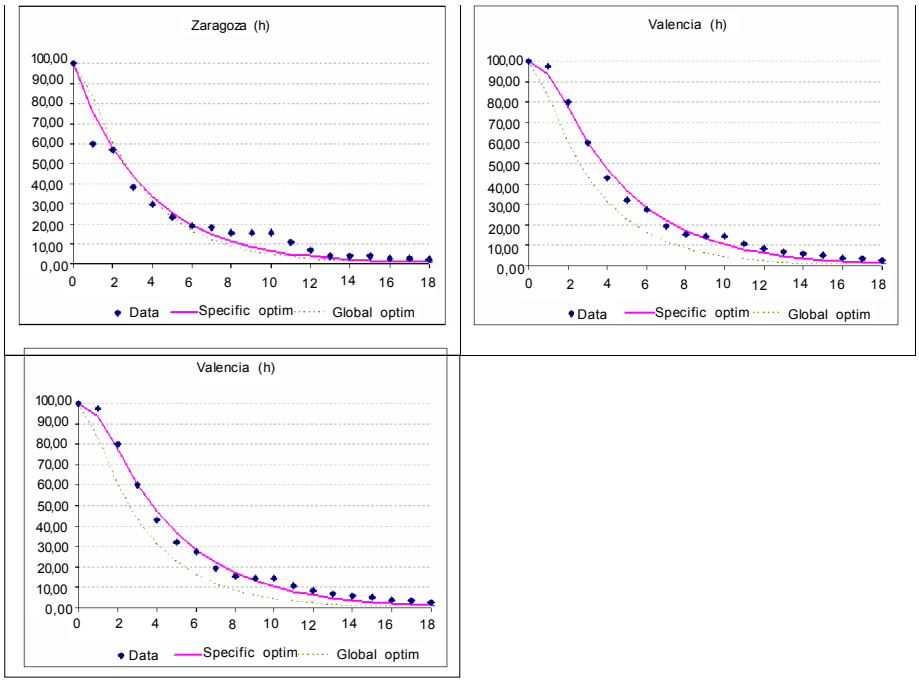
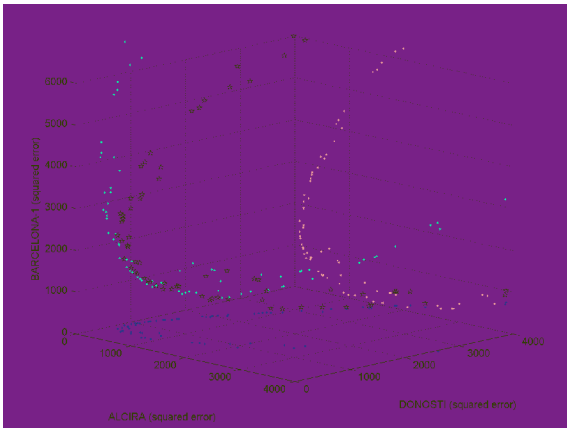


Figure 8. 3D views of Pareto front for model 3 (triads of different regions)



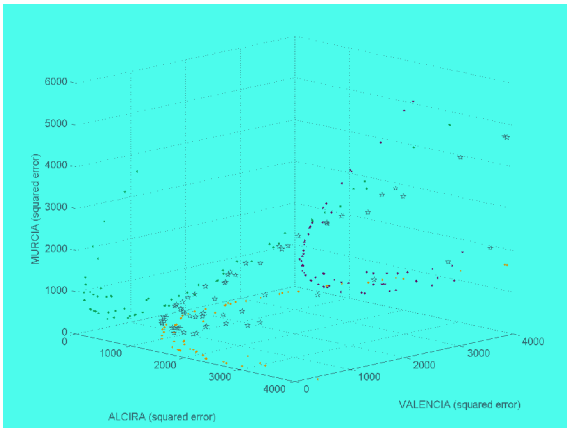


Figure 9. 2D views of Pareto front for model 3 (pairs of first region with the rest)

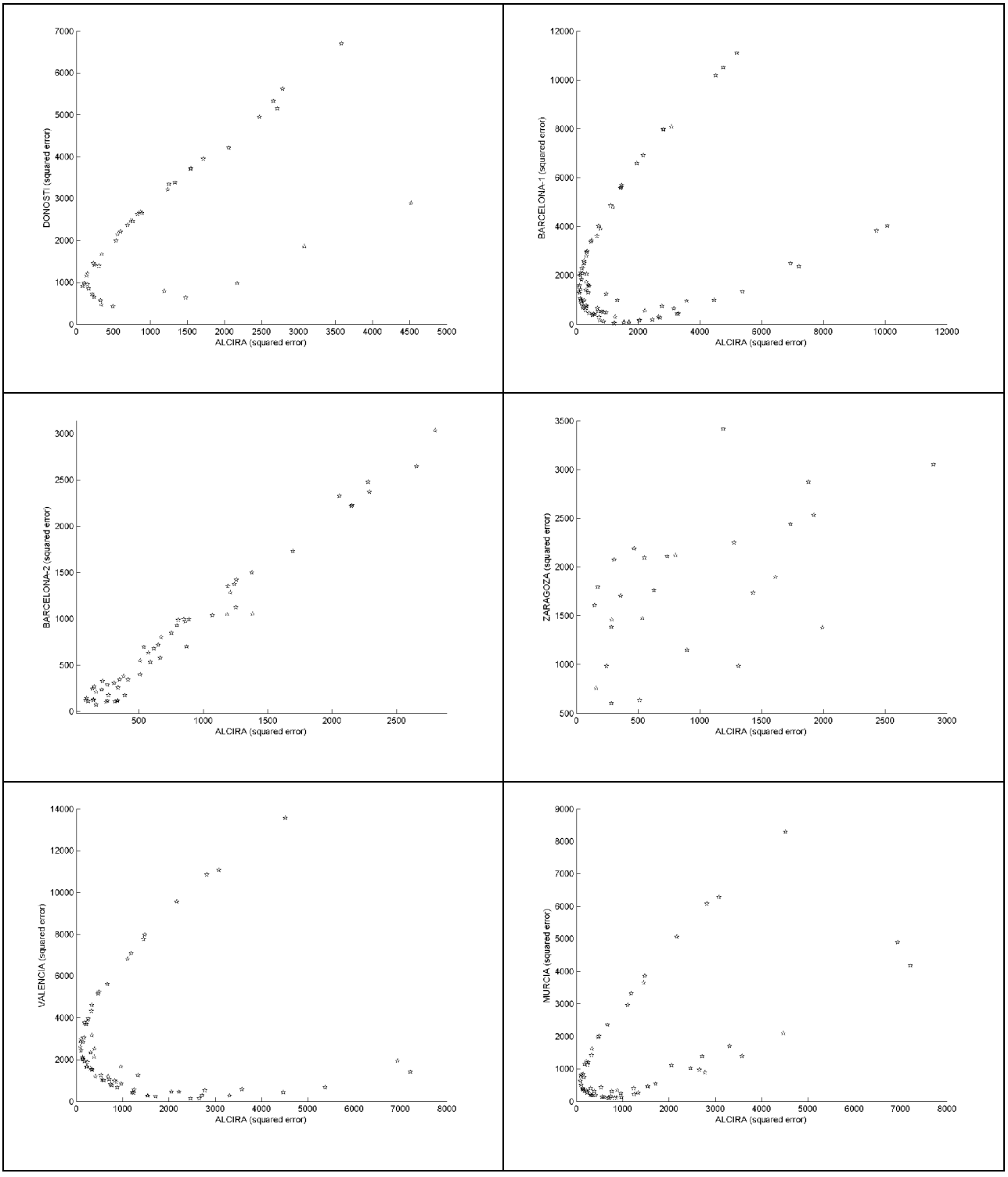


Table 12. SPEA-2 solutions for model 3, optima for specific regions

Series	Alfa	s	Squared Error Sum
Alcira	0,385	0,940	86,439
Donosti	0,515	0,995	439,959
Barcelona1	0,266	1,390	49,577
Barcelona2	0,417	1,430	65,450
Zaragoza	0,288	0,171	602,872
Valencia	0,279	2,680	176,931
Murcia	0,417	2,680	129,313
TOTAL			1550,540

Table 13. SPEA-2 solutions for model 3, optimum for total sum

Series	Alfa	s	Squared Error Sum
Alcira			205,77
Donosti			1.385,81
Barcelona1			735,85
Barcelona2			253,81
Zaragoza			979,84
Valencia			1.739,92
Murcia			342,52
TOTAL	0.32384	0.84405	5.643,53

the accumulative quadratic error. Evolutionary computation techniques optimize the parameter values of different possible models in order to adjust them to a real data distribution. Models represent different distributions that explain the behavior of reported claims after a catastrophe occurs. Besides, the application of multi-objective evolutionary algorithms allows obtaining a full description of possible solutions for this fitting problem, containing all particular cases that could be approached after individual analysis for data subsets. This capability is important for these types of applications, where there are not appropriate global parameters to explain the whole set of data, but specific parameters are needed to describe subsets corresponding to the same model under different situations.

ENDNOTES

[1] For we obtain the differential equation that describes the incurred but not reported claim amount in a deterministic model.

[2] These data are available from the authors upon request.

REFERENCES

Alegre, A., Devolder P., & Pérez, M. J. (2003). Modèle discret d'options sur risques catastrophiques. *Belgian Actuarial Bulletin, 3*.

Becerra, R. L., & Coello, C.A. (2005). A cultural algorithm for solving the job-shop scheduling problem. In Y. Jin (Ed.), *Knowledge incorporation in evolutionary computation* (pp. 37-55). Berlin: Springer-Verlag (Studies in Fuzziness and Soft Computing, 167).

Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. (2001). PESA-II: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)* (pp. 283-290). San Francisco: Morgan Kaufmann.

Corne, D. W., Knowles, J. D., & Oates, M. J. (2000). The Pareto envelope-based selection algorithm for multiobjective optimization. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, et al. (Eds.), *Parallel problem solving from nature—PPSN VI* (pp. 839-848). Berlin: Springer-Verlag.

Cox, S., & Pedersen, H. (1997). *Catastrophe risk bonds.* Working Paper Series No. 97(4),

Center for Risk Management and Insurance Research, Georgia State University, USA.

Cummins, D., & Geman, H. (1995). Pricing catastrophe insurance futures and call spreads: An arbitrage approach. *Actuarial Approach for Financial Risk, Proceedings of the 5th AFIR International Congress* (Vol. 3, pp.45-80), Bruxelles.

Deb, K. (1995). *Optimization for engineering design: Algorithms and examples*. New Delhi: Prentice-Hall.

Deb, K. (1999). Evolutionary algorithms for multi-criterion optimization in engineering design. In *Evolutionary algorithms in engineering and computer science* (Chapter 8). Chichester, UK: John Willey & Sons.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. New York: John Wiley & Sons.

Deb, K., Pratap, A., Agrawal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182-197.

Fonseca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discusión and generalization. In S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms* (pp. 416-423). San Mateo, CA: Morgan Kauffman.

Geman, H., & Yor, M. (1997). Stochastic time changes in catastrophe option pricing. *Insurance: Mathematics and Economics, 21,* 185-193.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Horn, J., Nafpliotsis, N., & Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Computation* (Vol. 1, pp. 82-87). Piscataway, NJ: IEEE Press.

Knowles, J. D., & Corne, D. W. (1999). The Pareto archived evolution strategy: A new baseline algorithm for Pareto multiobjective optimization. *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)* (Vol. 1, pp. 98-105).

Loubergé, H., Kellezi, E., & Gilli, M. (1999). Using catastrophe-linked securities to diversify insurance risk: A financial analysis of CAT Bonds. *Journal of Insurance Issues, 22,* 125-146.

Mueller, P. (2002). *Stock market reactions to the issue of CAT Bonds*. Master's thesis, Department of Banking and Finance, University of Lausanne, France.

Muermann, A. (2003, July). *Actuarial consistent valuation of catastrophe derivatives*. Working Paper 03-18, The Wharton School, University of Pennsylvania, USA.

Ohkura, K., & Matsumura, Y. (2001). Robust evolution strategies. *Applied Intelligence, 15,* 153-169.

Schwefel, H. P. (1981). *Numerical optimization of computer models*. New York: John Wiley & Sons.

Shaffer, J. H. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. *Proceedings of the 1st Interna-*

tional Conference on Genetic Algorithms (pp. 93-100).

Sigma. (2001). Capital market innovation in the insurance industry. *Swiss Re, 3*.

Sigma. (2003). The picture of ART. *Swiss Re, 1*.

Sigma. (2005). Natural catastrophes and man-made disaster in 2004: More than 300 000 fatalities, record insured losses. *Swiss Re, 1*.

Srinivas, N., & Deb K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221-248.

Tan, K. C., Khor, E. F., & Lee, T. H. (2005). *Multiobjective evolutionary algorithms and applications*. London: Springer-Verlag.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67-82.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation, 8*(2), 173-195.

Zitzler, E., & Thiele, L. (1998). *An evolutionary algorithm for multiobjective optimization: The strength Pareto approach.* Technical Report, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Switzerland.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257-271.

Zitzler, E., Laumanns, M., & Thiele, L. (2001, May). *SPEA2: Improving the strength Pareto evolutionary algorithm.* Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Switzerland.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In K. Giannakoglou, D. Tsahalis, J. Periaux, K. Papailiou, & T. Fogarty (Eds.), *Evolutionary Methods for Design, Optimization, and Control* (pp. 95-100). Barcelona: CIMNE.

Chapter LVIII
Modeling an Artificial Stock Market

Stéphanie Lavigne
Toulouse Business School, France

Stéphane Sanchez
Université Toulouse 1, France

ABSTRACT

This chapter presents an artificial stock market created to analyze market dynamics from the behavior of investors. It argues that information—delivered by financial intermediaries as rating agencies and considered as cognitive institution—directs the decisions of investors who are heterogeneous agents endowed with capabilities of learning in a changing environment. The objective is to demonstrate that information influences market dynamics as it allows the coordination of the decisions of investment in the same direction: information is a focal point for investors and contributes to generate a speculative dynamic on the market.

INTRODUCTION

Since their creation, stock markets have always experienced an alternation of phases of financial bubbles and crashes, the last important one being the bubble observed on the American and European stock markets between 1990 and 2000. The aim of the chapter is precisely to focus on the dynamic of prices to try to understand the phenomenon of financial bubble on stock markets.

The purpose is in fact to consider the emergence of a dynamic of stock prices as the result of investors' behaviors. On the one hand, we refer to the works that analyze economy as an *evolving complex system* (Arthur, Holland, LeBaron, Palmer, & Tayler, 1997): a market is a complex system in permanent evolution, whose product is explained by the numerous processes of interactions between its components. On the other hand, we consider market dynamics and stock prices as the consequence of the confrontation and evolution of investors' heterogeneous representations (Keynes, 1936; Kirman, 1997, 1999; Orléan, 1999; Tordjman, 1997).

To analyze the decisions of investors and their opinions about market dynamics, we propose an artificial model created thanks to the techniques of data-processing simulation, and more precisely to classifiers systems. These techniques allow apprehending an emergent dynamic as the result of the interactions between the decisions of investors composing the market.

Our model refers to the work of Arthur et al. (1997), known as the Santa Fe Artificial Stock Market, in which investors have to optimize the allocation of their portfolios between two kinds of assets: a stock and a bond. In this model, investors: (1) are characterized by a function of utility, (2) do not communicate, (3) form expectations according to a self-referential process, and (4) test different models of representation selecting only the ones that have generated profits in the past. The results of Arthur et al. (1997) notably demonstrate that when investors test few models, the market finally reaches an equilibrium state, whereas the market is characterized by a more complex dynamic when they explore several models of representation. Our artificial model differs from the work of Arthur et al. (1997) in the sense that: (1) investors do not optimize the value of their portfolios but are characterized by a "satisficing" behavior (Simon, 1979); (2) it allows testing the influence of institutions on financial markets and more precisely the impact of different kind of information on investors' representations; and (3) from a technical point of view, it is innovative because the model introduces two classifier systems as decision engines for each artificial investor.

Our work is original in the sense that it allows understanding behaviors according to the nature of the informational signals provided to investors. We consider two kinds of information: an *endogenous* one, representing information produced in the financial sphere and resulting from the interactions between investors; and an *exogenous* one, indicating information produced outside the market by financial intermediaries such as rating agencies or financial analysts. Thanks to this typology of information, we obtain two kinds of results. In simulations with endogenous signal, investors manage to coordinate their decisions: the stock market quickly converges towards a stationary state, whereas when investors dispose of exogenous information, the market is not characterized any more by a stable state but by the emergence of a financial bubble.

Finally, the model highlights the power of information in the orientation of market dynamics. We demonstrate that information polarizes the anticipations formed by investors with different representations on their environment: information directs the evolution of the market in the sense that it generates a financial bubble and is a framework for individual action.

A JUSTIFICATION OF THE ARTIFICIAL MODEL

The objective consists of modeling the emergence of a collective dynamic on the stock market starting from an analysis of investors' decisions. Data-processing simulation is mobilized as it allows analyzing the decision-making process of agents in situations of uncertainty by considering that they do not make optimal decisions but adapt their decisions to the changes of their environment. Generally, simulation is a relevant tools as soon as the market is characterized by an alternation of phases—in other words, when it is a question of analyzing a dynamic and an open system (Lane, 1993). Simulation allows studying a global dynamic without providing an analytical representation of it, and accounts for the emergence of a global phenomenon and for the adaptation of agents to this phenomenon.

One of the arguments often advanced in favor of a data-processing simulation is that market is composed of many investors who make their decisions simultaneously and whose networks of interactions are too numerous to be studied in an analytical way (Tordjman, 1997). Another justification, undoubtedly more global, supports that this technique is adapted to any kind of study in which the environment of agents is complex, in the sense that the information available exceeds the processing capacity of agents (Moss & Rae, 1992). Finally, simulation must be considered as a useful instrument to improve our analysis of phenomena of "order" or "disorder" on a market (Lesourne, 1991).

The artificial stock market is built thanks to classifier systems that are well adapted to the analysis of the emergence of a complex dynamic. Classifier systems recognize the importance of agents' cognition to explain their behaviors and market dynamics: investors evolve in a changing environment and are endowed with representations of their environment as well as capabilities of learning. Their representations are not fixed *ex-ante,* but are emergent and change over the course of time thanks to the process of adjustment of investors to the evolution of their environment. Learning is considered as the driving principle of the evolution of the market. Learning is primarily: (1) *inductive* because investors observe the sequence of their last decisions and try to find some regularities in stock price dynamics to build a general law (Arthur, 1994); (2) *path-dependent,* having resulted from the history of the interactions (David, 1985); and (3) *cumulative* because investors learn how to replay rules that have procured them satisfaction in the past (David, 1985; Arthur, 1994; Kirman, 1999; Tordjman, 1997). This conception of learning supposes that investors have a memory to record their past results and criteria of judgment to select a rule of investment (Paulré, 1997).

The period of learning is divided in two phases: an exploration phase during which investors try to discover profits related to each rule and another, during which they tend to replay actions which have generated profits. The rationality of investors enables them to summarize past profits and to create an index of profit for each activated rule, knowing that they tend to reinforce decisions that have already generated profit. In fact, investors' preferences are expressed through an index that synthesizes their last experiments.

But to which kind of rationality do we precisely refer to model investors? We postulate that their rationality is limited in the sense that each agent: (1) does not know the whole options that are offered to him (informational limits), and (2) is not able to evaluate all the consequences of his choices (computational limits) (Simon, 1979). Investors do not select rules that could allow them to maximize their portfolios' value, but more simply try to find a rule that could increase this value. Investors are also endowed with a cognitive rationality: they must carry out adequacy between the information they possess and their representations on their environment and on themselves (Lesourne, Orléan, & Walliser, 2002).

MICROSTRUCTURE OF THE MARKET

We analyze market dynamics considering stock prices as the result of the confrontation between the personal opinions of investors (Keynes, 1936; Orléan, 1999). More precisely, prices represent the average opinion of the investors' community[1] which means that investors account of the others and of their evolving environment to make their decisions (Keynes, 1936). While trying to guess the future average opinion to beat the market, inves-

tors can lead prices to excessive levels in comparison with real economic data.

Our stock market is a system in which it is possible to observe phenomena of self-organization: we recognize the capacity of the market to produce a structure, a behavior, and/or its own rules of functioning (Lesourne, 1991; Paulré, 1997). Self-organization is apprehended on two levels: first, it indicates the capacity of the market to produce some rules of investment. Then, it supposes that investors are able to adapt their behaviors to their changing environment. In fact, each investor transforms and structures the environment of the others, and market dynamics are open.

Investors and Firms

The artificial market consists of 20 investors that must allocate the amount of their portfolios (composed of liquidities and stocks) on 10 firms. The latter are different by their stock exchange capitalization: the number of stocks and stock prices are different for each firm. There is neither creation, nor destruction of firms and stocks during the simulations.

Besides, there are no transaction costs on the market, and information is free and accessible to all investors without discrimination.

To characterize each firm, we create an indicator that can be compared to a ranking of firms delivered by a rating agency. This indicator, called *expected percentage of growth (EPG),* attests the possibilities of real growth of the firm: it is information on the fundamental value of the firm. The rating agency diffuses a grading of the 10 firms according to this indicator, knowing that the EPG indicator can take values ranging between less than 10% and more than 10% of the current stock price. Negative values of the EPG indicator mean unfavorable prospects of growth of stock prices. On the contrary, positive values testify favorable prospects. EPG values are randomly given to each firm at the beginning of simulation, and they reflect information for stock prices for the following periods. The objective is to test the impact of EPG value on investors' strategies and on market dynamics.

To simulate investors and to observe the emergence of various investment strategies based on the "satisficing" postulate (Simon, 1979), two kinds of classifier systems (both based on ZCS; Wilson, 1994) are used to model the behavior of a unique investor. The first one generates decision regarding the purchase of stocks: an investor can decide to buy or to ignore stocks of a firm. The second one generates decisions regarding the sale of owned stocks: an investor can choose to sell or to conserve stocks of a firm. Using two classifier systems instead of a unique one is mainly due to the chosen rewarding strategy. Indeed, we assumed that "purchase" and "sell" rules are not evaluated with the same temporality. Mixing both kinds of classifiers in a unique system may quickly favor rules with instantaneous rewards ("sell" classifiers) over rules with continuous rewards ("purchase" classifiers). On the contrary, separating them avoids that any kind of rule becomes dominant and breaks convergence dynamics of the system. The two classifier systems are independent: they simply each produce the best investment order (i.e., the one that will generate more profit). Their necessary coordination emerges from the specific functioning scheme of classifier systems that detect, classify, and generalize environmental situations (the market configuration), from the arbitration of the auctioneer, and from the way it handles transactions between investors.

In addition to our specific rewarding scheme, we choose not to use a genetic algorithm to generate new rules of investment. Indeed, our goal is not to optimize a set of rules, but to

generate the emergence of a variety of good and bad strategies among investors. We do not obtain only investors that always make the good choice. Finally, we introduce covering in order to avoid situations where investors cannot provide an order.

The condition part of both kinds of classifier systems is mostly similar: it consists of a concatenation of exogenous and endogenous signals which can produce an effect on the behavior or on the cognitive state of an investor. Endogenous signals—which represent information produced by financial actors of the market—are (1) the historical data for stock prices of a firm over the last 20, last 10, and last two periods; (2) the configuration of stocks at the preceding period—do stocks have a tendency to be purchased or to be sold?; and (iii-a) the number of stocks investors can purchase considering their liquidities—"purchase" classifier system, or (3-b) the percentage of owned stocks of a firm—"sell" classifier system. The only exogenous data is the current EPG value. The historical data enable the postulation of the existence of "chartists" behaviors as well as the role of memory: do investors hold account of the past data to make their decisions? Do they have a short-term or a long-term memory?

The action parts of both classifier systems consist of the next investment order, a percentage of stocks to buy or to sell, and a purchase or sale price. To avoid a tendency to high speculation, we create an institutional rule: for their purchase orders, investors can propose a purchase price ranging between 0 and 10% of the current price; concerning their sale orders, they can propose a price ranging between 0 and less than 10% of the current price. This institutional rule validates the thesis of the organizing function of the market since we recognize that institutional rules are a framework for individual action.

Transactions on the Artificial Market

At the beginning of each simulation, the artificial market is created out of nothing with fictitious stock prices: prices and the allocation of portfolios on the various stocks are randomly generated.

A simulation is a succession of periods of transactions that reproduce the operation of the notebook of orders before the opening of the stock exchange: all investors make their decision in a simultaneous way, and transmit their purchase and sale orders for each firm to an auctioneer.

In real financial markets, to pass a stock exchange order, it is important to know the notebook of orders which indicates, at one given moment, the state of supply and demand on a value to produce an equilibrium price, and the price being determined to maximize the number of transactions on the market. Here, the auctioneer counts the orders, eliminates 'invalid' orders, and treats orders firm by firm in a random way. A purchase order will be accepted if investors have sufficient liquidities to buy stocks and if they manage to find a counterpart. A purchasing order will have priority on the other purchasing orders if it proposes the highest purchase price. Conversely, a sale order will have priority on the others if it proposes the weakest selling price. All orders cannot be carried out in the market, in particular when investors do not have sufficient liquidities at their disposal or do not find a counterpart.2 In conformity with the treatment of orders on real financial markets, orders are only valid on the day where they are emitted, in other words for only one period. Some investors can be 'rationed' over one period. But at the end of each period, orders that are not carried out are withdrawn from the notebook. Finally, at each

Table 1. Rewarding system

	Simulation without EPG	Simulation with EPG
Purchase	$G_{ij} = \sum_{k=i+1}^{j}(p_{i2} - p_{i1}) \times NO_k + \sum_{k=i+1}^{j}(p_{k2} - p_{k1}) \times NS_k$	$G_{ij} = p_{i1} \times EPG_i \times NP_i + \sum_{k=i+1}^{j}((p_{k2} - p_{k1}) \times NO_k + p_{k1} \times EPG_k \times NO_k) + \sum_{k=i+1}^{j}(p_{k2} - p_{k1}) \times NS_k$
Ignore	$G_i = -1 \times (p_{i2} - p_{i1}) \times N$	$G_i = -1 \times (p_{i2} - p_{i1}) \times N - p_{i1} \times EPG_i \times N$
Sell	$G_i = (p_{i2} - p_{i1}) \times NS_i$	$G_i = (p_{i2} - p_{i1}) \times NS_i - p_{i1} \times EPG_i \times NS_i$
Conserve	$G_i = (p_{i2} - p_{i1}) \times NO_i$	$G_i = (p_{i2} - p_{i1}) \times NO_i + p_{i1} \times EPG_i \times NO_i$

period, the notebook of orders is automatically set to zero (market clearing).

Between two periods, portfolios of investors are evaluated considering the evolution of stock prices. The generated rewards will be distributed to active classifiers.

The market is "built" by investors since market dynamics are the result of their interactions: the behavior of others is important for the decision making. Investors are not isolated calculators—their choices of investment relate to the choices and behaviors of others. However, this method of evaluation does not mean for all that investors only decide according to the opinion of the others. They can also decide to invest according to external information delivered by the rating agency: the EPG value.

The Rewarding System

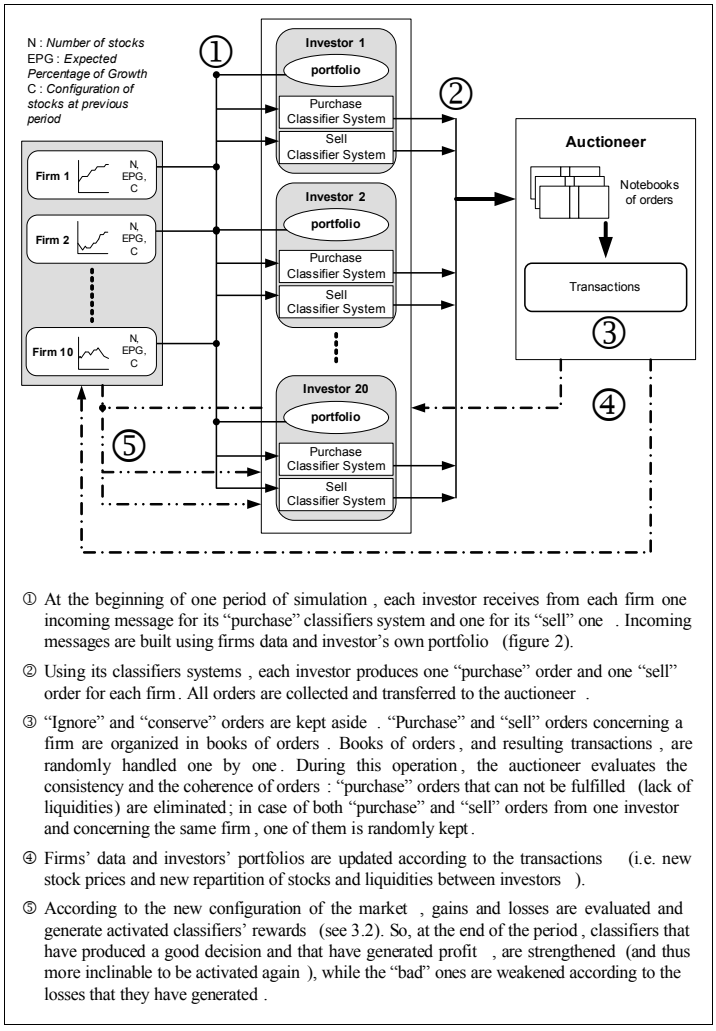
A classifier receives a reward based on the profit that its activated order has generated. Regarding the "ignore," "conserve," and "sell" orders, rewards are immediate: they represent immediate gain or loss of portfolio value[3] during the last transactions. On the contrary, a "purchase" order is never remunerated at the moment of the transaction, for when one investor purchases a stock, he only transforms his liquidities into stocks. The classifier is remunerated by the profits or losses that acquired stocks generate from the moment of their purchase to the moment of their sale.

The equations of Table 1 show calculation of rewards in both kinds of simulations: the simulations without influence of the exogenous data and the ones where investors take EPG values into account. In this last kind of simulation, the EPG parameter does not have a real influence on the stock price of a firm, so investors cannot learn automatically how to use it. To compensate for this, we choose to incorporate this parameter in reward calculation, making the assumption that the information provided by the rating agency is always correct.

For each period T_i of the simulation and for each firm, we consider the following data:

- p_{i1}: stock price at T_i before the transaction (i.e., stock price at the time of decision-making);
- p_{i2}: stock price at T_i after the (i.e., new emergent price on the market);
- G_i: profit for the classifier involved in the decision of investment for T_i;

Figure 1. The artificial stock market



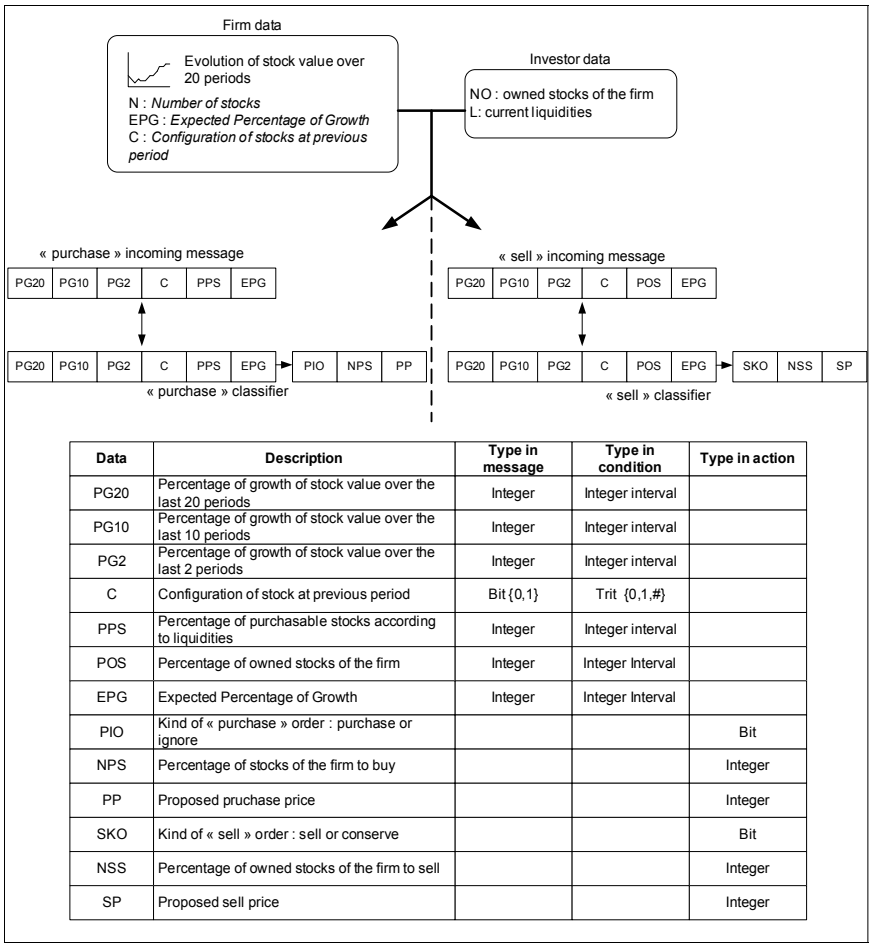
- G_{ij}: profit for the rule of investment from T_i to period T_j;
- N: number of stocks of a firm;
- NP_i: number of purchased stocks at T_i;
- NO_i: number of owned stocks T_i;
- NS_i: number of sold stocks T_i; and
- EPG_i: EPG value at T_i.

Figures 1 and 2 summarize the functioning of our artificial market and present the description of the two classifier systems.

SIMULATIONS AND RESULTS

We present the results of two kinds of simulations to understand the emergence of particular market dynamics. The first kind designs cases in which investors have only endogenous informational signals, whereas the second type includes information delivered by the rating agency. For each simulation, we analyze market dynamics thanks to a stock index.[4] This index represents the evolution of firms' capi-

Figure 2. Structures of messages and classifiers (investment rules) of both kind classifiers systems ('purchase' and 'sell' systems)



Data	Description	Type in message	Type in condition	Type in action
PG20	Percentage of growth of stock value over the last 20 periods	Integer	Integer interval	
PG10	Percentage of growth of stock value over the last 10 periods	Integer	Integer interval	
PG2	Percentage of growth of stock value over the last 2 periods	Integer	Integer interval	
C	Configuration of stock at previous period	Bit {0,1}	Trit {0,1,#}	
PPS	Percentage of purchasable stocks according to liquidities	Integer	Integer interval	
POS	Percentage of owned stocks of the firm	Integer	Integer Interval	
EPG	Expected Percentage of Growth	Integer	Integer Interval	
PIO	Kind of « purchase » order : purchase or ignore			Bit
NPS	Percentage of stocks of the firm to buy			Integer
PP	Proposed pruchase price			Integer
SKO	Kind of « sell » order : sell or conserve			Bit
NSS	Percentage of owned stocks of the firm to sell			Integer
SP	Proposed sell price			Integer

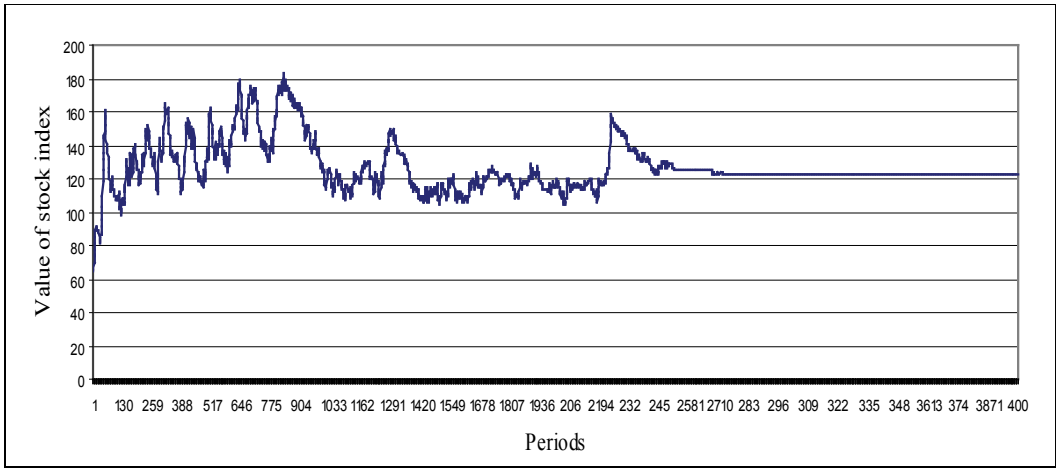
talization and allows comparing the global performance of the market to the performance of the different firms: do investors reproduce the performances of the market, or under-perform or out-perform the market? We also propose a typology of investors according to their performances of management (do their portfolio values increase or decrease) and according to their strategies of investment (which rules of investment do they use).

Each simulation is carried out over 4,000 periods, one period being theoretically comparable to one day of pre-opening to the stock exchange.

Coordination of the Decisions of Investment around a Stationary State

The results of simulation, in which investors have only endogenous signals, look like the results of the Efficient Market Theory, according to which the market converges towards a state of equilibrium (Fama, 1970). This state of equilibrium means that prices contain all information available on the market: prices allow investors to make their decisions without having a perfect knowledge of the global functioning of the market. But how is it possible to

Figure 3. Stock index evolution



justify the coordination of investors in the same direction in our artificial model?

Market Dynamics: Towards a Stationary State

The analysis of the stock index reveals that it grows on average of 90% on the whole period for all the simulations carried out. This progression is not continuous: the market always goes through a phase of strong growth, then through a decrease of firms' capitalization, to finally stabilize itself after the period of learning. More than to stabilize itself, the market balances between the period *t=2000* and the period *t=3000* for the whole simulations carried out. Investors manage to coordinate their behaviors without having an identical reference for all at the beginning, reference on which they could have coordinated each other. Then how is it possible to justify such a stable state?

We cannot explain this coordination by the fact that investors have an identical idea of the equilibrium market price, for there is no reference, no precedent, no convention at the beginning of each simulation (Schelling, 1960). Therefore, we propose an explanation starting from the process of learning that characterizes investors. At the start, rules are mobilized in a random way by investors. If the first emergent rules lead to an increase in the portfolio's value, these rules have a strong probability of being selected again, in accordance with the functioning of classifier systems. Then investors are likely not to explore all possible alternatives because of their limited computational capabilities. And if all behave in the same way, each one persisting in the exploitation of the first emergent rules that have generated profit, they can succeed in coordinating with each other, and prices finally stabilize after the period of learning: "*...the place where the price stabilizes is random, it depends on the direction of the first 'good' rules which emerge*" (Tordjman, 1997, p. 891). Finally, coordination of the decisions towards a stable state is justified by the fact that investors persist in the exploitation of the first emergent rules and do not explore many alternatives (Holland, 1975).[5]

Typology of Investors

An analysis of investors' strategies reveals three recurring kinds of behavior of investment.

Type 1 includes investors whose performances are poor and negative, although these

Figure 4. Portfolio of investors belonging to TYPE 1

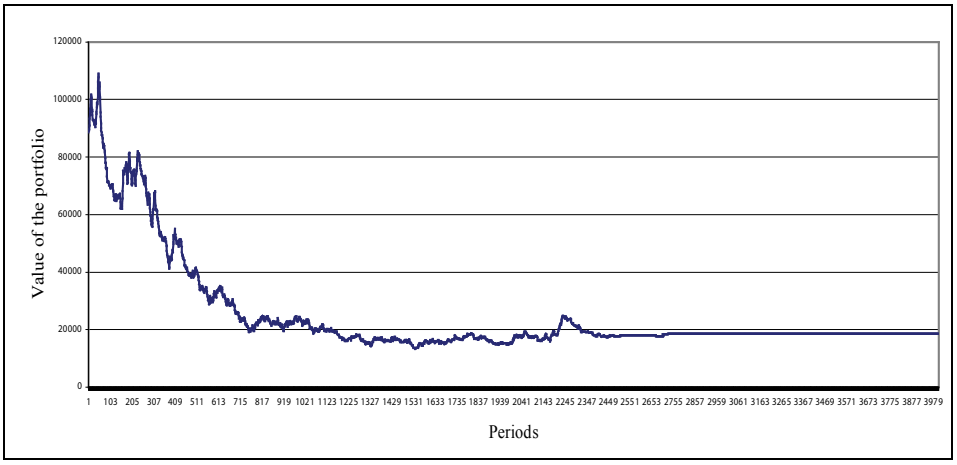
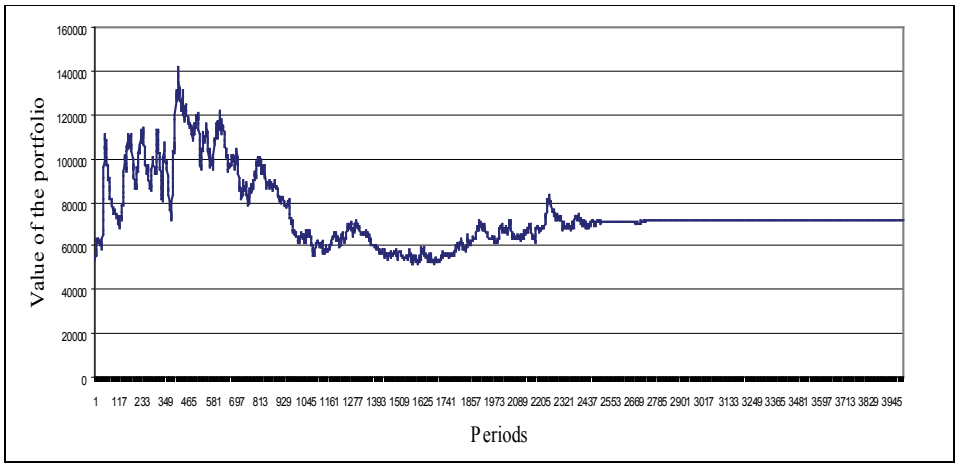


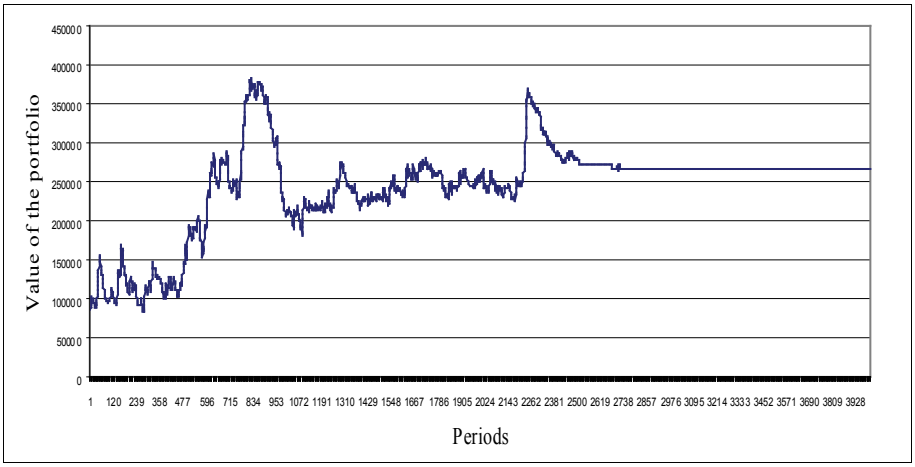
Figure 5. Portfolio of investors belonging to TYPE 2



investors manage the largest portfolios at the start.6 These investors have a very aggressive behavior: at the beginning, they purchase stocks until exhaustion of their liquidities, then they massively sell stocks before stabilizing their portfolios at the end of the period. They buy whatever the configuration of the stock on the market is, and they propose sale and purchase prices so that their orders are treated in priority. The rule that generates the highest profit consists of purchasing a stock whatever its configuration. Finally, they are speculators without precise strategy with respect to prices, or with respect to the others. They behave as isolated calculators and only seek short-term profits by adopting very aggressive positions: they continuously purchase and sell stocks.

Type 2 includes investors whose performances are positive while remaining lower than the progression of the stock index. These investors purchase and sell stocks in a very frequent way, but they differ from the first type by the adoption of specific rules: they always purchase a small quantity of stocks and they sell stocks in a more massive way. They generally buy when stocks have a tendency to be sold—

Figure 6. Portfolio of investors belonging to TYPE 3



in other words when a potential offer exists on the market—and they sell whatever the stocks' configuration. These investors are strategic calculators in the sense that they learn how to split up their orders, and they take account of the existence of an offer on the market to make their decisions. So, they do not behave as pure isolated calculators.

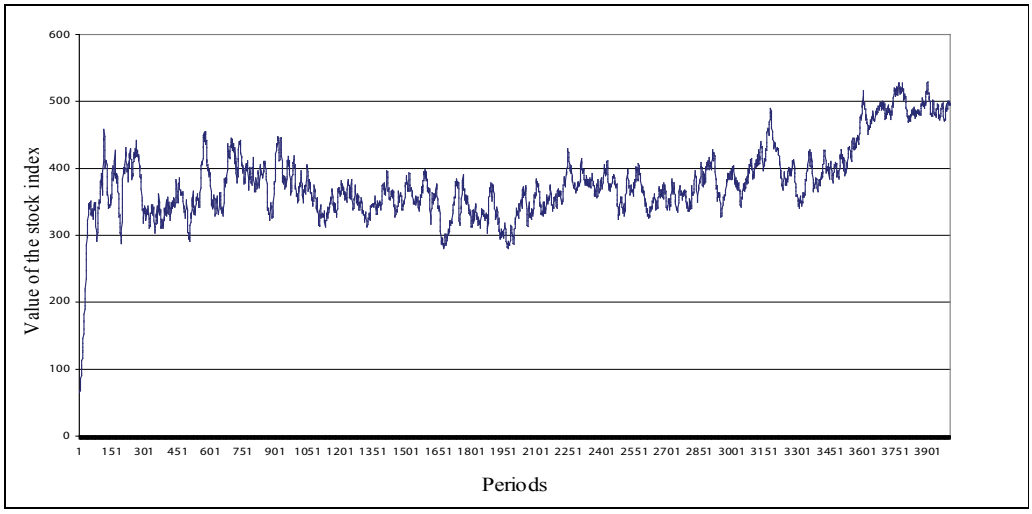
Type 3 gathers investors whose performances are positive and much higher than the progression of the stock index. They manage to out-perform the average performance of the market. They purchase stocks when they have a tendency to be sold at the preceding period and sell stocks whatever their configuration. Contrary to investors belonging to type 2, they are characterized by behaviors very targeted on certain firms—that is, by a strong concentration of their portfolios on a limited number of firms. The rule that generates the highest profit for this group consists of conserving stocks whatever their configuration. These investors combine a short-term behavior characterized by rapid movements of purchases and sales on some firms, and a medium-term behavior as their stable holdings on a targeted number of firms attest it. As the overview of these investors is not only the short term, we qualify them as "careful" investors.

This first kind of simulation, in which investors have only endogenous signals, enables us to find stylized facts observed on real financial markets, namely: (1) aggressive investors behaving as isolated calculators and being not averse to risk; (2) investors taking account of the others (strategic calculators); and (3) careful investors adopting complex strategies combining stable holdings on some firms and frequent movement of purchases and sales on others. However, this simulation proposes results that are only valid for the case of a pure speculation market. Indeed, investors make their decisions from informational signals that are the product of their decisions: there is no information on the fundamental value of firms.

When Exogenous Information Generates a Financial Bubble

In the second kind of simulation, investors dispose of two kinds of signals to make a decision: endogenous information and a signal delivered by a rating agency. In this case, we demonstrate that the market is self-organized according to a more complex dynamic. Initially, the market is characterized by an alternation of phases of financial bubbles and crashes, then

Figure 7. Stock index evolution



by the persistence of a financial bubble. We justify this dynamic by the presence of the exogenous information. But how does the EPG signal precisely operate on the decisions of investors and on market dynamics?

Emergence and Persistence of a Speculative Dynamic

Our analyses reveal that the stock index grows on average 600% over the period for all the simulations to be carried out. The market dynamics are *open-ended* since they are characterized by a continuous growth of stock prices, in other words by a phase of financial bubble. Contrary to the first type of simulation, it is not possible to observe any stabilization or coordination of investors' decisions on a particular point.

However, the progression of the stock index does not reflect the evolution of the capitalization of the various firms. Firms whose EPG value is negative or null (i.e., firms with a bad ranking) are those that carry out the worst performances with regard to the progression of their capitalization. Besides, investors have short-term holdings on these firms since their purchase and sale orders are very frequent. On the other hand, firms whose EPG value is positive (i.e., firms with a good ranking) are those that obtain higher performances than the progression of the stock index. Investors have medium-term or even long-term holdings on these firms.

These results tend to demonstrate that investors take stable and important holdings in firms with positive EPG value for they have discovered, after the period of learning, that investing in these firms was rewarded. Finally, investors who adopt behaviors in conformity with the information delivered by the rating agency are those who carry out the best performances.

These results, largely inductive, validate the idea that the rating agency has a real capacity on the orientation and the evolution of the system: it allows the coordination of investors on some firms and generates a speculative dynamic. But how is it possible to justify this mechanism of coordination?

To make their decisions, investors formed representations which they successively test, knowing that they use EPG signal to forge expectations on the stocks' prospects of growth.

If all investors take account of the EPG signal,[7] they will all decide to invest in firms with positive EPG value. This behavior will mechanically cause an increase in the stock price, the supply of stocks becoming higher to the demand. Carrying out profits on firms with a strong EPG value thanks to plus-values, the rule will be rewarded and considered as a 'good' rule. So investors will continue to mobilize this rule by a process of self-reinforcement, and this behavior will generate a new increase of stock prices. Finally, there is a self-realization of the prospects of growth announced by the rating agency: while following the exogenous information and while investing in the 'recommended' firms, investors generate an increase of those firms' capitalization. Market dynamics that result from this mechanism of learning by "self-reinforcement" of the good rules are not characterized by a coordination of investors on a stable state, but rather by a dynamic of financial bubble.

Typology of Investors

The analysis of the rules selected by investors reveals again three recurring kinds of behavior.

Type 1 includes, as previously, investors who manage at the start the largest portfolios and whose performances are, however, weak and negative. They speculate in an important way on all stocks and do not have any stable holdings. We cannot advance any particular factor to determine their strategy: they take account of neither the configuration of the stocks, the trend of stock prices, nor the EPG indicator. The EPG signal is not a determining factor for the decision of investment. Finally, these investors are short-sighted speculators behaving as isolated calculators, without any directing principle relating to the decision of investment. Their orders are massive and frequent on all stocks.

Type 2 includes investors whose performances are positive while remaining lower than the progression of the stock index. Contrary to the preceding group of investors, we highlight particular criteria of investment. First, the EPG indicator is a determining factor: investors have stable holdings in firms with positive EPG value, and they speculate in a permanent way in firms whose EPG value is negative or null. At the start, investors adopt a short-term strategy, and after the period of learning, they adopt medium-term behavior on firms recommended by the rating agency. At the end of each simulation, the rule of investment that generates the highest profit is the one that consists of conserving

Figure 8. Portfolio of investors belonging to type 1

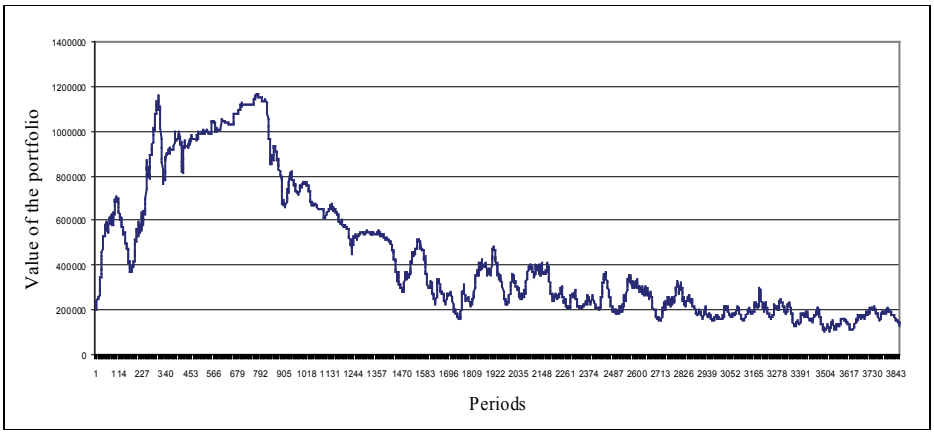


Figure 9. Portfolio of investors belonging to type 2

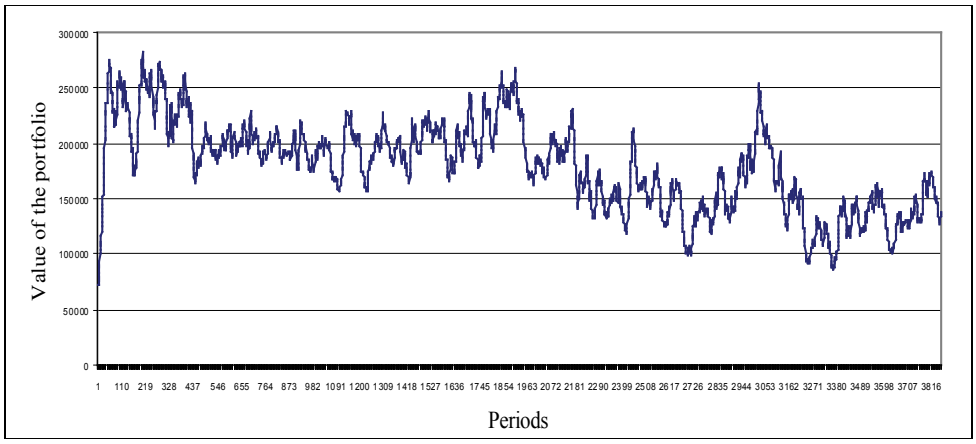
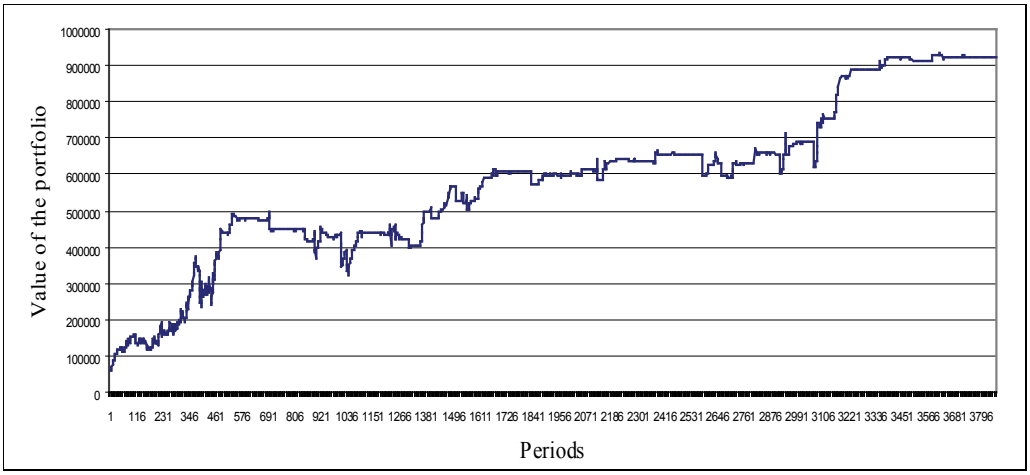


Figure 10. Portfolio of investors belonging to type 3



stocks whose EPG value is positive. Globally, these investors purchase stocks when they have a tendency to be sold and sell stocks when they have a tendency to be purchased at the preceding period. So their behaviors depend on the existence of a potential offer on the market at the previous period. Finally, they do not behave as isolated calculators, but take into account the others and have stable holdings on firms recommended by the agency.

Lastly, ***type 3*** includes investors whose portfolio growth is higher than the one of the stock index. For this group, the EPG indicator is a determining factor of the decision of investment. Globally, these investors have very stable holdings—and even quasi-monopoly behaviors—in firms with positive EPG value and a very aggressive behavior on firms with negative or null EPG value. In other words, they adopt a medium-term strategy on firms recommended by the rating agency and an aggressive behavior on firms with the worst prospects of growth. In a very clear way and concerning their decision on firms with negative EPG value,

these investors purchase stocks when their prices increase, sell stocks when they drop, and conserve stocks when their prices are stable (in accordance with the principle of destabilizing speculation; De Long, Shleifer, Summers, & Waldmann, 1990). Generally, this group of investors learns to propose a very high purchase price—near 10% of the current price—so that their orders are treated in a priority way. Lastly, the rule that generates the highest profit at the end of the period of learning is the decision to conserve stocks whose EPG value is positive. Finally, investors of type 3 manage to outperform the market by adopting a complex strategy: (1) they consider the other agents through the recognition of a potential offer on the market; (2) they conform to the information delivered by the rating agency to decide to invest in such or such firm; and (3) they adopt a careful strategy that includes conserving stocks whose EPG value is positive—that is, stocks recommended by the rating agency.

CONCLUSION

The main interest of this model is to register market dynamics from the point of view of cognitive economics which grants a central role to investors' representations and to their processes of interactions. It is a question of analyzing their way of reasoning in a situation of uncertainty through the formation of their representations, symbolized by the rules of investment. The purpose is especially to understand the emergence of a global dynamic on the market, starting from the interactions of dispersed investors who are endowed with capabilities of learning. Investors are able to adapt to the evolution of their environment that evolves under the effect of their own decisions of investment.

The model allows the prediction of several elements for the potential dynamic of the market and makes it possible to find stylized facts of financial markets.

The model brings light to the question of the coordination of the decisions of heterogeneous and decentralized agents. Indeed, it allows partly answering an old question: how an economic order can emerge, starting from the interactions of investors motivated by the search for their personal interests—the increase of their portfolios' values. We demonstrate that when investors have only endogenous informational signals, which are the result of their interactions, they succeed in coordinating each other on a stable state. This result, all the more surprising as there is no precedent or no common reference enabling them to coordinate quickly, has been explained by the process of learning and by the functioning of classifier systems.

The artificial model highlights the power of a cognitive institution—the information delivered by the rating agency and represented by the EPG signal—in the orientation of market dynamics. This information has the power to direct investors' representations in the same direction. The rating agency has a capacity of influence since it contributes to the generation of a speculative dynamic. Indeed, when they take into account the EPG indicator and invest in firms recommended by the rating agency, investors cause mechanically, by their decisions, a rise of the stock prices: there is a self-realization of the prospects of growth announced by the agency.

In fact, the artificial model enables us to postulate that information delivered by the rating agency can generate a financial bubble. However, the purpose is not to deliver a judgment on the function or the utility of rating agencies. It is rather to underline the influence that a particular kind of information can exert on the representations of investors and on market dynamics. In a sense, the model pleads for recognition of public regulation since we

demonstrate that institutions have a capacity to act on the representations of the agents (in inciting them by an inflow of information) and thus on market dynamics.[8]

REFERENCES

Arthur, W. B. (1994). Inductive reasoning and bounded rationality. *American Economic Review, 84*(2), 406-411.

Arthur, W. B., Holland, J. H., LeBaron, B., Palmer, R., & Tayler, P. (1997). Asset pricing under endogenous expectations in an artificial stock market. In W. B. Arthur, S. N. Durlauf, & D. Lane (Eds.), *The economy as an evolving complex system II* (pp. 15-44). Redwood City, CA: Addison-Wesley.

David, P. (1985). CLIO and the economics of QWERTY. *American Economic Review, 75,* 332-337.

De Long, J. B, Schleifer, A., Summers, L. H., & Waldmann, R. J. (1990). Positive feedback investment strategies and destabilizing rational speculation. *Journal of Finance, 45*(2), 379-395.

Fama, E. F. (1970). Efficient markets capital: A review of theory and empirical work. *Journal of Finance, 25,* 383-417.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Keynes, J. M. (1936). *The general theory of employment, interest and money*. New York: Harcourt Brace and World.

Kirman, A. P. (1997). The economy as an evolving network. *Journal of Evolutionary Economics, 7*(4), 339-353.

Kirman, A. P. (1999). Aggregate activity and economic organization. *Revue Européenne des Sciences Sociales, XXXVII*(114), 189-230.

Lane, D. (1993). Artificial worlds and economics, part II. *Journal of Evolutionary Economics, 3*(2), 177-197.

Lesourne, J. (1991). *Economie de l'ordre et du désordre*. Paris: Collection Economica.

Lesourne, J., Orléan, A., & Walliser, B. (2002). *Leçons de microéconomie évolutionniste*. Paris: Editions Odile Jacob.

Marengo, L., & Tordjman, H. (1996). Speculation, heterogeneity and learning: A simulation model of exchange rate dynamics. *Kyklos, 49*(3), 407-438.

Moss, S., & Rae, J. (1992). *Artificial intelligence and economic analysis. Prospect and problems*. Edward Elgar Publishing.

Orléan, A. (1999). *Le pouvoir de la finance*. Paris: Editions Odile Jacob.

Paulré, B. (1997). Evolutionnisme contemporain et auto-organisation. *Economie Appliquée, L*(3), 121-150.

Schelling, T. C. (1960). *The strategy of conflict*. Cambridge, MA: Harvard University Press.

Simon, H. (1979). Rational decision making in business organization. *American Economic Review, 69*(4), 493-513.

Tordjman, H. (1997). Spéculation, hétérogénéité des agents et apprentissage. Un modèle de marché des changes artificiel. *Revue Economique, 48*(4), 869-897.

Tordjman, H. (1998). Evolution: History, changes and progress. In J. Lesourne & A. Orlèan (Eds.), *Advances in self-organization and evolutionary economics* (pp. 9-36). Coll Economica.

Walliser, B. (2000). *L'économie cognitive*. Paris: Editions Odile Jacob.

Wilson, S. W. (1994). ZCS: A zeroth level classifier system. *Evolutionary Computation, 2*(1), 1-18.

KEY TERMS

Cognitive Institution: An institution which allows the polarization of the anticipations formed by actors with different representations on their environment or their respective characteristics (Walliser, 2000).

Complex System: (1) A system that constantly evolve and unfolded over time (Arthur, 2000). (2) A system with multiple elements adapting or reacting to the pattern these elements create (Arthur, 2000). (3) A system in which the information-processing capacities of agents are limited relative to the information available to them (Moss & Rae, 1992).

Information: A signal, message, or perception that produces an effect (a decision) on agents' behavior or on their cognitive state.

Limited Rationality: Each agent (1) does not know all the options offered to him (informational limits), or (2) is not able to evaluate all the consequences of his choices (computational limits) (Simon, 1979). This concept implies a satisficing behavior for each investor.

Market Dynamics: The result of the interactions between the decisions of the various investors evolving in a complex environment. This result can be described on a global level without referring to the specific attributes of the microeconomic agents composing the market and depends on the initial conditions of the system (Lane, 1993).

Path-Dependent Learning: The decisions of an agent result from the history of the interactions of investors in the market (David, 1985).

Self-Organization: A global structure, which did not exist at the start, can appear as the result of the many interactions between investors (Lesourne, 1991). Self-organization design the capacity of a system to produce—in a way that is not necessarily voluntary or conscious—a structure, an organization, a behavior, and/or its own rules of functioning (Paulré, 1997; Lesourne, 1991).

ENDNOTES

1 The approach of Keynes (1936) differs from the standard financial theory according to which stock price corresponds to the actualized sum of future flow of earnings that the stock will generate during the life of the firm.

2 Investors make their decisions under budgetary constraints: purchase is impossible when liquidities are not sufficient.

3 In case of "ignore" orders, loss or gain is only evaluated in order to determine in which case it is better not to buy stocks.

4 The stock index is calculated from the aggregation of stock prices of the 10 firms composing the market. The stock index value corresponds to the arithmetical mean of firms' capitalization.

5 Holland (1975) underlines that the number of alternatives complicates the phase of exploration: agents with limited capabilities are compelled to simplify the problem and will not explore the entire set of alternatives.

6 Negative performances correspond to a decrease in portfolios value.

7 Our system lightly encourages investors to take care of the EPG signal.

8 This model presents preliminary results. A perspective could consist of modifying the ranking proposed by agencies to measure the effect of this shock on investors' behaviors and on market dynamics.

About the Authors

Jean-Philippe Rennard is a senior professor at the Grenoble Graduate School of Business, France, and head of the Department of Management and Technology. He received his PhD from the University Pierre Mendès France of Grenoble. An economist, he specializes in industrial development; a computer scientist, he is deeply involved in biologically inspired computing. The author of a reference book on artificial life, he now mainly works on the applications of biologically inspired computing to economics and management.

* * *

J. R. Abraham is in the final year of his PhD studies in the Advanced Compution in Design and Decision-Making Lab (ACDDM) at the University of West of England (UWE), UK. His work focuses on information extraction and succinct visualization of the output from cluster-oriented genetic algorithms (COGAs). He received his Bachelor of Engineering in electronic and communication engineering from Madras University, India, in 2000 and an MSc in autonomous systems from Exeter University in 2002. He is also currently working as visiting lecturer in the School of Computer Science at UWE. His current research interests include the application of evolutionary computing, and pattern recognition and data mining techniques in engineering design applications.

Vito Albino previously served as a visiting scholar at the University of Cincinnati and as a visiting professor at the University of South Florida. From 1988 to 2000 he taught engineering management at the University of Basilicata, Italy. He is now full professor of innovation and project management and director of the Department of Mechanical and Management Engineering at the Polytechnic of Bari, Italy. A member of several national and international associations, he is the author of more than 100 papers published in national and international journals and conference proceedings in the fields of project management, operations management, and firm clusters.

David Al-Dabass is chair of intelligent systems in the School of Computing & Informatics, Nottingham Trent University, UK. He is a graduate of Imperial College, holds a PhD, and has completed post-doctoral and advanced research fellowships at the Control Systems Centre, UMIST. He is fellow of the IEE, IMA, and BCS, and editor-in-chief of the *International Journal of Simulation: Systems, Science and Technology.* He currently serves as chairman of the UK Simulation Society and is a member of the European Simulation Council for SCS Europe. For more details, see his Web site at http://ducati.doc.ntu.ac.uk/uksim/dad/webpage.htm.

Nikolaos Ampazis received a BEng (Hons) in electrical and electronic engineering from the Imperial College of Science Technology and Medicine at the University of London in 1992, an MSc degree with distinction in Information Processing and Neural Networks from King's College London in 1995, and a PhD in neural computation from the National Technical University of Athens in 2001. He is currently an assistant professor in the Department of Financial and Management Engineering at the University of the Aegean, Greece. His main research interests are theory and learning algorithms of computational intelligence, operations research, and information retrieval.

Carl Anderson has been involved in agent-based modeling and complex systems for the last 10 years. With a PhD in mathematical biology, his prior research focused on generic principles of work organization, insect societies as complex adaptive systems, models of division of labor, self-organization and complexity, task partitioning swarm intelligence, and nature-inspired solutions for business and industry. He now heads a "blue skies" Synthetic Intelligence R&D Group at Qbit, LLC in Bethesda, Maryland, USA.

Robert Axelrod is the Arthur W. Bromage distinguished university professor of political science and public policy at the University of Michigan, USA. He has appointments in the Department of Political Science and the Gerald R. Ford School of Public Policy. Prior to coming to Michigan, he taught at the University of California, Berkeley (1968-1974). He holds a BA in mathematics from the University of Chicago and a PhD in political science from Yale. Dr. Axelrod is also president of the American Political Science Association. He is best known for his interdisciplinary work on the evolution of cooperation which has been cited in more than 500 books and 4,000 articles.

Claude Baron graduated from the National Institute of Applied Sciences (INSA) Toulouse, France, where she studied computer science. She holds a PhD and is a senior lecturer at INSA, France, where she teaches real-time systems and system reliability. Her research interests are software and process modeling and simulation. She is now considering the management of risks linked with the design process, and studies of the possibility to manage them using evolutionary programming. She does her research at the LESIA laboratory, where she is in charge of the Critical Embedded Systems (SEC) Group.

Jason Barr is currently an assistant professor of economics at Rutgers University, Newark, USA. He received his BS from Cornell University in 1992 and his PhD in economics from Columbia University in 2002. His areas of interest include agent-based economics, industrial organization, and firm behavior. He has also done work in other fields such as cooperative game theory and education. His work has appeared in such journals as the *Journal of Economic Dynamics and Control, Journal of Economic Behavior and Organization,* and *Contemporary Economic Policy.*

Yaneer Bar-Yam is president of the New England Complex Systems Institute.

Nicola Baxter joined BT PLC in 2001 after graduating with a master's degree in physics from Manchester University. Since joining, she has been working in the Strategic Analysis and Research unit, on the application of agent-based modeling to business and economic problems. Recent projects

have been the development of a decision support tool to enhance understanding of customer relationship management and the analysis of trading arrangements in bandwidth markets.

Alain Berro is a lecturer in computer science at Toulouse University (IRIT) in France. His main interests are multi-objective optimizations and optimizations in dynamic environments. His analyses are focused on the study of the robustness and improvement of the different existing methods. His research has led to new heuristics based on evolutionary techniques and applied to economic problems.

Riccardo Boero received a degree in economics in 2000 from the Università di Turino, Italy, and a PhD in economics in 2004 from the University of Pavia, Italy. He is currently participating in the PhD program in sociology at the University of Surrey, UK. He is interested in theoretical issues and empirical applications of complexity science in the field of social sciences. He is particularly interested in theoretical aspects of complex systems, simulation tools and techniques, empirical foundations of models, cooperation and competition dynamics, and localized development systems

Anthony Brabazon lectures in the School of Business at University College Dublin, Ireland. His research interests include natural computing and the application of natural computation to the domain of finance. He has in excess of 100 journal, conference, and professional publications, and has been a member of the program committee of multiple conferences on evolutionary computation, as well as acting as reviewer for several journals. He has also acted as consultant to a wide range of public and private companies in several countries. Along with Dr. Michael O'Neill, he founded and heads the Natural Computing Research and Applications Group at University College Dublin (http://ncra.ucd.ie).

Alessandro Brun holds a master's degree with honors in production and management engineering and a PhD on the same subject. He is a teacher of industrial technology, quality management, and sourcing and purchasing courses at Politecnico di Milano, Italy, and deputy director of the MBA program at MIP (the business school of Politecnico di Milano). His main research streams are related to supply chain management, operational improvements in manufacturing and service industries, and total quality management.

Charlotte Bruun is associate professor of economics at Aalborg University, Denmark. She has worked with agent-based computational economics since the mid-1990s. Her main scientific contributions are in monetary economics, with simulation studies on pure-credit economies and agent-based macroeconomics in a Keynesian and Schumpetarian approach.

Nunzia Carbonara received the Laurea degree in electronic engineering in 1995 and a PhD in engineering of advanced production systems in 2000 from Polytechnic of Bari, Italy. She is now assistant professor at the same institution. Her current research interests include geographical clusters, local development, innovation processes, and agent-based simulation. She is involved in many national and international research projects and associations, and she is author of several papers published in national and international journals and conference proceedings.

Rajesh Chakrabarti is assistant professor of finance at Georgia Institute of Technology's College of Management, USA. He holds an MBA from the Indian Institute of Management and a PhD in Management from UCLA.

Nirupam Chakraborti, a professor at the Indian Institute of Technology, Kharagpur, India, earned his PhD from the University of Washington. He conducts research on genetic algorithms, has edited journal issues on this topic, and taught courses on them worldwide.

Mattia Ciprian is a mechanical engineer and PhD candidate in Corporate Finance at University of Trieste, Italy. Since 2003 he collaborates with many financial societies; he is currently involved in multidisciplinary research for a national project named "Complex Systems in Economics."

Carlos Artemio Coello Coello holds a PhD in computer science from Tulane University, USA. He is currently an associate professor at CINVESTAV-IPN, México. He has authored and co-authored more than 120 technical papers and several book chapters. He has also co-authored the book *Evolutionary Algorithms for Solving Multi-Objective Problems* (Kluwer Academic Publishers, 2002).

Pierre Collet is an associate professor at the Université du Littoral, Côte d'Opale, France. He earned an MSc in operating systems and architecture in 1990, and a PhD in computer-aided surgery in 1997 from the Université Paris XI Orsay. He specialized in optimization using nature-inspired algorithms such as ant colony optimization, evolutionary computing, and genetic programming. He is a member of the program committees of many conferences on related issues, and a referee of international journals and PhD theses. He was the general chair of the 5th International Conference on Artificial Evolution (EA'01) and co-organizer of EA'03 and EA'05. In 2006, he will be the chairman of the Computer Applications in Health Care track of the 20th Symposium on Applied Computing of the ACM (SAC'06), and the program chair of EuroGP'06, the premier event in genetic programming. He is currently the president of the French Society for Artificial Evolution (http://ea.inria.fr).

David Collings was educated at Cambridge University and Imperial College where he was respectively awarded a PhD and first degree in physics. In 1996 he joined BT, PLC, UK at Adastral Park and has been working in the Strategic Analysis and Research (STAR) Business Modeling Group, researching new tools and techniques for business modeling. He has published a number of papers in the fields of solid state physics and business modeling, and is the author of a commercial software package.

Lee Cooper is professor emeritus and director of the Venture Development Project at the Price Center for Entrepreneurial Studies, UCLA Anderson School of Management, USA. His current efforts involve strategic marketing planning for new technology ventures.

Ernesto Costa is full professor at the Centro de Informatica e Sistemas da Universidade de Coimbra. He received a 3rd Cycle Thesis from the University of Paris VI in 1981 and a PhD in

computing science from UC in 1985. His main research interests are in the areas of evolutionary computation and machine learning. He is the founder and head of the Evolutionary and Complex Systems Group at UC. He participated in several projects, was awarded three international prizes, organized several international scientific events, and has published more than 100 papers in books, journals, and conference proceedings.

Prithviraj Dasgupta is assistant professor of computer science at the University of Nebraska, Omaha, USA. He received his PhD from the University of California, Santa Barbara in 2001. His research interests include software agents and multi-agent systems for peer-to-peer networks, online dynamic pricing, and mobile agent systems. He has published more than 25 research articles in leading journals and conferences on these topics.

Herbert Dawid studied mathematical economics at the Vienna University of Technology, where he earned his doctoral degree in 1995. He has served as assistant and associate professor at the University of Vienna, associate professor at the University of Southern California in Los Angeles, and since 2003 holds a chair for economic policy at Bielefeld University, USA. He is also a member of the Institute of Mathematical Economics at Bielefeld University. Dr. Dawid's research focuses on the economics of innovation, industrial organization, economic dynamics, evolutionary game theory, and agent-based computational economics. He is associate editor of the *Journal of Economic Dynamics and Control* and of the *Journal of Economic Behavior and Organization*; he also serves on several other editorial boards.

John Debenham is professor of computer science at the University of Technology, Sydney (UTS), Australia. He has a longstanding interest in artificial intelligence. Between 1985 and 1998, he worked on the design of expert, knowledge-based systems. That work is reported in the two research monographs: *Knowledge Systems Design* (Prentice-Hall, 1989) and *Knowledge Engineering: Unifying Knowledge Base and Database Design* (Springer-Verlag, 1998). His recent work has focused on multi-agent systems. During the period 1998 to 2002, he published extensively on multi-agent systems for managing complex business processes. From 2002 to 2005, his work focused on e-negotiation and argumentation agents. He is particularly interested in the design of agents that can exploit the information-rich environment of electronic markets. Professor Debenham is a member of the E-Markets Research Group in the Faculty of IT at UTS.

Giovanna Di Marzo Serugendo is a senior researcher in the Information Systems Department at the University of Geneva, Switzerland. She holds a PhD from the Swiss Federal Institute of Technology in Lausanne in the field of software engineering. Her current interests are related to autonomous systems, self-assembly of software, self-organizing, and self-adapting systems, realized using specification carrying code and reputation systems. She co-chaired different international working groups related to the engineering of self-organizing systems, and she is currently editor-in-chief of the new *ACM Transactions on Autonomous Adaptive Systems*.

Danilo DiStefano studied computational chemistry at the University of Padova, Italy. In 2004, he received a master's in bioinformatics from the University of Bologna, Italy. Currently, he is a

contract researcher at AREA Science Park, Trieste, Italy. His current research interests include unsupervised neural networks, cluster analysis, and textual data mining.

Christos Dimopoulos received his PhD from the University of Sheffield, UK, in 2001. He has published a number of articles in international scientific journals. He was awarded the "Outstanding Paper of the Year" award in 2002 by the Neural Network Council of IEEE. His research interests include the design of single, multi-objective, and hybrid optimization algorithms for the solution of problems in the area of production research. he is currently an assistant professor of computer science at Cyprus College, Cyprus.

Jon Dron is a senior lecturer in the School of Computing, Mathematical, and Information Sciences at the University of Brighton, UK, where he leads a distance-taught degree in Internet computing and teaches in the areas of networking, communication, and learning technologies. His research interests are primarily focused on the design and use of networked technologies to enhance community and group communication, with a particular emphasis on self-organizing systems for learning and teaching. He has developed (and continues to develop) a number of Web-based learning environments designed to facilitate both collective and collaborative learning. He is a national teaching fellow and certified practitioner of the Higher Education Academy, England.

Eva Ebenhöh is working as a research assistant at the Institute of Environmental Systems Research at the University of Osnabrück, Germany. Her work focuses on the implementation of human decision making as multi-agent models. She holds a degree in applied systems science, and her diploma thesis was titled "Foreign Debts of Developing Countries and Sustainable Development."

Tom Erez studied mathematics in the Hebrew University; he is currently employed as a researcher at the Institute for Scientific Interchange in Torino, Italy, in the Complexity and Multi-Agent Systems Division, where he works under the supervision of Professor Soron Solomon. Together they conduct projects in various themes, from network science and socio-economical modeling to collective knowledge management. His main research focus, however, is in artificial cognition and machine learning.

Peyman Faratin is a postdoctoral associate at MIT's Laboratory for Computer Science, USA.

Tiago Ferra de Sousa is an assistant in the School of Technology of the Instituto Politecnico de Castelo Branco, Portugal. He is a member of the Evolutionary and Complex Systems Group at the University of Coimbra; his main research interests are related to data mining, bio-informatics, and biologically inspired computation, mainly in the areas of evolutionary computation and swarm intelligence. His publications include 10 peer reviewed chapters/articles in books, journals, and conference proceedings.

Stefka Fidanova received her MSc in mathematical modeling and her PhD in computer science from Sofia University, Bulgaria, in 1988 and 1999, respectively. In 1999 she was a postdoctoral researcher at INRIA Sophia-Antipolis, France. From 2000 to 2002 she was a Marie Curie fellow at

IRIDIA, Free University of Brussels, Belgium. She is a currently an associate professor at the Institute of Parallel Processing at the Bulgarian Academy of Science, Bulgaria.

J. García has been an associate lecturer of computer science at the Universidad Carlos III de Madrid, Spain since 2000. He received a degree in telecommunication engineering from the Universidad Politécnica de Madrid in 1996 and a PhD from the same university in 2001. He currently works with the Research Group of Applied Artificial Intelligence. He previously was part of the Data Processing and Simulation Group at the Universidad Politécnica de Madrid. He has participated in several national and European projects related to air traffic management. His main interests are computational intelligence applied to engineering aspects in the context of data fusion and computer vision, navigation, and air traffic management. He is author of more than 20 journal articles and has published in more than 40 international conference proceedings.

Ilaria Giannoccaro earned a Laurea degree in mechanical engineering in 1998 from the Politecnico di Bari, Italy, and a PhD in engineering management from the University of Tor Vergata, Rome. Her main research interests are supply chain management and computational organizational theory. She is involved in many research projects at the Polytechnic of Bari, and is author of numerous papers published in national and international journals and conference proceedings in the fields of SC coordination, firm clusters, and contract and revenue sharing.

David Goldberg is the Jerry S. Dobrovolny distinguished professor in entrepreneurial engineering at the University of Illinois at Urbana-Champaign, USA, and director of the Illinois Genetic Algorithms Laboratory. His 1989 text, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley), is one of the most cited texts in computer science, and helped launch worldwide interest in genetic and evolutionary computation. His recent book, *The Design of Innovation: Lessons from and for Competent Genetic Algorithms* (Kluwer, 2002), explores the connections between the design of scalable genetic algorithms and processes of human innovation. In 2004, he was named chief scientist of the Web infrastructure company, Nextumi, Inc.

Timothy Gosling is a research student at the University of Essex, UK, currently completing his PhD in supply chain modeling and strategy generation using evolutionary computation. He has a wide range of interests in artificial intelligence including distributed problem solving and agent technology, evolutionary computation, supply chain modeling, bargaining, negotiations, and games. He also maintains an interest in artificial life and ecological simulation.

Eliécer Gutiérrez holds an ME in industrial engineering from the Universidad de Puerto Rico at Mayagüez, an MSc in computer science, and a BSc in computer science from the Universidad de los Andes, Colombia. Currently he is a PhD student in the Industrial Engineering Department at Universidad de los Andes. His research interests include combinatorial and evolutionary optimization, and their applications to production control and logistics.

Pedro Isasi earned graduate and PhD degrees in computer science from the Universidad Politécnica de Madrid, Spain. He is currently full professor for the EVANNAI Group in the Computer Science Department of the University Carlos III of Madrid, Spain. His main research areas

are artificial intelligence, optimization problems, and automatic learning, together with evolutionary computation techniques and neural networks.

Nanlin Jin is a PhD student at the University of Essex, UK. Her PhD research focuses on using co-evolutionary algorithms to solve bargaining problems. She has broad interests in artificial intelligence, especially co-evolution, evolutionary algorithms, genetic programming, adaptive learning, and constraint satisfaction. Her research also relates to game theory, bargaining theory, and experimental economics.

Mak Kaboudan is a professor of business statistics at the University of Redlands, USA. He taught at Penn State for 16 years before moving to Redlands in 2001. His PhD is in economics from West Virginia University. He has expertise in econometrics, forecasting, fuzzy logic, genetic programming, neural networks, wavelets, and spatial statistics. His contributions are in refereed journal and book chapters. He is published in the *Journal of Economic Dynamics and Control, Journal of Forecasting, Fuzzy Sets and Systems, Journal of Applied Statistics, Computers and Operations Research, Computational Economics, Physics Letters A,* and *New Mathematics and Natural Computing.*

Massimiliano Kaucic holds a BS in mathematics and is working towards his PhD in financial mathematics at the University of Trieste, Italy. Since 2003 he has been a research fellow for corporate finance and collaborates for a national project named "Complex Systems in Economics."

Wolfgang Kerber earned his PhD in 1988 from the University Erlangen-Nürnberg, Germany. He previously served as professor of economics at the Ruhr-University Bochum, and since 1997, he has been professor of economics at Philipps-Universität Marburg, Germany.

Mark Klein is a principal research scientist at the MIT Center for Coordination Science, USA, as well as an affiliate at the MIT AI Lab and the New England Complex Systems Institute.

Halina Kwasnicka is professor of computer science at the Institute of Applied Informatics, Wroclaw University of Technology, Poland. She specializes in artificial intelligence. Her research focuses on evolutionary computation, machine learning, image understanding, and hybrid systems. Her main interests are data mining (especially from a real medical data basis), meta-heuristics and multi-agent systems applied to timetabling problems, and efficiency of evolutionary algorithms.

Witold Kwasnicki is professor of economics at the Institute of Economic Sciences, University of Wroclaw, Poland. His main scientific and educational interests focus on a few, seemingly dispersed, areas, namely: evolutionary approach to socio-economic development; evolutionary economics; innovation processes, entrepreneurship, and knowledge management in a state of rapid technological change; liberalism and the Austrian economics; history of economic thought; and methodology and practice of computer simulation of socio-economic systems.

Stéphanie Lavigne holds a PhD in economics and is professor of strategy at the Toulouse Business School, France, where she is actively involved in research. She is also associated with the

research institute LEREPS-GRES at the University of Toulouse I. She works on institutional investors, models of capitalism, "financiarization" of firms' strategies, information on stock markets, and evolutionary models.

Isabelle Leroux is a lecturer in business administration at Le Mans University (GAINS-ARGUMANS), France. Her main interest is industrial clustering as it applies in the biotechnology and petroleum industries, and network analysis within the social and health sectors. More particularly she works on the links developed by firms with local authorities and research laboratories. She studies strategic relations and opportunistic behaviors between these actors involved in a common local governance.

Henrique Lopes Cardoso studied computing and management, and received his master's degree in Artificial Intelligence from the University of Porto, Portugal in 1999. He is an assistant lecturer and a PhD student in the Electrical Engineering and Computation Department of the Faculty of Engineering of the University of Porto. He also serves as a researcher at the Artificial Intelligence and Computer Science Laboratory of the same university. His research interests include distributed AI, normative multi-agent systems, virtual organizations, and adaptive learning agents.

Azahar Machwe has an undergraduate degree in physics from Delhi University, India, and a master's in information technology from the University of the West of England, Bristol, UK. He is currently in his second year of his PhD studies in evolutionary computing and interactive evolutionary design at the Advanced Computation in Design and Decision-Making Lab (ACDDM). His research interests include intelligent software agent technologies and evolvable interactive systems.

Robert Marks is professor of economics at the Australian Graduate School of Management, Australia, and general editor of the *Australian Journal of Management.* His current research focuses on oligopoly behavior, market design using agent-based models, and industry evolution.

L. Martí earned a BSc with honors in mathematics and computer science from the Universidad de La Habana, Cuba, in 1998. There he was selected as the best graduate in the research field. He remained at the center as a member of the Laboratory of Artificial Intelligence; working first as a junior professor and later on as an instructor professor. From 2001 to 2002 he was granted a research fellowship in the Department of Mathematics and Informatics at the Università degli Studi di Udine, Italy. He is currently completing his PhD at the Group of Applied Artificial Intelligence of the Universidad Carlos III de Madrid, Spain. His main research interests are: machine learning, soft computing, neuro-fuzzy systems, and hybrid neural systems.

Robin Matthews is the director of Center of International Business Policy at Kingston University, UK and the J. M. Keynes professor of management at the Academy of National Economy in Moscow.

Andrés L. Medaglia holds a PhD in operations research from North Carolina State University, a MSc in industrial engineering from the Universidad de los Andes, Colombia, and a BSc in industrial engineering from the Pontificia Universidad Javeriana. Currently he is an associate professor in the Industrial Engineering Department and researcher of the Centro de Optimización y Probabilidad Aplicada (http://copa.uniandes.edu.co) at the Universidad de los Andes. His research interests include large-scale optimization, combinatorial optimization, and multi-objective evolutionary optimization. His research on multi-objective evolutionary optimization has been published by Kluwer Academic Publishers and the *European Journal of Operational Research*.

Ugo Merlone is associate professor of mathematics applied to economics at the University of Torino, Italy. His main interests are mathematical models of organizations and of human behavior in cooperative interactions. He is the author of several papers published in scholarly journals.

David Midgley is professor of marketing at INSEAD, France. His research focuses on the diffusion of innovations and marketing strategy.

Ioannis Minis is a professor in the Department of Financial and Management Engineering of the University of Aegean, Greece. His areas of expertise include design and management of production and operations systems. Dr. Minis holds a PhD in mechanical engineering from the University of Maryland, USA, and has held the positions of assistant and associate professor of mechanical engineering at the same university (1988-1997). He has been conducting research in systems for over 20 years, and has authored several book chapters, and more than 35 journal articles and 45 articles in conference proceedings. Dr. Minis was awarded the Best Paper Award in Database Management at the 1992 ASME CIE Conference, and he was the recipient of the 1993 Earl E. Walker Outstanding Young Manufacturing Engineer Award from the Society of Manufacturing Engineers.

Asunción Mochón is a professor at the Universidad Nacional de Educación a Distancia, UNED (www.uned.es), Spain. She is working in the Department of Applied Economics. In December 2003, she earned her PhD degree working with auction theory. Past and currently research addresses auction theory and genetic algorithms, focused on computational intelligence.

Sarit Moldovan is a PhD candidate at the School of Business Administration, Hebrew University of Jerusalem, Israel. Her main research is about the antecedents and consequences of word-of-mouth communications. Her background is in psychology and computer programming, and she is using both behavioral research and computer simulations in order to understand the individual micro behavior and infer from it on market performance at the macro level. Ms. Moldovan received a scholarship from the Yeshaya Horowitz Association through the Center for Complexity Science, and won the Marketing Science Institute's Alden G. Clayton Doctoral Dissertation Proposal Competition.

J. M. Molina received his PhD in telecommunication engineering from the Universidad Politecnica de Madrid in 1997. He has been a member of the System, Signal and Radio Communi-

cations of the University Politecnica of Madrid since 1992. He is associate professor of the Computer Science Department of Universidad Carlos III de Madrid, Spain, being enrolled in the Complex Adaptive Systems Group. His current research focuses on the application of soft computing techniques (NN, evolutionary computation, fuzzy logic, and multi-agent systems) to engineering problems such as RADAR, robot control, and vision.

Isaac Naveh holds a master's degree in computer science from the University of Missouri, USA. His research interests include hybrid cognitive models and multi-agent learning.

Giulia Nogherotto received the academic degree in International Trade Economics from the University of Trieste, Italy in 2003. She has been a PhD student in corporate finance and a research fellow since 2003. She is currently working for a national project named "Complex Systems in Economics."

Bart Nooteboom is professor of innovation policy at Tilburg University, The Netherlands. His research interests include: innovation, entrepreneurship, organizational learning, inter-firm collaboration and networks, and trust. He has published six books and 200 papers on these and related topics. He was formerly a member of a government committee on technology policy, and currently serves as a member of the Royal Netherlands Academy of Arts and Sciences. He was awarded the Kapp prize for his work on organizational learning and the Gunnar Myrdal prize for his work on trust.

Eugénio Oliveira is full professor at the University of Porto, Portugal. He earned his PhD from the New University of Lisbon in 1984. He is coordinator of NIADR (Distributed Artificial Intelligence Group) in the Faculty of Engineering and co-director of LIACC (Artificial Intelligence and Computer Science Laboratory) at the University of Porto. He is director of both the master's course on informatics engineering and of the doctoral program in informatics engineering at the University of Porto. He was also a co-founder of the AgentLink European Network of Excellence. He is coordinating research projects involving learning in multi-agent environments, emotion-based agent architectures, and agent-based electronic institutions for virtual organizations.

Michael O'Neill is a lecturer in the School of Computer Science and Informatics at University College Dublin, Ireland. He has over 70 publications in the area of natural computing, including the seminal book on grammatical evolution published by Kluwer in 2003, and more recently the publication of *Biologically Inspired Algorithms for Financial Modeling,* published by Springer in 2006. He received his PhD in evolutionary computation (grammatical evolution) from the University of Limerick in 2001, his primary degree (BSc) from University College Dublin in 1996 majoring in biochemistry, and also the HDip in computer science from University College Dublin in 1997.

Claudia Pahl-Wostl is professor for the "Management of Resource Flows," and endowed chair of the German Environmental Foundation at the Institute of Environmental Systems Research, University of Osnabrück, Germany. This new research field is devoted to developing innovative concepts of managing transformation processes towards sustainability, with an emphasis on

an improved understanding of dynamics and management of actor networks and the development of indicators for environmental, economic, and social sustainability. She is particularly interested in participatory agent-based social simulation.

I. C. Parmee established the Advanced Computation in Design and Decision-Making Lab (ACDDM) at Bristol, University of the West of England (UWE), UK in 2000. The research focuses upon the integration of people-centered, computational intelligence technologies with complex design and decision-making processes. Current research is highly multi-disciplinary involving software, drug, engineering, and supply chain design. He is also the scientific director of ACT (www.ad-comtech.co.uk), which specializes in a range of computational intelligence strategies to provide search, optimization, and modeling capabilities across complex industrial and commercial problem domains. These areas of activity are underpinned by 16 years of experience of application of these technologies in close collaboration with industry.

Valentino Pediroda is a research assistant at the University of Trieste, Italy; his research is mainly related to the multidimensional approach in complex design. Recently the developed technologies have been used for financial and economical problems.

M. J. Pérez is actuarial mathematic assistant professor at the Department of Business Administration, Universidad Carlos III de Madrid, Spain. She holds a PhD in economic and enterprise sciences, and a bachelor's degree in economic, enterprise, actuarial and financial sciences from the University of Barcelona. She teaches undergraduate- and graduate-level courses on mathematics and financial analysis at the University Carlos III de Madrid, as well as the online master's in financial interests program at the Universitat Oberta de Catalunya. Her research has focused on the actuarial valuation of the mechanisms of transference of risks, traditional (reinsurance) and alternative, associated to the occurrence of natural catastrophes and terrorist attacks. In particular, she develops models to calculate, from the total claims by wrecks derived from a catastrophe, underlying claims ratios such as insurance-linked options, swaps, and catastrophe bonds. In addition, she has begun working on subjects dealing with the temporary structure of interest rates. Dr. Pérez has presented her research at numerous international congresses in the area of insurances, including insurance mathematics and economics, among others.

Andreas Pyka is associate professor of economics at the University of Augsburg, Germany. After he finished his PhD in 1998 in the field of collective innovation processes, he spent almost two years as a postdoctoral fellow working for the European project, "Self-Organizing Innovation Networks" in Grenoble. In 2001 he was Hertha-Firnberg research fellow and worked as visiting professor at the Austrian Research Centers in Vienna. His research topics are Neo-Schumpeterian economics and innovation research, with a particular focus on collective innovation processes and the industrial organization of innovation. Methodologically he is engaged in the fields of agent-based-computational approaches as well as approaches coming from complexity theory.

David Quintana holds first degrees in business administration and computer science, and a PhD in Finance. He is an assistant professor of computer science at the University Carlos III of

Madrid, Spain, where he lectures on subjects relating to artificial intelligence. His research interests include computational finance and evolutionary computation.

Ana Paula Rocha holds a degree in electrical engineering and computation, and received her PhD in distributed artificial intelligence from the University of Porto, Portugal in 2002. She is a professor in the Electrical Engineering and Computation Department of the Faculty of Engineering at the University of Porto. She is a researcher at the Artificial Intelligence and Computer Science Laboratory at the same university, and is a member of the AgentLink Network of Excellence and the respective SIG on electronic commerce. Her research interests include distributed artificial intelligence, virtual enterprises, negotiation, and learning agents.

Samuel Rochet graduated from the National Engineering School of Brest in 2003. He obtained a master's in automatic systems in Toulouse in 2004, and joined the LESIA, INSA, France where he currently is a PhD student. His fields of research are the assistance during project control and evolutionary methods of optimization.

Juliette Rouchier is a research assistant in cognitive economics at GREQAM, CNRS, France, presently on mission in Japan. After earning a degree in mathematics, she stayed for her master's and PhD thesis at CIRAD (International Centre for Agriculture and Development), also in France. Before joining CNRS she worked at the Center for Policy Modeling in Manchester. Her studies address how trust, social links, and moral debts shape negotiation and outcomes in many types of markets. Her main methodology is to mix field study and multi-agent simulations to build models. She belongs to the Commod Group.

Nicole J. Saam received her MA and DPhil in political science from the University of Stuttgart, Germany. She obtained her habilitation (PD) in sociology from the University of Mannheim, Germany. From 1993 to 2001 she worked at the Institute of Sociology of the University of Munich, Germany. Since 2001 she has been temporary professor at the universities of Mannheim, Leipzig, Marburg, and Erfurt. Her research addresses social theory, especially rational choice; social science methodology, especially social simulation; sociology of organization; sociology of economics; and political sociology.

Yago Sáez received a computer engineering degree from the Universidad Pontificia de Salamanca, Spain, in 1999, and a PhD in computer science from the Universidad Politécnica de Madrid in 2005. Currently, he is an assistant professor for the EVANNAI Group in the Computer Science Department of the University CARLOS III of Madrid, Spain. His main research areas encompass interactive evolutionary computation, evolutionary computation applied to finance and economics, and optimization problems.

Stéphane Sanchez holds a PhD in computer science and is a teaching and research assistant at the Université Toulouse I, France. He works on optimization and emergence of strategies or behaviors in dynamic environments. In this context, his studies focus on the use and the improvement of evolutionary algorithms such as genetic algorithms and classifiers systems. He applies his work to both virtual reality and economic simulations.

Francesco Saraceno majored in economics at the University of Rome, where he obtained a PhD in economic theory in 1999. He later received a PhD in economics from Columbia University. In 2000 he joined the Council of Economic Advisors for the Italian Prime Minister's Office. He has been on leave since March 2002, now working at the Observatoire Français des Conjonctures Économiques in Paris, France. His main research interests are European fiscal and monetary policy, disequilibrium economics, and the theory of organizational behavior. He has published in several international journals, including the *Journal of Economic Dynamics and Control,* the *Journal of Economic Behavior,* and the *Scottish Journal of Political Economy.*

Hiroki Sayama is an assistant professor in the Department of Human Communication at the University of Electro-Communications, Japan.

Lijun Shan is a PhD candidate at the National University of Defense Technology, China. She received her BSc and MS degrees in computer science in 2001 and 2003, respectively. Her research interests are in software engineering and distributed computing, especially agent-oriented methodology and service-oriented computing. She can be reached via e-mail at lijunshancn@yahoo.com.

Rui Shen is a PhD candidate at National Laboratory for Parallel and Distributed Processing, China. He received his BS and MS degrees in computer science in 2002 and 2005, respectively. His research interests are in software engineering, distributed computing, especially multi-agent systems, programming languages, distributed middleware, Semantic Web, and Web-based/e-commerce systems. He can be reached via e-mail at shenrui98@yahoo.com.

Peer-Olaf Siebers is a research fellow in the ASAP Group in the School of Computer Science at the University of Nottingham, UK. He recently completed a PhD degree in manufacturing simulation at Cranfield University. His main research interest is the application of computer simulation to study human-oriented complex adaptive systems. Complementary fields of interest include distributed artificial intelligence, biologically inspired computing, game character behavior modeling, and agent-based robotics. Currently he is investigating the role of management practices in closing the productivity gap that exists between the UK and its major international competitors, particularly the USA.

Arlindo Silva lectures in the School of Technology of the Instituto Politecnico de Castelo Branco, Portugal. He is a member of the Evolutionary and Complex Systems Group at the University of Coimbra; his main research interests are related to biologically inspired computation, mainly in the areas of evolutionary computation and swarm intelligence. He has more than 30 peer reviewed publications in books, journals, and conference proceedings.

Sorin Solomon is professor of physics at the Hebrew University of Jerusalem, Israel and director of the Division of Multi-Agent Systems, Lagrange Interdisciplinary Laboratory for Excellence in Complexity, Institute of Scientific Interchange, Italy. He also serves as chair of the EU Commission Expert Group on Complexity Science, as well as scientific coordinator of: GIACS, a

coordination action of about 100 complexity groups; CO3 on the emergence of collective dynamics from random noise; and –DAPHNet, integrating medical measurements into a coherent organism. He is further involved with the Center for Complexity Science, the *International Journal of Modern Physics,* and the *Journal of Statistical Mechanics*. He authored *Microscopic Simulation of Financial Markets,* published by Academic Press.

Ron Sun is professor of cognitive science at Rensselaer Polytechnic Institute, USA. His research interests ranges from cognitive architectures to social simulation, and from hybrid connectionist/symbolic systems to human skill learning.

Kuldar Taveter holds a PhD in English and is a research fellow at the University of Melbourne, Australia. Previously, he was working as a research scientist for VTT Information Technology (Technical Research Centre of Finland), which acts as a mediator between academics and industry. His research interests are agent-oriented domain analysis, and simulation and automation of business and manufacturing processes based on such analysis, as well as software agents and ontologies. At the University of Melbourne, Dr. Kuldar is working in the areas of agent-based business process automation and optimization (in cooperation with industry) and ontology reconciliation.

Pietro Terna is full professor of economics at the University of Torino, Italy. His recent works are in the fields of artificial neural networks and economic and financial modeling, and of social simulation with agent-based models; in this direction he has been pioneering the use of Swarm.

Edward Tsang is a professor of computer science at the University of Essex, UK. He is also the deputy director of Center for Computational Finance and Economic Agents. He has broad interest in artificial intelligence, including heuristic search, computational finance and economics, constraint satisfaction, combinatorial optimization, scheduling, and evolutionary computation.

Neil Urquhart, a native of Scotland, is a lecturer in the School of Computing at Napier University, Scotland. He earned his PhD in the field of evolutionary algorithms and multi-agent systems. His current research interests include automated problem generators, distributed systems, and emergent computing with an emphasis on real-world problems.

Thomas Vallée is an assistant professor of economics at the LEN, University of Nantes. He earned his PhD from the same university in 1999; his principal field of specialization was dynamical and evolutionary games. Other areas of research and teaching interest are the economics of technological innovation and networks. Dr. Vallée has published articles in the *Journal of Evolutionary Economics, Journal of Macroeconomic Dynamics, Journal of Computational Economics,* and *The Revue Economique.*

Harko Verhagen earned an MSc in sociology in 1992 and a PhD in computer science in 2000. Subsequently he worked as assistant professor in the Department of Computer and Systems Sciences at Stockholm University, Sweden. His teaching focuses on multi-agent systems, semiology,

and scientific writing, and his research focuses on integrating social and social philosophical theories with multi-agent systems The overarching goal is the development of an agent architecture combining BDI types of reasoning with social reasoning (norms, etc.) to use in "hybrid social spaces," where human and artificial agents co-exist, interact, and create a social system.

Juan Guillermo Villegas holds an MSc in industrial engineering from the Universidad de los Andes, Colombia, and a BSc in production engineering from the Universidad EAFIT, Colombia. He recently joined the Industrial Engineering Department of the Universidad de Antioquia, Colombia. His research interests include multi-objective evolutionary optimization, facility location, and vehicle routing.

Gerd Wagner is professor of Internet technology at Brandenburg University of Technology at Cottbus, Germany. His research interests include agent-oriented modeling and agent-based simulation, foundational ontologies, (business) rule technologies, and the Semantic Web. He has published one book, *Foundations of Knowledge Systems* (Kluwer Academic, 1998), and he is a steering committee member of the international rule markup language standardization effort *RuleML.org* and the leader of a work package on rule markup in the European research network *REWERSE.net*.

Ji Wang is a professor of computer science at the National Laboratory for Parallel and Distributed Processing, China. He received his PhD in computer science in 1995. His research interests include software engineering, formal methods, and programming methodology. He can be reached via e-mail at jiwang@mail.edu.cn.

Klaus Wersching studied industrial engineering at the Technical University Darmstadt, Germany. Since 2003 he is working as a research assistant under the Chair for Economic Policy of Dr. Herbert Dawid at Bielefeld University, Germany. Mr. Wersching is working on PhD thesis in the field of agent-based simulation models of industry dynamics. His particular interest lies in the interplay of different forms of proximity and innovation.

Ali A. Yassine is an assistant professor of general engineering at the University of Illinois at Urbana-Champaign (UIUC), USA, and the director of the Product Development Research Lab. He teaches a graduate class on systems and entrepreneurial engineering and an undergraduate class in operations research. His research involves managing the development process of complex engineering systems, design process modeling, and IT-enabled concurrent engineering. Prior to joining UIUC, Dr. Yassine was a research scientist at the MIT Center for Technology, Policy and Industrial Development (CTPID) from 1999-2002. He also consulted on numerous occasions for the automotive (Ford Motor Company and General Motors) and telecommunications (Global One) industries in areas of decision analysis and product development management. Dr. Yassine earned his BE in mechanical engineering in 1988 from the American University of Beirut, and received his MS and PhD degrees in 1989 and 1994 in industrial and manufacturing engineering from Wayne State University, Michigan.

Tian-Li Yu is a member of the Illinois Genetic Algorithms Laboratory and a PhD student in the Computer Science Department at the University of Illinois at Urbana-Champaign, USA. He received his BS in electrical engineering in 1997 from the National Taiwan University and his MS in computer science in 2003 from the University of Illinois at Urbana-Champaign.

Hong Zhu is a professor of computer science at Oxford Brookes University, UK. He received BSc, MSc, and PhD degrees in computer science from Nanjing University, China, in 1982, 1984, and 1987, respectively. His research interests are in software engineering which include software development methodologies, software testing, and Web-based applications. He can be reached via e-mail at hzhu@brookes.ac.uk.

Marta Zorzini is a PhD student in the Department of Management, Economics, and Industrial Engineering, Politecnico di Milano, Italy. Her current research interests are in production planning with a particular emphasis on make-to-order systems, and in after-sales service.

Index

Volume I, pp. 1-394 / Volume II, pp. 395-925

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Y