SPRINGER OPTIMIZATION AND ITS APPLICATIONS 40

Michael J. Hirsch · Panos M. Pardalos
Robert Murphey (Editors)

Dynamics of Information Systems

Theory and Applications

Springer

DYNAMICS OF INFORMATION SYSTEMS

DYNAMICS OF INFORMATION SYSTEMS

Theory and Applications

Edited By

MICHAEL J. HIRSCH
Raytheon, Inc.
Orlando, Florida, USA

PANOS M. PARDALOS
University of Florida
Department of Industrial and Systems Engineering
Gainesville, Florida, USA

ROBERT MURPHEY
Air Force Research Laboratory
Munitions Directorate
Eglin AFB, Florida, USA

Springer

Editors

Michael J. Hirsch
Raytheon, Inc.
3323 Pelham Road
Orlando, Florida, 32803
USA
mjh8787@ufl.edu

Robert Murphey
Air Force Research Laboratory
Munitions Directorate
101 W. Eglin Boulevard
Eglin AFB, Florida, 32542
USA
murphey@eglin.af.mil

Panos M. Pardalos
University of Florida
Department of Industrial
& Systems Engineering
303 Weil Hall
Gainesville, Florida, 32611-6595
USA
pardalos@ise.ufl.edu

ISSN 1931-6828
ISBN 978-1-4419-5688-0 e-ISBN 978-1-4419-5689-7
DOI 10.1007/978-1-4419-5689-7
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010925362

Mathematics Subject Classification (2010): 37D40, 65Kxx, 68U35, 68Txx

Cover illustration: “Temple of Poseidon at Cape Sounion, circa 440 BC”. Photo taken by Elias Tyligadas.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Dynamics of Information plays an increasingly critical role in our society. Networks affect our lives every day. The influence of information on social, biological, genetic, and military systems must be better understood to achieve large advances in the capability and understanding of these systems. Applications are wide-spread and include: the detection of terrorist networks, the design of highly functioning businesses and computer networks, modeling the distributed sensory and control physiology of animals, quantum entanglement, genome modeling, multi-robotic systems, and industrial and manufacturing safety.

Classical Information Theory is built upon the notion of entropy which states that for a message to contain information it must dispel uncertainty associated with the knowledge of some object or process. Hence, large uncertainty means more information, small uncertainty means less information. For a networked system, classical information theory describes information that is both joint and time varying. However, for networked systems, information theory can be of limited value. It is cumbersome if not confusing to define joint and conditional information in even relatively small (Bayesian) networks. The curse of dimensionality is one large factor. So is causality, which is functionally critical to determine yet often difficult to ascertain. Entropy does not attend to the value or influence of information, even though in a network, some information, though potentially large in its entropy, could have little value or influence on the rest of the network, while another, less entropic, piece of information may have a great deal of influence on the rest of the system. How information flows and is modified through a system is not dependent upon entropy but more likely on how potentially useful the information is. How the value of information is linked to the connectedness of the network (and vice versa) is critical to analyzing and designing high performing distributed systems, yet is not well studied.

This book presents the state of the art concerning how information, usually in the form of sensing and control, influences the evolution of a distributed or networked system. Fundamentally, this deals with the potential influence information has on the system and how that information flows through a system and is modified in time and space. The chapters in this book relate to concepts that increase knowledge of the relational aspects of information as opposed to the entropic content of information.

We gratefully acknowledge the financial support of the Air Force Research Laboratory and The Center for Applied Optimization at The University of Florida. We thank the contributing authors, the anonymous referees, and Springer Publishing for making the publication of this book possible.

Michael J. Hirsch
Panos M. Pardalos
Robert Murphey

Contents

Chapter 1
The Role of Dynamics in Extracting Information Sparsely Encoded in High Dimensional Data Streams

Mario Sznaier, Octavia Camps, Necmiye Ozay, Tao Ding, Gilead Tadmor, and Dana Brooks

Summary A major roadblock in taking full advantage of the recent exponential growth in data collection and actuation capabilities stems from the *curse of dimensionality*. Simply put, existing techniques are ill-equipped to deal with the resulting overwhelming volume of data. The goal of this chapter is to show how the use of simple dynamical systems concepts can lead to tractable, computationally efficient algorithms for extracting information sparsely encoded in multimodal, extremely large data sets. In addition, as shown here, this approach leads to nonentropic information measures, better suited than the classical, entropy-based information theoretic measure, to problems where the information is by nature dynamic and changes as it propagates through a network where the nodes themselves are dynamical systems.

1.1 Introduction

The recent exponential growth in data collection and actuation capabilities has the potential to profoundly impact society, with benefits ranging from safer, self- aware environments, to enhanced image-based therapies. A major road-block to realizing this vision stems from the curse of dimensionality. Simply put, existing techniques are ill-equipped to deal with the resulting overwhelming volume of data.

This chapter discusses the key role that dynamics can play in timely extracting and exploiting actionable information that is very sparsely encoded in high dimensional data streams. Its central theme is the use of dynamical models as information encoding paradigms. Our basic premise is that spatio-temporal dynamic information can be compactly encapsulated in dynamic models, whose rank, a measure of

This work was supported in part by NSF grants IIS–0713003 and ECCS–0901433, AFOSR grant FA9550–09–1–0253, and the Alert DHS Center of Excellence.

M. Sznaier (✉) · O. Camps · N. Ozay · G. Tadmor · D. Brooks
ECE Department, Northeastern University, Boston, MA 02115, USA
e-mail: msznaier@coe.neu.edu

T. Ding
Department of Electrical Engineering, Penn State University, University Park, PA 16802, USA

the dimension of useful information, is often far lower than the raw data dimension. This premise amounts to a reasonable "localization" hypothesis for spatio-temporal correlations, and is a given in mechanical and biological processes. Embedding problems in the conceptual world of dynamical systems makes available a rich, extremely powerful resource base, leading to robust solutions, or in cases where the underlying problem is intrinsically hard, to computationally tractable approximations with sub-optimality certificates. For instance, in this context, changes in the underlying process can be detected by simply computing the rank of a Hankel matrix constructed from the data and missing information can be recovered by solving a rank minimization problem that can be relaxed to a tractable semidefinite program. A third example is the comparison of data streams in order to establish whether they correspond to time traces of the same phenomenon: it is vastly easier to quantify the difference between two dynamic models (often requiring only a rank comparison) than to search for elusive overlapping time sequences and then compare two such very high dimensional data streams. Finally, the use of dynamic models leads naturally to nonentropic information measures, better suited for problems where the information is by nature dynamic and changes as it propagates through a network where the nodes themselves are dynamical systems. These ideas are illustrated with several examples from different applications, including change detection in video sequences, motion segmentation, and uncovering copromoted genes.

1.2 Key Subproblems Arising in the Context of Dynamic Information Extraction

The challenges entailed in exploiting dynamic information sparsely encoded in very large data sets are illustrated in Fig. 1.1: In all cases, decisions must be taken based on events discernible only in a small fraction of a very large data record: a short video sequence adds up to megabytes, yet the useful information (a change of behavior of a single target), may be encoded in just a portion of a few frames, e.g., less than 10^{-6} of the total data. Similarly, the data from the diauxic shift experiment shown in Fig. 1.1(c) consists of 342×10^3 data points from the time traces of 1,920 promoters, (e.g., a total of 19 Mb of data), yet only a few critical time instants and promoter correlations are of interest. Additional challenges arise from the quality of the data, often fragmented and corrupted by noise. Addressing these challenges requires solving the following subproblems:

A: Nonlinear Embedding of Dynamic Data Finding low dimensional manifold structures in data, a hallmark of machine learning, is a key precursor to both dimensionality reduction and robust information extraction. Existing static techniques ([2] and references therein) provide low dimensional embeddings, but fail to exploit the large gap between data dimension and dynamic rank. As we will show in this chapter, this can be accomplished by employing low rank dynamic models to capture time/parameter dependence on low dimensional manifolds that maximally absorb stationary high dimensions and nonlinearities.

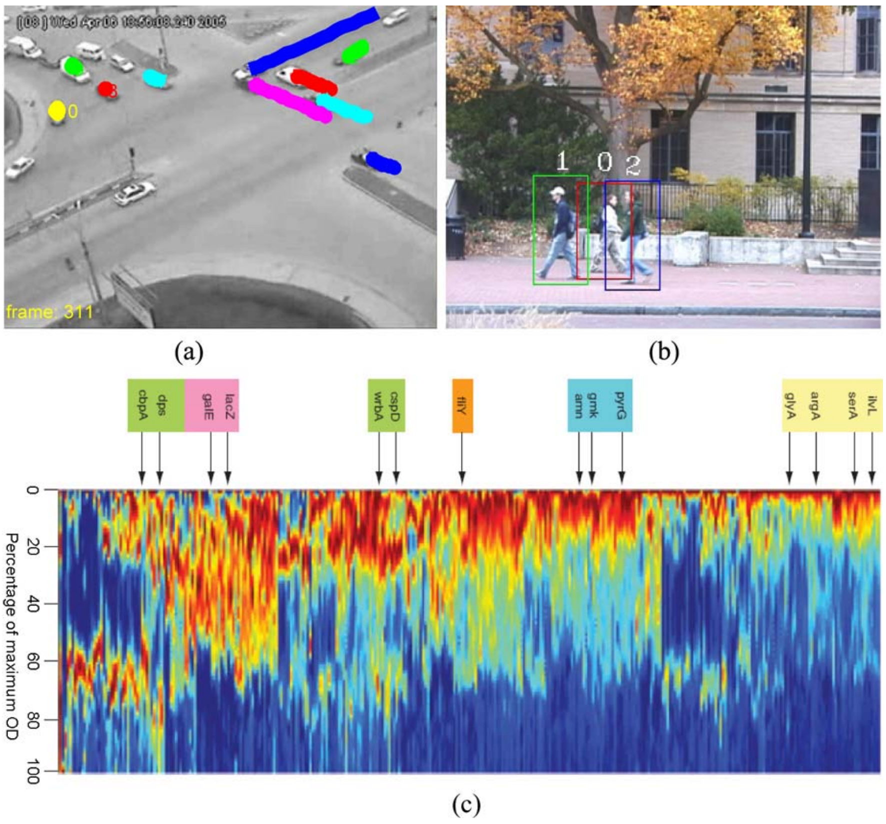


Fig. 1.1 Examples of sparsely encoded information. (**a**) Detecting a traffic accident. (**b**) Tracking a person in a crowd. (**c**) *gfp*-visualized promoter activation during a diauxic shift experiment in E. coli [1]. In all cases fewer than $\mathcal{O}(10^{-3})$ to $\mathcal{O}(10^{-6})$ of the data is relevant

B: Uncovering Structures Embedded in Data A key step in information extraction is the ability to find structures embedded in the data. For example, when analyzing data generated by an unknown number N_o of sources, it is of interest to identify the number of sources, associated substreams, and the individual dynamics. This is commonly accomplished by searching for statistical correlations or exploiting a priori known structural constraints. For example, independently moving objects in a video clip are efficiently detected by factorizing a matrix of feature trajectories [3–5]. However, methods based on correlations and (application dependent) a priori information alone are fragile to missing/corrupted data and have trouble disambiguating structures with overlapping *kinematic* or *statistical properties*. As shown here, these difficulties can be avoided by seeking *dynamically coherent* substreams, e.g., subsets that can be jointly explained by low rank models. Further, this task can be efficiently carried out *without* explicitly finding these models, by estimating ranks of Hankel matrices constructed from time traces. Incorporating a priori available information allows for retaining the advantages of existing methods while substantially improving robustness.

C: Dynamic Data Segmentation The goal here is to partition the data record into maximal, disjoint sets within which the data satisfies a given predicate. Examples include segmenting a video sequence of a person into its constituent activities, or identifying time periods where a given group of gene promoters is active. While this problem has been the object of considerable research in the past decade, it remains very challenging in cases involving noisy data, where most existing methods lead to computationally demanding problems [6, 7], with poor scaling properties. As we will show in the sequel, the use of dynamics provides a unified, efficient approach to robust segmentation. In its simplest form, the idea is to group data according to the complexity of the model that explains it. Intuitively, models associated with homogeneous data, e.g., a single activity or metabolic stage, have far lower complexity than those jointly explaining multiple datasets. Boundaries are thus characterized by a step increase in model complexity. In turn, these jumps in model complexity can be efficiently detected by examining the singular values of a matrix directly constructed from the data.

D: Dynamic Interpolation Data streams are often fragmented: clinical trial patients may miss appointments, targets may be momentarily occluded. The challenges here are to (i) identify fragments belonging to the same data sets (for instance, "tracklets" corresponding to a track of a single target, fragmented due to occlusion), and (ii) interpolate the missing data while preserving relevant dynamical invariants embedded in it. The latter is particularly important in cases where a transition is mediated by the missing data. An example is detecting an activity change from video data, when the transition point is occluded. Formulating the problem as a minimum order dynamical interpolation one leads to computationally attractive solutions, whereby values for missing data are selected as those that do not increase the complexity—or rank—of the model underlying the data record.

E: Hypothesis Testing and Distributed Information Sharing Examples include determining whether (possibly nonoverlapping) data streams correspond to the same process or assessing whether a data set is a realization of a given process. In turn, this entails computing worst-case distances between data and model predictions, a task that can be efficiently accomplished by combining concepts from dynamical systems and information based complexity. Situations involving multiple information sources and users require the ability to (i) maintain consistent data labeling across sources, and (ii) mitigate the communications and computational burdens entailed in sharing very large datasets. Both issues can be efficiently addressed by exploiting the dynamical models underlying the data. Briefly, the idea is to identify a dynamic operator mapping the dynamic evolution of data projections over individual manifolds, amounting to a *dynamical registration* between sources. Sharing/comparing data streams then entails transmitting only the (low order) projections of dynamic variables and running these projections through the interconnecting operator.

In the remainder of this chapter, we show how the use of dynamic models that compactly encapsulate relevant spatio-temporal information provides a unified set of tools leading to computationally efficient solutions to problems A–E above. In all cases, these solutions will be illustrated with practical examples.

1.3 Nonlinear Embedding of Dynamic Data

In the past few years, considerable research has been devoted to the problem of non-linear dimensionality reduction via manifold embedding. Briefly, the idea is to obtain lower complexity data representations by embedding it into low dimensional non-linear manifolds while preserving spatial neighborhoods. Commonly used methods include locally linear embeddings (LLE) [8], Isomap [9], Laplacian Eigenmaps [10], Hessian LLE [11], and Semidefinite Embedding [12, 13]. These methods successfully exploit spatial correlations to achieve (often substantial) dimensionality reduction. However, they fail to take advantage of the temporal correlations that are characteristic of dynamic data. As we show next, constraining target manifolds to those spanned by feasible *dynamical* trajectories enables additional (substantial) dimensionality reduction and provides robustness against missing data and outliers.

The starting point is the realization that since projections to/from manifolds can be modeled as memoryless nonlinearities, the problem of jointly identifying the embedding manifold, the dynamics characterizing the evolution of the data on this manifold, and the projection operators can be recast into the Hammerstein/Wiener system identification problem illustrated in Fig. 1.2. Here, $\Pi_i(.)$ and $\Pi_o(.)$ are memoryless nonlinearities, S is a linear time invariant (LTI) system that describes the temporal evolution of the data on the manifold, and $\mathbf{u} \in R^{n_u}$, $\mathbf{d} \in R^{n_d}$, $\mathbf{u}_m \in R^{n_{u_m}}$ and $\mathbf{y} \in R^{n_y}$, with $n_d \gg n_y$, $n_u \gg n_{u_m}$ represent the respective input (for instance, a vector composed of past values of the output and a stochastic driving signal), the raw data, and their projections on the low dimensional manifold. A potential difficulty here stems from the fact that, as recently shown in [14], robust identification of Hammerstein/Wiener systems is generically NP-hard. However, efficient, computationally tractable relaxations that scale polynomially with problem size (both manifold dimension and number of temporal data points) can be obtained by pursuing a *risk-adjusted* approach. The main idea is to identify first the (piecewise) linear dynamics by sampling the set of signals $(\mathbf{u}_m, \mathbf{y})$, and attempting to find, for each sample (typically a subset of a ball in ℓ^2) an LTI operator S (the dynamics on the manifold) compatible with existing a priori information and such that $\mathbf{y} = S\mathbf{u}_m$. As shown in [15, 16], both steps reduce to a convex optimization problem via the use of Parrot's theorem on norm-preserving matrix expansions and standard results

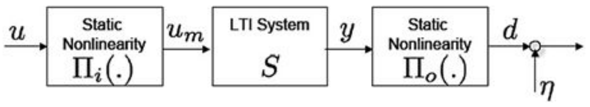


Fig. 1.2 Hammerstein/Wiener System Structure. Wiener system

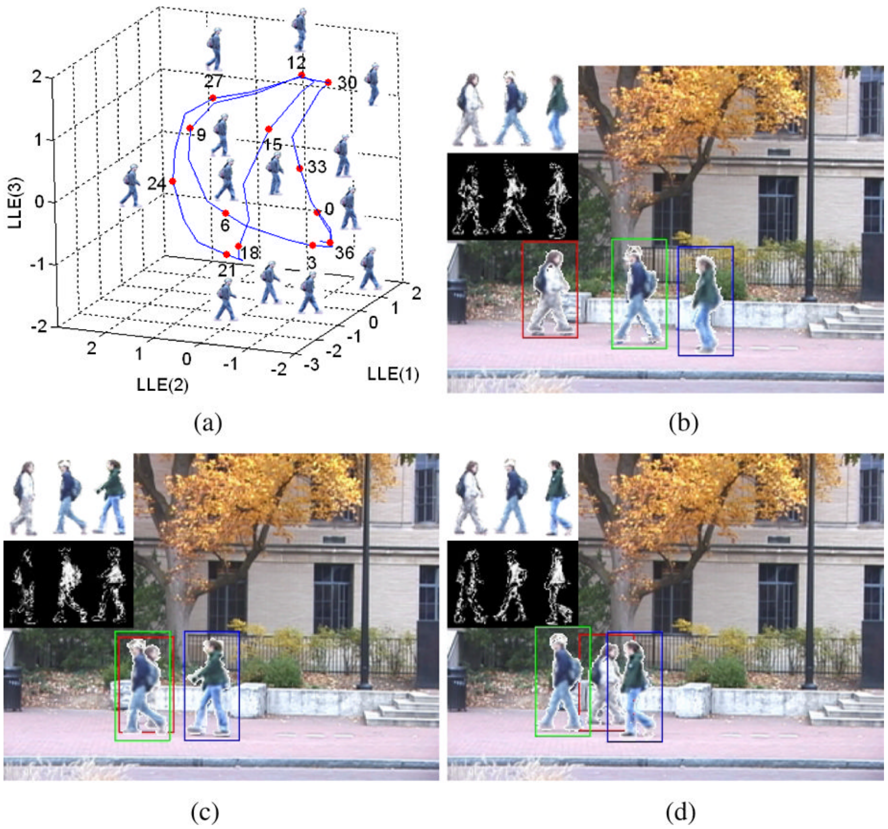


Fig. 1.3 (**a**) Sample 3-dimensional manifold extracted from a walking sequence. (**b**)–(**d**) Use of dynamics on this manifold to predict target position and appearance

on interpolation. The effectiveness of this approach is illustrated in Fig. 1.3, where the application of these ideas enabled sustained tracking of multiple subjects in a cluttered outdoor scene. Here, recasting the problem into a nonlinear identification form allowed for reducing the problem to an identification/prediction one in a 3-dimensional manifold.

It is worth emphasizing that this approach has the, hitherto unavailable, ability to exploit the synergy between the data embedding and dynamic modeling problems to improve robustness and computational properties. Robustness is improved by *automatically* discarding manifolds incompatible with a priori existing information on the dynamics, while computationally attractive models result from maximally absorbing nonlinearities in the manifold structure. Further, the *consistency set* [17] associated with the identification problem provides the means to (in)validate assumptions about the geometry of the manifolds and to quantify the approximation error. Thus, viewing data as a manifestation of hidden dynamics allows a synergy between machine learning (the manifold structure), identification theory (theoretical underpinnings, computational framework) and information based complexity (worst case prediction-error bounds).

1.4 Structure Extraction from High Dimensional Data Streams

Structure extraction methods based on correlations and (application dependent) a priori information alone are often fragile to missing/corrupted data and have trouble disambiguating structures with overlapping *kinematic* or *statistical properties*. As an illustrative example, consider time traces $\mathbf{p}_{ti} = (u_{ti}, v_{ti})^T, t = 1 \ldots n$ of n_p features P_i, $i = 1, \ldots, n_p$, from a single rigid object. Kinematic constraints imply that the rank of the "measurements" matrix $\mathcal{W}_{1:F} \doteq [\mathbf{p}_{ti}] \in R^{2n \times n_p}$ is at most 4 [18]. The number N_o of *independent* rigid bodies can thus be estimated by factorizing that matrix into rank 4 submatrices. Yet this approach fails to disambiguate objects with partially shared motion, as illustrated in Fig. 1.4(a): Here, $\text{rank}(\mathcal{W}) = 7$ due to shared propeller rotations; hence any segmentation based solely on factorizing $\mathcal{W}$ will fail to distinguish this case from the case of just two independently moving propellers. The root-cause is that properties that are invariant under row permutations in $\mathcal{W}$ are limited to revealing geometric dependencies but *ignore dynamic constraints*.[1] As shown next, these ambiguities can be solved through the use of dynamical models that exploit both sets of constraints.

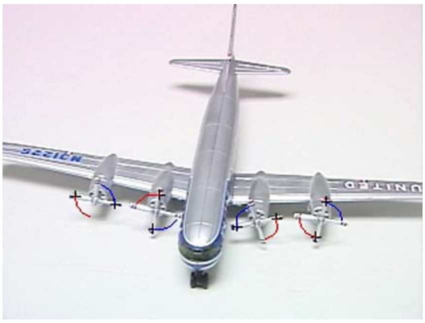
The starting point is the realization that for two points $\mathbf{p}_r, \mathbf{p}_s$ belonging to the same source, the time evolution of $y_{r,s}(k) \doteq \mathbf{p}_r(k) - \mathbf{p}_s(k)$ does not carry information about the overall *group* motion of the source. Equivalently, states associated with group motion are *unobservable* from $y_{r,s}$ if $\mathbf{p}_r$ and $\mathbf{p}_s$ belong to the same dynamic cluster. Hence, the associated Hankel matrix is rank deficient [17] vis-a-vis the case of points from different sources. This leads to the following simple dynamic clustering algorithm:

(i) For each pair (r, s), form the Hankel matrix $\mathsf{H}_{y_{r,s}}$ of pairwise differences $y_{r,s}(k) = \mathbf{p}_r(k) - \mathbf{p}_s(k)$:

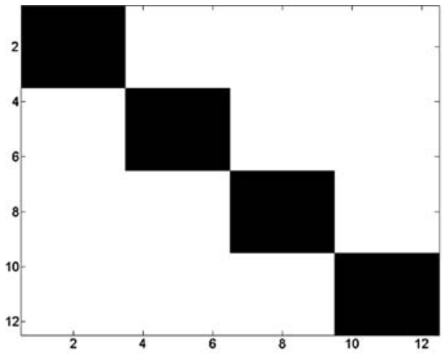
$$\mathbf{H}_y = \begin{bmatrix} \mathbf{y}(1) & \mathbf{y}(2) & \cdots & \mathbf{y}(\frac{n}{2}) \\ \mathbf{y}(2) & \mathbf{y}(3) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{y}(\frac{n}{2}) & \cdots & \cdots & \mathbf{y}(n) \end{bmatrix} \tag{1.1}$$

(ii) Group points according to the minimum value of $\text{rank}[\mathsf{H}_{y_{r,s}}]$.

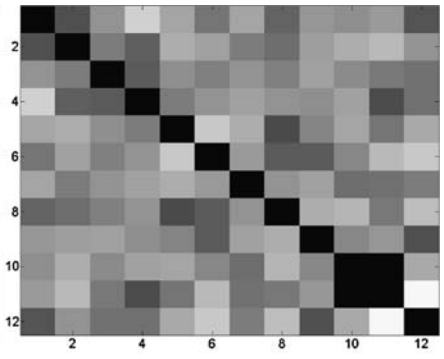
[1] Any permutation of the rows of $\mathcal{W}$ satisfies the same geometric constraints, but corresponds to different time trajectories.



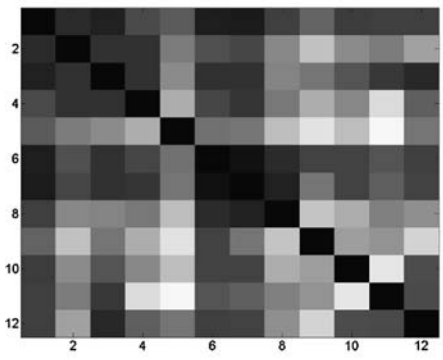
(a)



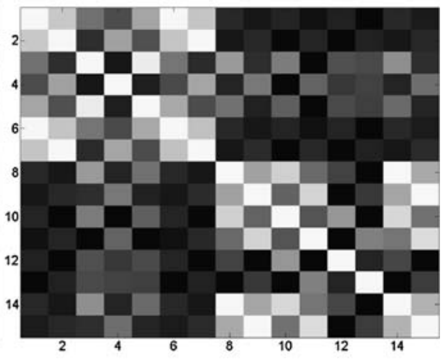
(b)



(c)



(d)

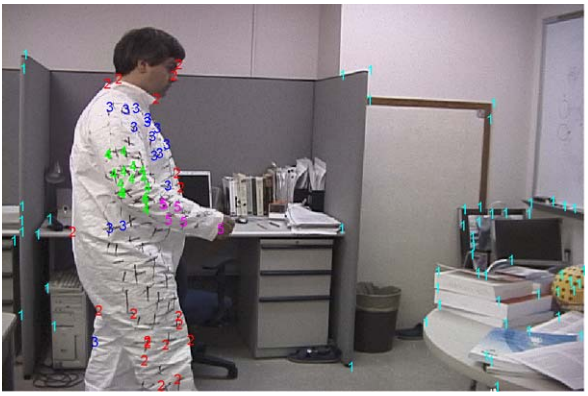


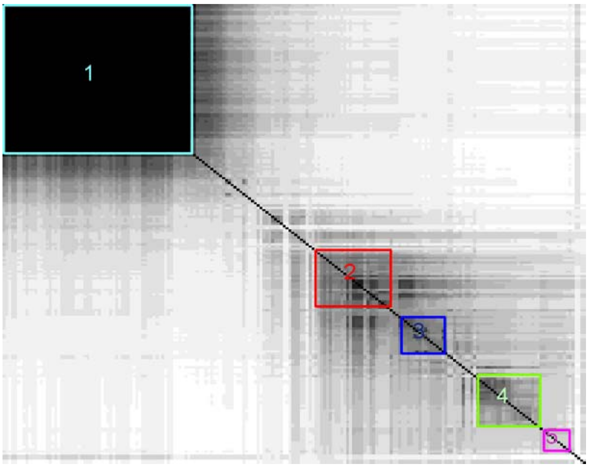
(e)

Fig. 1.4 (**a**) Right and left wing propellers move in opposite directions at the same speed. (**b**) Dynamics based segmentation. (**c**) Costeira–Kanade segmentation. (**d**) Zelnik–Manor–Irani segmentation. (**e**) GPCA segmentation

In this context, robust handling of noisy measurements $\hat{y}(k) = y(k) + \eta(k)$, is accomplished by simply replacing "rank" by the number of singular values above the covariance of the measurement noise,[2] leading to an algorithm computationally no more expensive than a sequence of SVDs. The effectiveness of this approach is illustrated in Fig. 1.4 where darker matrix elements indicate higher correlations: As shown there, the dynamics based approach achieves perfect segmentation, while methods relying solely on factorizations of $\mathcal{W}$ [5, 19, 20] fail.

An interesting property of the dynamics based approach to segmentation, illustrated in Fig. 1.5, is the ability to provide a *hierarchical* segmentation according to the complexity of the joint dynamics. This is key to model the behavior of a target composed by several components acting in a dynamically correlated fashion, e.g., the limbs of a walking person or co-regulated genes. The aggregate behaves as a nonrigid object, whose components share motion modes.


(a)

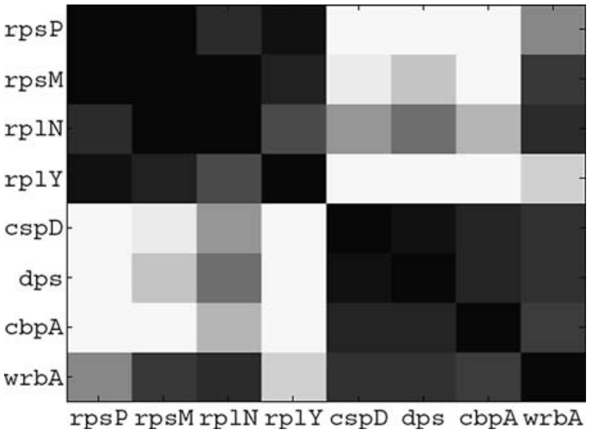


(b)

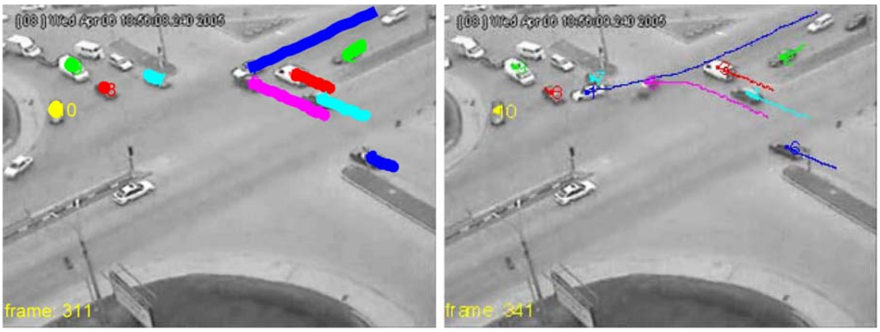
Fig. 1.5 (**a**) Sample frame. (**b**) Structures found using dynamic rank (darker color indicates higher dynamic correlation). The hierarchy in the *lower right corner* corresponds to different portions of the body. (**c**) Dynamic correlation between genes in the diauxic shift experiment of Fig. 1.1(c). The two identified groups correspond to growth related (*top left*) and stationary (*bottom right*) genes. The fainter correlation between wrbA and (rpsM,rplN) was unexpected

[2]In this case $\mathsf{H}_{\hat{y}} = \mathsf{H}_y + \mathsf{H}_\eta$, and, under ergodicity assumptions, $\mathsf{H}_\eta^T \mathsf{H}_\eta$ is an estimate of the covariance matrix of the noise.

Fig. 1.5 (Continued)



(c)

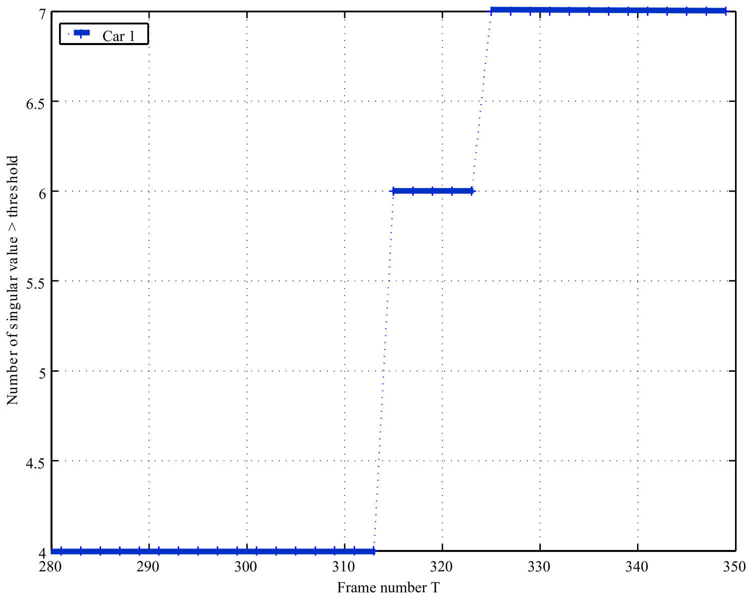


(a)

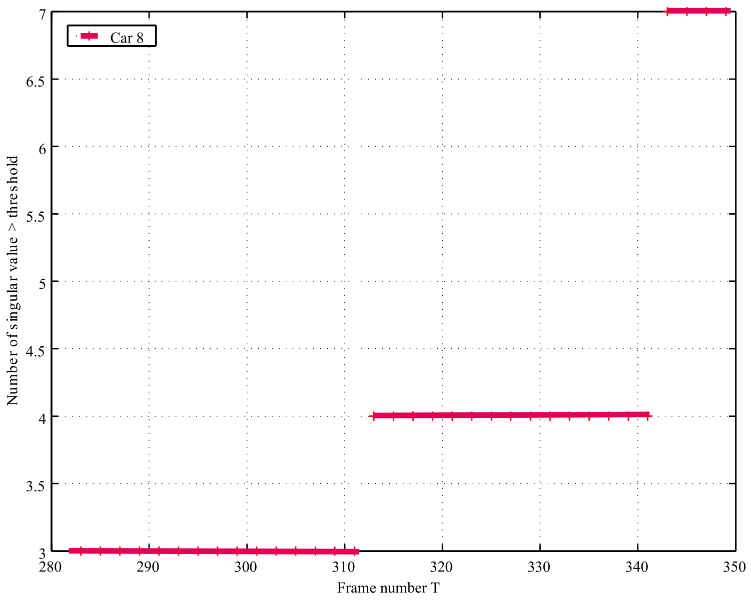
Fig. 1.6 Crash detection. (**a**) Frames 311 and 341. Hankel rank time traces: Car 1 (**b**) and Car 8 (**c**)

1.5 Robust Dynamic Data Segmentation

In principle, changes in the processes underlying a given data record can be detected by a two-tiered approach: identification of an underlying set of models (the consistency set) followed by a model (in)validation step to detect points at which new data are inconsistent with all the models in the set. However, the entailed computational complexity is high, roughly n^5 for n data points. A fast, computationally efficient alternative can be obtained by searching for points where the complexity of the underlying model changes. The main idea behind this approach is the fact that models associated with homogeneous data have far lower complexity than those jointly explaining multiple datasets. Further, the complexity of the (unknown) model can be estimated from the experimental data by computing the number $N_{sv,\sigma}(\mathsf{H}_y)$ of (significant) singular values of a Hankel matrix similar to $\mathbf{H}_y$ in (1.1). Hence, the data record can be segmented according to discontinuities in $N_{sv,\sigma}(\mathsf{H}_y)$. Figure 1.6 illustrates the effectiveness of this approach in detecting contextually abnor-



(b)



(c)

Fig. 1.6 (Continued)

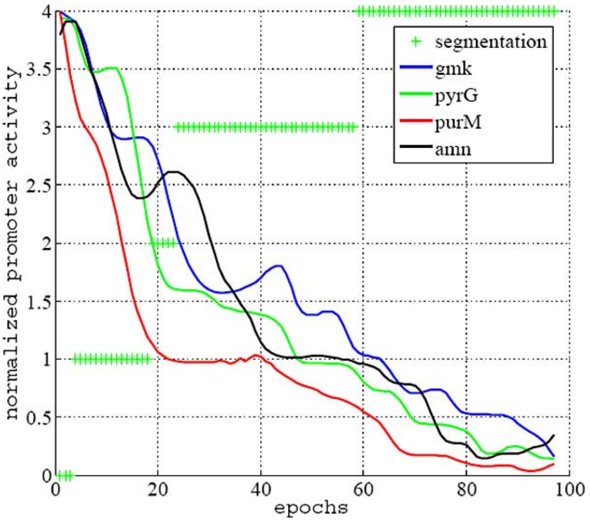


Fig. 1.7 Detecting transitions in an E. coli culture via Hankel rank. Jumps at 20 and 57 correspond to shifts from metabolizing glucose to lactose, to stationary phase, respectively

mal behavior—an accident—evidenced by a jump in the Hankel rank. An application of this technique to detecting changes in promoter activity in E. coli is shown in Fig. 1.7.

The approach outlined above works well for cases where the noise is moderate and adequately characterized as an ℓ^2 bounded signal. Cases where these conditions do not hold (for instance, ℓ^∞ noise) can be handled by a modification of this idea (detecting mode changes in piecewise affine models) as follows. The starting point is the assumption that the data record has been generated by a piecewise affine model of the form:

$$\mathcal{H} : f\left(\mathbf{p}_{\sigma(t)}, \left\{\mathbf{x}(k)\right\}_{k=t-i}^{t+j}\right) = \eta_f \tag{1.2}$$

where f is an affine function[3] of the parameter vector $\mathbf{p}_{\sigma(t)}$ which takes values from a finite unknown set according to a piecewise constant function $\sigma(t)$, and η_f denotes an unknown noise signal. Here, i and j are positive integers that account for the memory of the model (e.g., $j = 0$ corresponds to a causal model, or $i = j = 0$ corresponds to a memoryless model). Next, consider the sequence of *first order differences* of the parameters $\mathbf{p}_{\sigma(t)}$, given by

$$\mathbf{g}(t) = \mathbf{p}_{\sigma(t)} - \mathbf{p}_{\sigma(t+1)} \tag{1.3}$$

Clearly, a nonzero element of this sequence corresponds to a *change* in the underlying model. Hence, partitioning the data record into maximal homogeneous sequences is equivalent to finding a hybrid model of the form (1.2), consistent with the a priori information (e.g., a bound on $\|\eta\|_{\ell^\infty}$) and experimental data, such that the number of nonzero elements of the vector $\mathbf{g}(.)$ is minimized. Formally, defining

[3]That is, $f(\mathbf{p}_{\sigma(t)}, \{\mathbf{x}(k)\}_{k=t-i}^{t+j}) = A(\mathbf{x})\mathbf{p}_{\sigma(t)} + \mathbf{b}(\mathbf{x})$.

$\delta(t) = \|\mathbf{g}(t)\|_\infty$, the objective is to minimize $\|\delta\|_{\ell^o}$, the number of nonzero elements of δ, subject to (1.2). Using the fact that the *convex envelope* of $\|\cdot\|_{\ell^0}$ in R^N is the ℓ^1-norm [21], this nonconvex problem can be relaxed to:

$$\begin{aligned} &\text{minimize}_{\mathbf{p}(t),\eta(t)} \left\|\{\mathbf{g}\}\right\|_{\ell_1} \\ &\text{subject to} \quad f\big(\mathbf{p}(t), \{\mathbf{x}(k)\}_{k=t-i}^{t+j}\big) = \eta(t) \quad \forall t \\ &\qquad\qquad\quad \left\|\{\eta\}\right\|_* \le \epsilon \end{aligned} \tag{1.4}$$

Since f is an affine function of $\mathbf{p}(t)$, (1.4) has a convex feasibility set $\mathcal{F}$. Thus, using the ℓ_1 norm leads to a convex, computationally tractable relaxation. The resulting solution can be further improved using the iterative procedure proposed in [22], based on solving, at each iteration, the following weighted ℓ_1-norm minimization over the convex feasible set $\mathcal{F}$:

$$\begin{aligned} &\text{minimize}_{z,g,p,\eta} \sum_{t=1}^{T-1} w_t^{(k)} z_t \\ &\text{subject to} \quad \left\|\mathbf{g}(t)\right\|_\infty \le z_t \quad \forall t \\ &\qquad\qquad\quad f\big(\mathbf{p}(t), \{\mathbf{x}(k)\}_{k=t-i}^{t+j}\big) = \eta(t) \quad \forall t \\ &\qquad\qquad\quad \left\|\{\eta\}\right\|_* \le \epsilon \end{aligned} \tag{1.5}$$

where $w_i^{(k)} = (z_i^{(k)} + \delta)^{-1}$ are weights with $z_i^{(k)}$ being the arguments of the optimal solution at the k^{th} iteration and $z^{(0)} = [1, 1, \ldots, 1]^T$; and where δ is a (small) regularization constant that determines what should be considered zero.

The choice of $*$, the norm characterizing the noise, is application dependent. For instance, the ℓ^∞-norm performs well in finding anomalies, since in this case the change detection algorithm looks for *local* errors, highlighting outliers. On the other hand, when a bound on the ℓ^1 or ℓ^2-norm of the noise is used, the change detection algorithm is more robust to outliers and it favors the continuity of the segments (i.e., longer subsequences). In addition, when using these norms, the optimization problem automatically adjusts the noise distribution among the segments, better handling the case where the noise level is different in different segments.

1.5.1 Example 1: Video Segmentation

Segmenting and indexing video sequences have drawn significant attention due to the increasing amounts of data in digital video databases. Systems that are capable of segmenting video and extracting key frames that summarize the video content can substantially simplify browsing these databases over a network and retrieving important content. An analysis of the performances of early shot change detection algorithms is given in [23]. The methods analyzed in [23] can be categorized into two major groups: (i) methods based on histogram distances, and (ii) methods based on variations of MPEG coefficients. A comprehensive study is given in [24] where a formal framework for evaluation is also developed. Other methods include those where scene segmentation is based on image mosaicking [25, 26] or frames are segmented according to underlying subspace structure [27].

Given a video sequence of frames $\{\mathcal{I}(t) \in \mathbb{R}^D\}_{t=1}^T$, the video segmentation problem can be solved by first projecting the data into a lower dimensional space, using for instance Principal Component Analysis (PCA), and then applying the sparsification algorithm described above to the projected data (to exploit the fact that the number of pixels D is usually much larger than the dimension of the subspace where the frames are embedded):

$$\mathcal{I}(t) \longmapsto \mathbf{x}(t) \in \mathbb{R}^d$$

Assuming that each $\mathbf{x}(t)$ within the same segment lies on the same hyperplane not passing through the origin[4] leads to the following hybrid model:

$$\mathcal{H}_1 : f\big(\mathbf{p}_{\sigma(t)}, \mathbf{x}(t)\big) = \mathbf{p}_{\sigma(t)}^T \mathbf{x}(t) - 1 = 0 \tag{1.6}$$

Thus, in this context algorithm (1.5) can be directly used to robustly segment the video sequence. It is also worth stressing that as a by-product this method also performs *key frame extraction* by selecting $\mathcal{I}(t)$ corresponding to the minimum $\|\eta(t)\|$ value in a segment (e.g., the frame with the smallest fitting error) as a good representative of the entire segment.

The content of a video sequence usually changes in a variety ways: For instance, the camera can switch between different scenes (e.g., shots); the activity within the scene can change over time; objects, or people can enter or exit the scene, etc. There is a hierarchy in the level of segmentation one would require. The noise level ϵ can be used as a tuning knob in this sense.

Figure 1.8 shows the results of applying this approach to a video sequence, `drama.avi`, available from http://www.open-video.org. The original mpeg files were decompressed, converted to grayscale, and title frames were removed. Each sequence shows a different characteristic on the transition from one shot to the other. The camera is mostly nonstationary, either shaking or moving. For comparison, results using GPCA, a histogram based method and an MPEG method for segmenting the sequences with optimal parameters (found by trial and error) are also shown. Table 1.1 shows the Rand indices [28] corresponding to the clustering results obtained for this sequence and three others from the same database (`roadtrip.avi`, `mountain.avi`, and `family.avi`) using the different methods, providing a quantitative criteria for comparison. Since the Rand index does not handle dual memberships, the frames corresponding to transitions were neglected while calculating the indices. These results show that indeed the sparcity method does well, with the worst relative performance being against MPEG and B2B in the sequence

[4]Note that this always can be assumed without loss of generality due to the presence of noise in the data.

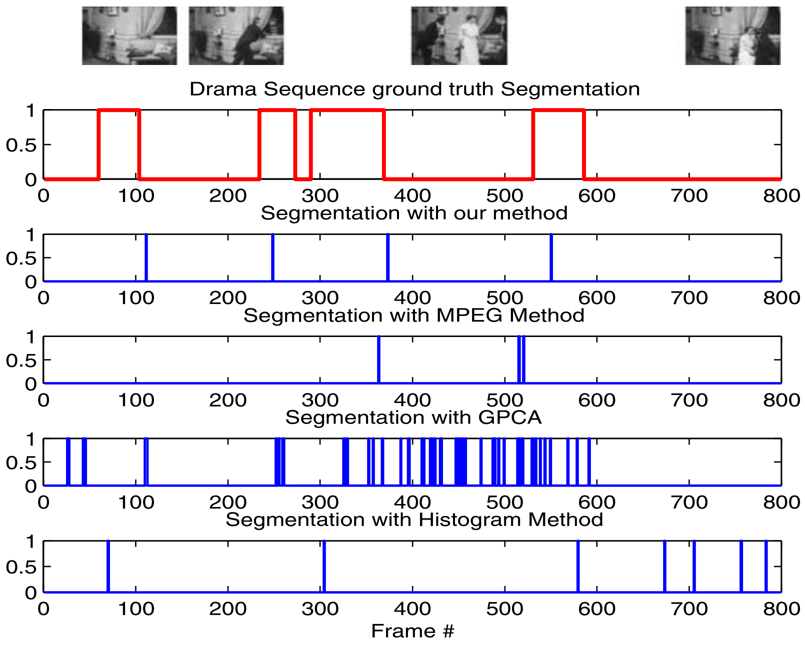


Fig. 1.8 Video segmentation as a hybrid system identification

Table 1.1 Rand indices

	Roadtrip	Mountain	Drama	Family
Sparsification	0.9373	0.9629	0.9802	0.9638
MPEG	1	0.9816	0.9133	0.9480
GPCA	0.6965	0.9263	0.7968	0.8220
Histogram	0.9615	0.5690	0.8809	0.9078

Roadtrip. This is mostly due to the fact that the parameters in both of these methods were adjusted by a lengthy trial and error process to yield optimal performance in this sequence. Indeed, in the case of MPEG based segmentation, the two parameters governing cut detection were adjusted to give optimal performance in the Roadtrip sequence, while the five gradual transition parameters were optimized for the Mountain sequence.

1.5.2 Example 2: Segmentation of Dynamic Textures

Modeling, recognition, synthesis, and segmentation of dynamic textures have drawn a significant attention in recent years [29–32]). In the case of segmentation tasks, the most commonly used models are mixture models, which are consistent with the hybrid model framework.

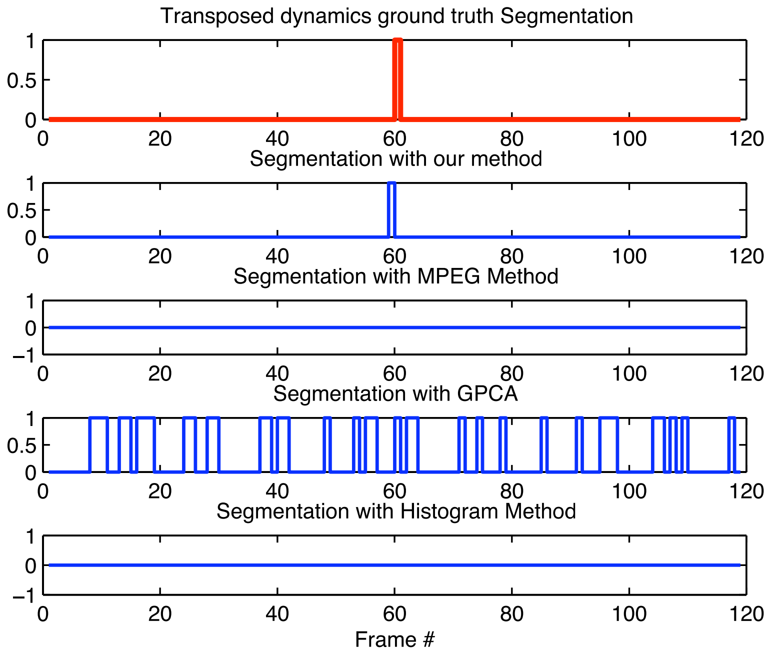


Fig. 1.9 Comparison of segmentation results for the Smoke sequence concatenated with its transposed dynamics

In the sparsification framework described earlier in this section, the problem of temporal segmentation of dynamic textures reduces to the same mathematical problem as the video segmentation problem, with the difference that now the underlying hybrid model should take the dynamics into account. First, dimensionality reduction is performed via PCA ($\mathcal{I}(t) \longmapsto \mathbf{y}(t) \in \mathbb{R}^d$) and then the reduced-order data is assumed to satisfy a simple causal autoregressive model similar to the one in [31]. Specifically, in this case the hybrid model is given by

$$\mathcal{H}_2 : f\left(\mathbf{p}_{\sigma(t)}, \{\mathbf{y}(k)\}_{k=t-n}^{t}\right) = \mathbf{p}_{\sigma(t)}^T \begin{bmatrix} \mathbf{y}(t-n) \\ \vdots \\ \mathbf{y}(t) \end{bmatrix} - 1 = 0 \tag{1.7}$$

where n is the regressor order. This model, which can be considered as a step driven autoregressive model, was found to be effective experimentally.[5] The power of this approach is illustrated in Figs. 1.9 and 1.10 where two very challenging sequences were segmented. The first sequence consists of a patch of dynamic texture (smoke)

[5]The independent term 1 here accounts for an exogenous driving signal. Normalizing the value of this signal to 1, essentially amounts to absorbing its dynamics into the coefficients $\mathbf{p}$ of the model. This allows for detecting both changes in the coefficients of the model and in the statistics of the driving signal.

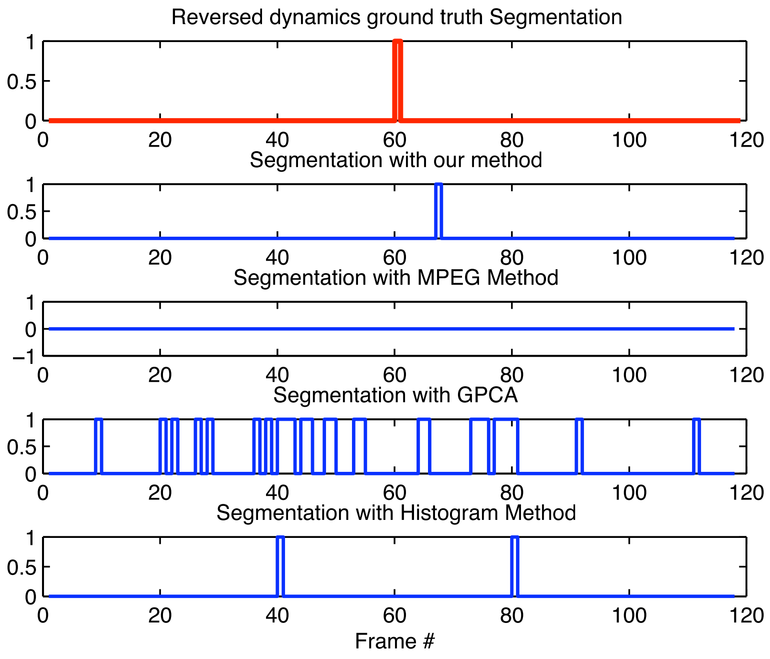


Fig. 1.10 Comparison of segmentation results for the River sequence concatenated with its reversed dynamics

appended in time to another patch from the same texture but transposed. Thus, the two subsequences have the same photometric properties but differ in the main motion direction. The second sequence was generated using another dynamic texture (river) by sliding a window both in space and time (by going forward in time in the first half and by going backward in the second), thus reversing the dynamics due to the river flow.

1.6 Constrained Interpolation of High Dimensional Signals

Consider first the simpler case of interpolating noiseless data, generated by a single LTI system, with McMillan degree bounded by some known n_o. Formally, given a partial sequence $\mathbf{d}_g = \{d_1, \dots, d_q, d_{q+r}, \dots, d_n\}$, the goal is to estimate the missing elements $\mathbf{d}_x = \{d_{q+1}, \dots, d_{q+r-1}\}$ that optimally fit the given data. Intuitively, the best fitting missing elements are those that require adding the least number of modes to the existing model in order to explain the new data. Using the fact that the order of the underlying model is given by the rank of the corresponding Hankel matrix (under the assumption that $n \gg n_o$), this problem can be recast into the following rank minimization form: $\mathbf{d}_{x\,o} = argmin_{\mathbf{d}_x} \text{rank}(\mathsf{H})$ where H is the Hankel matrix associated with the completed sequence $\mathbf{d} = \{d_i\}$. In this context, noise can be readily handled by simply adding a new variable v such that the measured data

$y = d + v$, and a suitable noise description of the form $v \in \mathcal{N}$, a convex, compact set. Finally, since rank minimization is NP-hard [33], using the convex relaxation proposed in [34] leads to the following algorithm:

Algorithm 1: HANKEL RANK MINIMIZATION BASED INTERPOLATION/ PREDICTION

Input at time k: N_h: Horizon length; $\mathcal{I}_a \subseteq [k - N_h, k]$: set of indices of available measurements (with $\text{card}(\mathcal{I}_a) \geq n$); $\mathcal{I}_e \subseteq [k - N_h, k+1]$: set of indices of data to be estimated; with $\mathcal{I}_a \cup \mathcal{I}_e = \mathcal{I}$; available data $y_\ell,\ \ell \in \mathcal{I}_a$; set membership description of the measurement noise $v \in \mathcal{N}$.

Output: Estimates $\hat{\zeta}_\ell$ of $\zeta_\ell, \forall \ell \in \mathcal{I}_e \cup \mathcal{I}_a$

1. Let ζ^* denote the following sequence:

$$\zeta_i^* = \begin{cases} y_i - v_i & \text{if } i \in \mathcal{I}_a \\ x_i & \text{if } i \in \mathcal{I}_e \end{cases}$$

where v, x are free variables, and form the matrix

$$\mathsf{H}_\zeta \doteq \begin{bmatrix} \zeta_{i_1}^* & \zeta_{i_2}^* & \cdots & \zeta_{i_{n+n_u+1}}^* \\ \zeta_{i_2}^* & \zeta_{i_3}^* & \cdots & \zeta_{i_{n+n_u+2}}^* \\ \vdots & \vdots & \ddots & \vdots \\ \zeta_{i_{n+1}}^* & \zeta_{i_{n+2}}^* & \cdots & \zeta_{i_{2n+n_u+1}}^* \end{bmatrix}$$

2. (Approximately) minimize rank$[\mathsf{H}(x, v)]$ by solving the following convex problem in x, v, R, S:

$$\begin{array}{ll} \text{minimize} & Tr(\mathsf{R}) + Tr(\mathsf{S}) \\ \text{subject to} & \begin{bmatrix} \mathsf{R} & \mathsf{H}(x) \\ \mathsf{H}(x)^T & \mathsf{S} \end{bmatrix} \geq 0, \quad \{v_\ell\} \in \mathcal{N}. \end{array}$$

3. Estimate/predict the output ζ_ℓ from the noisy measurements y_ℓ by:

$$\hat{\zeta}_i = \begin{cases} y_i - v_i & \text{if } i \in \mathcal{I}_a \text{ (estimation)} \\ x_i & \text{if } i \in \mathcal{I}_e \text{ (interpolation/prediction)} \end{cases}$$

Examples of application of this idea are shown in Fig. 1.11, where it was used to establish target identity across occlusion, and in Fig. 1.12 where nonlinear embeddings were used first to map the data to a low order manifold where the rank-minimization based interpolation was performed, followed by a remapping of the data to pixel space. Finally, Fig. 1.13 shows how a combination of dynamic interpolation and Hankel-rank based segmentation is able to detect occluded events.

It is worth mentioning that the ideas discussed in this section are directly applicable to hybrid models of the form (1.2). In this case, minimizing the rank is roughly

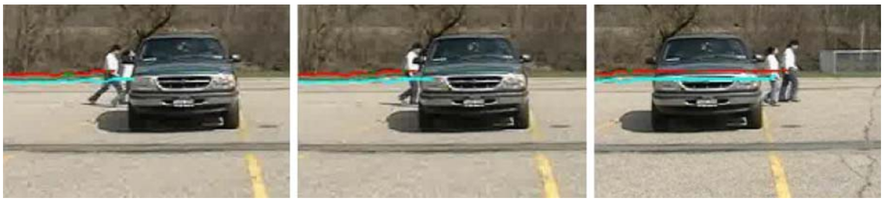


Fig. 1.11 Data interpolation/association across occlusion. (Note that targets change relative positions while occluded)

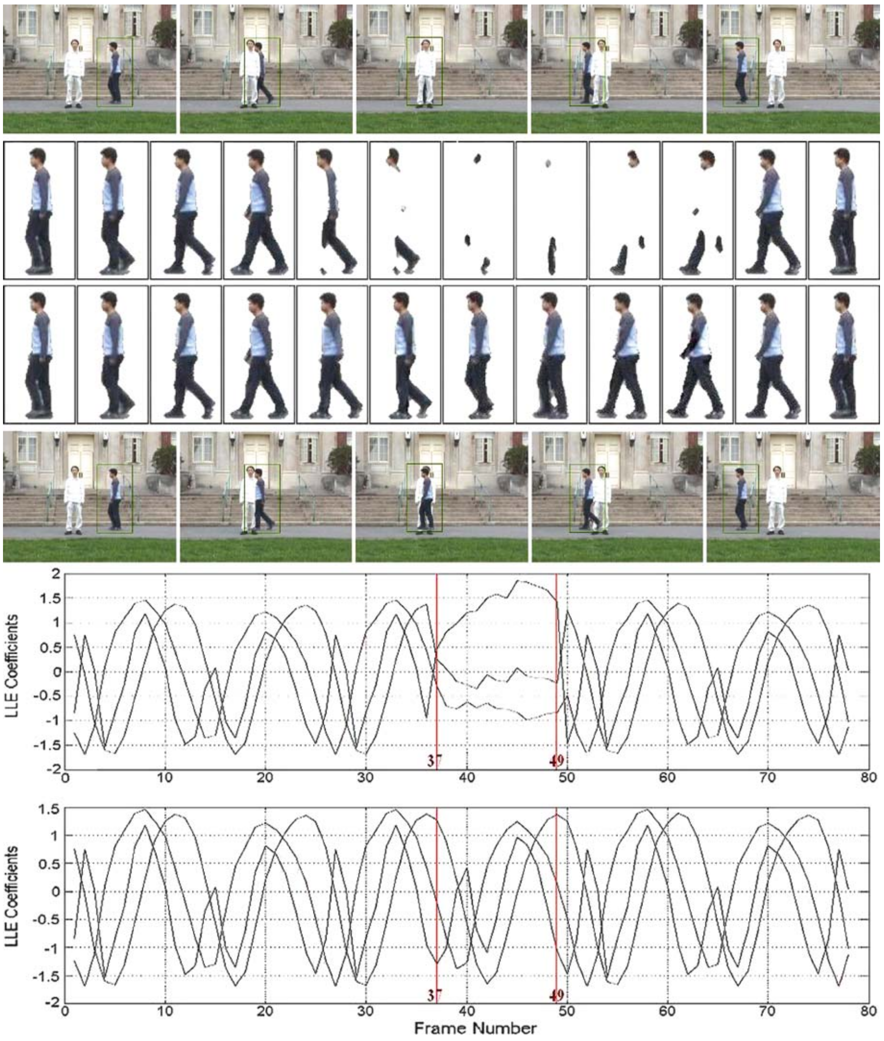
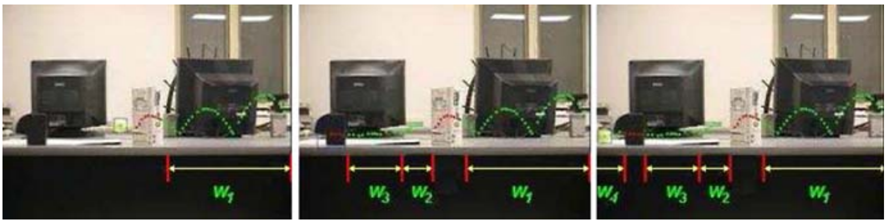


Fig. 1.12 Missing data (*second* and *fifth rows*) interpolated by rank minimization on 3D manifolds (*third* and *sixth rows*)



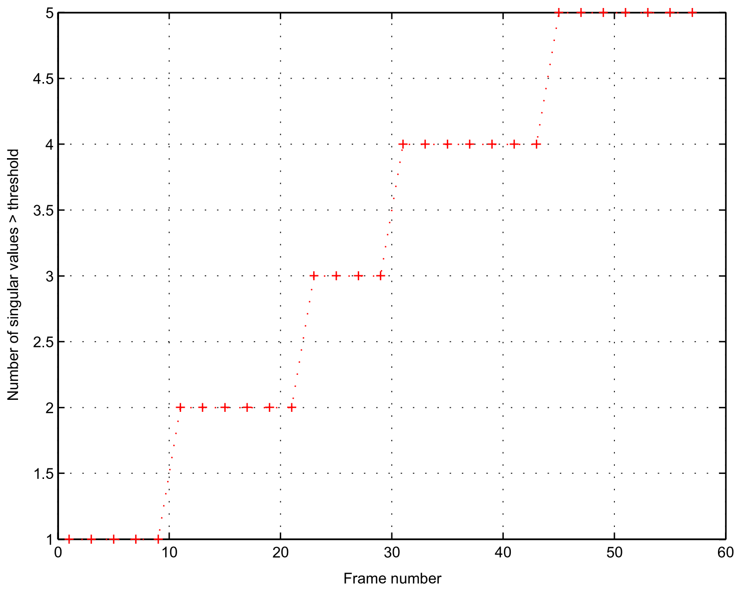


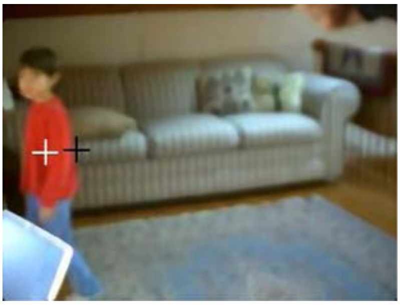
Fig. 1.13 Occluded event detection. *Top*: Dynamic data interpolation. *Bottom*: Hankel rank plot showing events

equivalent to interpolating the data so that the resulting underlying model exhibits the minimum number of jumps.

1.7 Hypothesis Testing and Data Sharing

A salient feature of the dynamics-based information extraction framework is its ability to furnish application-relevant *worst-case bounds* on distances between data and model predictions and, significantly, between nonoverlapping data streams, in terms of their respective models. These bounds lead to computationally efficient hypothesis testing techniques. Consider a data stream $\{y_k\}_{k=0}^{N-1}$ generated by an underlying model of the form:

$$y_{k+1} = \mathcal{F}[\mathbf{y}_k, \mathbf{e}_k], \qquad \mathbf{y}_k \doteq [y_k, \dots, y_{k-n}], \qquad \mathbf{e}_k \doteq [e_k, \dots, e_{k-n}] \tag{1.8}$$



Frame	13	14	15	16	17	18	19	20
Actual error	8.87	6.14	10.04	13.03	10.31	15.72	19.50	26.04
Worst case bound	13.00	15	17	19	21	23	25	27

Fig. 1.14 *Top*: Prediction (*black cross*) versus Ground Truth (*white cross*). *Bottom*: Id error

where e is a stochastic input, and $\mathcal{F}$ is the (unknown) evolution operator. Collecting all available a priori information about $\mathcal{F}$ and e (e.g., known dynamic modes and noise statistics) into sets $\mathcal{S}$ and $\mathcal{N}$ reduces the problem of finding $\mathcal{F}$ to a finite dimensional optimization via an extended Caratheodory–Fejer interpolation framework [35]. This method is *interpolatory*, hence it generates a model

$$\mathcal{F}_{id} \in \mathcal{T}(\mathbf{y}) \doteq \left\{\mathcal{F} \in \mathcal{S}\colon\ y_{k+1} = \mathcal{F}[\mathbf{y}, \mathbf{e}], \mathbf{e} \in \mathcal{N}\right\}$$

the set of all models consistent with both the a priori information and the experimental data. Since the actual (unknown) model that generated the data must also belong to $\mathcal{T}(y)$, a bound on the worst case prediction error of the identified model $\mathcal{F}_{id}$ is given by

$$\left\|\hat{\mathbf{y}} - \mathbf{y}\right\|_* \le \sup_{\mathcal{F}_1, \mathcal{F}_2 \in \mathcal{T}(\mathbf{y})} \left\|\mathcal{F}_1[\mathbf{y}, \mathbf{e}] - \mathcal{F}_2[\mathbf{y}, \mathbf{e}]\right\|_* = \mathcal{D}\left[\mathcal{T}(\mathbf{y})\right] \tag{1.9}$$

where $\|.\|_*$ is a suitable norm and $\mathcal{D}(.)$ denotes the diameter of the set $\mathcal{T}(\mathbf{y})$. When the sets $\mathcal{S}$ and $\mathcal{N}$ are convex, computing this bound reduces to a convex optimization problem [17, Lemma 10.3]. Note that these bounds are computed only once and remain valid as long as the underlying dynamics do not change.

Figure 1.14 compares the actual and upper bound of the error in a human tracking application. In this experiment the measured position in frame 12 was propagated forward using the identified dynamics and the bounds computed by solving a single linear programming problem. If other targets with similar dynamics or photometric properties are present, trackers can safely discard candidates falling outside these bounds.

In addition to the **low cost data gating** illustrated above, the worst case bounds provided by $\mathcal{D}[\mathcal{T}(\mathbf{y})]$ can be used to robustly assess the distance between non-overlapping data streams. The idea is to measure this quantity in terms of the distance between the corresponding (model) consistency sets. Intuitively, two partial data streams are considered to be manifestations of the same underlying process if they can be generated by the same dynamic model. The introduction of the consistency set in this context allows for taking into consideration data-quality issues

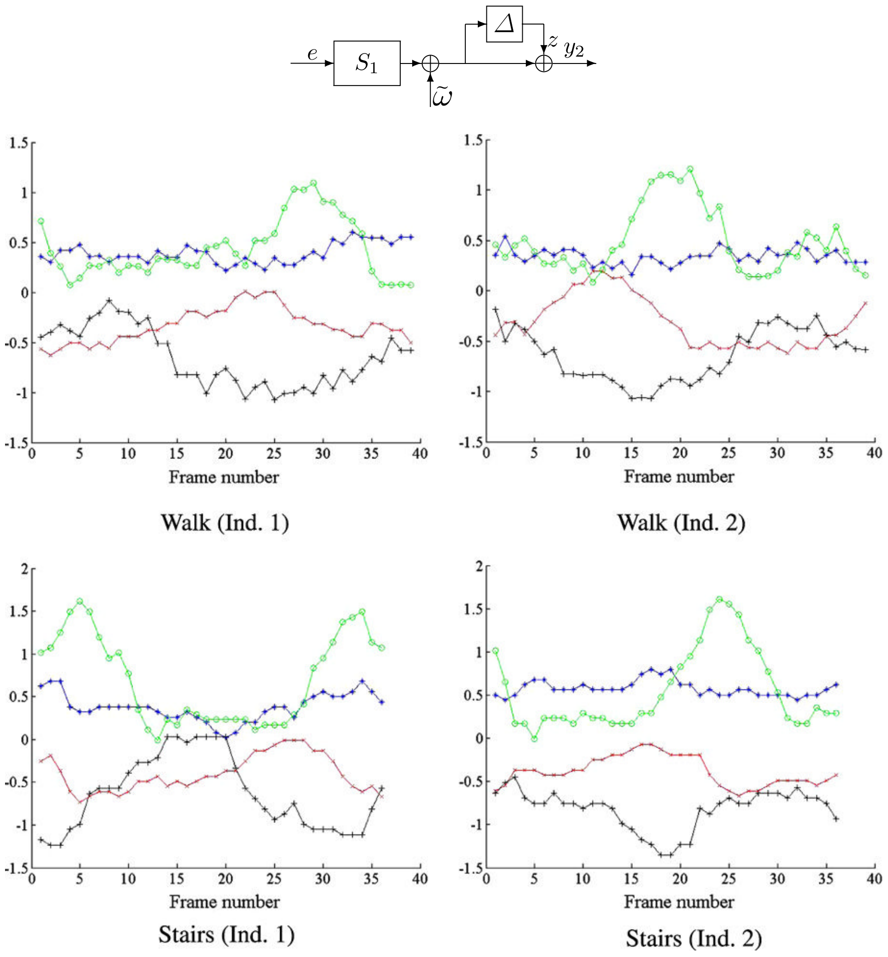


Fig. 1.15 *Top*: Distance between data streams as a model (in)validation problem. *Bottom*: sample joint traces for different activities

(relatively few observations, corrupted by noise) and a priori information. Computing the exact distance between consistency sets is costly, but it can be relaxed to the model (in)validation form shown in Fig. 1.15 (top). Given data streams y_1, y_2, the idea is to identify a nominal model S_1 associated with y_1 and a deformation operator Δ so that the pair (S_1, Δ) generates y_2. As shown in [36], computing the minimum norm $\gamma_{\min} \doteq \min \|\Delta\|_\infty$ over the set of all operators with this property reduces to a convex linear matrix inequality optimization problem. Thus, the value $\gamma_{\min}$ provides a computationally tractable upper bound on the distance between consistency sets.

This idea is illustrated next, using as an example the problem of gait classification. The experimental data listed in Table 1.2 and plotted in Fig. 1.15 (bottom), consists of 30 vector sequences, taken from 5 different persons, named A, B, C, D,

Table 1.2 Experimental data

Person	Walking	Running	Staircase
A	1, 2	16 to 18	25 to 27
B	3 to 8	11 to 15	21 to 24
C	9, 10	none	28 to 30
D	none	19	none
E	none	20	none

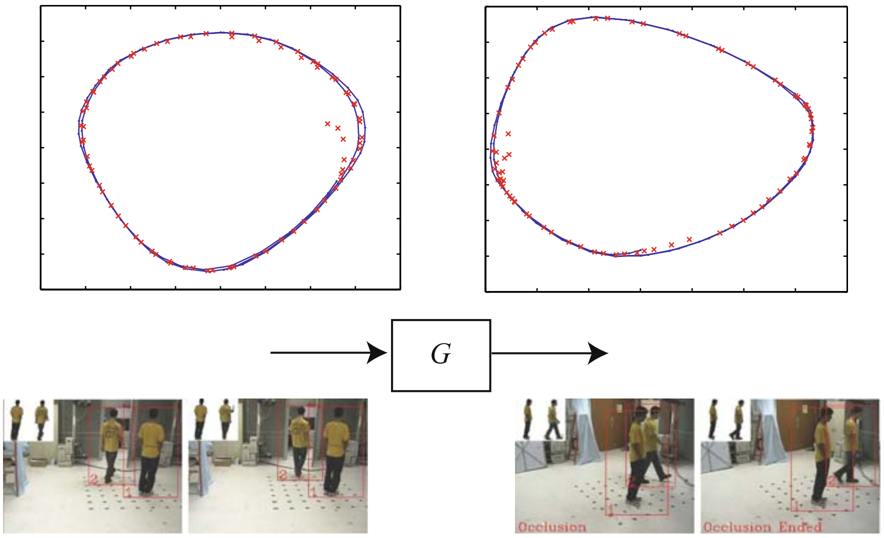


Fig. 1.16 Mapping manifolds between 2 sensors used to recreate an occluded person

and E. Each sequence contains measurements of the angles of the shoulder, elbow, hip, and knee joints of a person walking, running, or walking up a staircase. For illustrative sake, these sequences are numbered from 1 to 30 so that the first 10 correspond to walking, the second set of 10 to running and the third set of 10 to walking up a staircase.

Table 1.3 shows the distance from each data set to the dynamic model representing each activity. For each sequence, these nominal models were obtained by first finding a model associated with each of the remaining sequences and then selecting as representative of each class the model closest to its center (e.g., the one solving $\min_i \max_j \|S_i - S_j\|_\infty$). Note that nearest neighbor classification using this metric can successfully recognize 25 out of the 27 sequences under consideration; it only confuses 2 sequences, (y_{26} and y_{29}, belonging to persons A and C walking up a staircase) as walking sequences. The failure is due to the fact that in these instances the experimental data record is too short to disambiguate between activities.

Table 1.3 Right: distance between data and models (in the $\mathcal{H}_\infty$ norm). In each row † denotes the model whose output best matches the given sequence

Sequence	S_{walk}	S_{run}	S_{stair}
y_1	$0.1743^\dagger$	0.6758	0.5973
y_2	$0.2333^\dagger$	0.5818	0.2427
y_3	$0.0305^\dagger$	0.6843	0.6866
y_4	$0.0410^\dagger$	0.6217	0.5305
y_5	$0.0819^\dagger$	0.6915	0.6069
y_6	$0.0001^\dagger$	0.6879	0.7688
y_7	$0.0900^\dagger$	0.6892	0.9188
y_8	$0.2068^\dagger$	0.6037	0.7883
y_9	$0.0001^\dagger$	0.6926	0.6028
y_{11}	0.9265	$0.3415^\dagger$	1.0000
y_{12}	0.9676	$0.2452^\dagger$	0.9325
y_{13}	1	$0.0002^\dagger$	0.9323
y_{14}	1	$0.0002^\dagger$	0.9903
y_{15}	1	$0.0002^\dagger$	0.8999
y_{16}	1	$0.0005^\dagger$	0.5707
y_{17}	0.9220	$0.0532^\dagger$	0.5437
y_{18}	1	$0.0004^\dagger$	0.6961
y_{19}	1	$0.3545^\dagger$	0.8374
y_{21}	0.9631	0.5002	$0.3174^\dagger$
y_{22}	0.7952	0.4122	$0.0577^\dagger$
y_{23}	0.7215	0.4089	$0.0936^\dagger$
y_{24}	0.8499	0.4456	$0.0805^\dagger$
y_{25}	0.7252	0.5928	$0.3962^\dagger$
y_{26}	$0.6828^\dagger$	0.7127	0.8827
y_{27}	0.5553	0.5818	$0.4682^\dagger$
y_{28}	0.2650	0.6801	$0.1699^\dagger$
y_{29}	$0.0391^\dagger$	0.6102	0.1470

In the case of **distributed data sources,** the high costs (both in bandwidth and computational cost) entailed in sharing information can be avoided by (i) associating to each source a set of intrinsic coordinates on a low dimensional manifold, and (ii) using robust identification techniques [36] to identify dynamic models for the mappings between the projections of the different local data sources (e.g., sensors) onto the respective manifolds (see Fig. 1.16). Then only these low dimensional projections need to be exchanged between nodes, and each node can reconstruct the data observed by other nodes, simply by applying the interconnecting models. Figure 1.16 shows an application of these ideas to the problem of tracking and disambiguating two virtually identical targets.

1.8 Conclusions

Arguably, one of the hardest challenges entailed in exploiting actionable information sparsely encoded in high volume data streams is the development of scalable, tractable methods capable of dealing with the overwhelming volume of data [37]. Recent work (manifold embedding [2], compressive sensing, [38–40]) have led to substantial progress in addressing this issue. However, these methods stop short of fully exploiting the gap between data dimensionality and the rank of the *dynamical* system underlying the data record.

As shown in this chapter, the use ***dynamic models as an information encoding paradigm***, can lead to both, substantial dimensionality reduction and computationally attractive algorithms for data extraction/interpretation. Dynamic structures can be tractably discovered from the data in a way which leverages their inherently lower dimensionality. One key feature is the ability of dynamic representations to produce quantifiable measures of uncertainty as *provable error bounds* on the validity of the data interpretation suggested by the model. Another is their relative computational simplicity: in many cases postulating the existence of such a model and associated invariants (e.g., model order) is enough to develop computationally attractive, robust solutions to problems such as segmentation, interpolation, and event detection. We believe that these techniques hold the key to render practical several applications, ranging from self-aware environments to automatic discovery of co-regulated genes, that are currently at the proof-of-concept stage, and where the major roadblock is precisely the lack of techniques to robustly handle the extremely high volume of (often relatively low quality) data.

Acknowledgements The authors are indebted to Professor Uri Alon, Weizmann Institute, and Dr. Alon Zaslaver, Caltech, for providing the diauxic shift experimental data used in Figs. 1.1(c), 1.5(c), and 1.7. The research that generated this data was supported by the Kahn Family Foundation at the Weizmann Institute of Science. The authors are also grateful to Professor Stefano Soatto, UCLA, for providing the data used in the gait classification example shown in Fig. 1.15.

References

1. Zaslaver, A., Bren, A., Ronen, M., Itzkovitz, S., Kikoin, I., Shavit, S., Liebermeister, W., Surette, M.G., Alon, U.: A comprehensive library of fluorescent transcriptional reporters for Escherichia coli. Nat. Methods **3**, 623–628 (2006)
2. Lee, J.A., Verleysen, M.: Nonlinear Dimensionality Reduction. Springer, Berlin (2007)
3. Costeira, J., Kanade, T.: A multibody factorization method for independently moving objects. Int. J. Comput. Vis. **29**, 159–179 (1998)
4. Vidal, R., Ma, Y., Soatto, S., Sastry, S.: Two–view multibody structure from motion. Int. J. Comput. Vis. **68**, 7–25 (2006)
5. Zelnik-Manor, L., Irani, M.: Degeneracies, dependencies and their implications in multi-body and multi-sequence factorization. In: Proc. of the 2003 IEEE Conf. on Computer Vision and Pattern Recognition, pp. 287–293 (2003)
6. Ma, Y., Vidal, R.: A closed form solution to the identification of hybrid arx models via the identification of algebraic varieties. In: Hybrid Systems Computation and Control, pp. 449–465 (2005)

7. Bemporad, A., Garulli, A., Paoletti, S., Vicino, A.: A bounded-error approach to piecewise affine system identification. IEEE Trans. Autom. Control **50**, 1567–1580 (2005)
8. Saul, L.K., Roweis, S.T.: Think globally, fit locally: unsupervised learning of low dimensional manifolds. J. Mach. Learning Res. **4**, 119–155 (2003)
9. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2323 (2000)
10. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**, 1373–1396 (2003)
11. Donoho, D.L., Grimes, C.E.: Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. In: Proceedings of the National Academy of Arts and Sciences, vol. 100, pp. 5591–5596 (2003)
12. Weinberger, K.Q., Sha, F., Saul, L.K.: Learning a kernel matrix for nonlinear dimensionality reduction. In: Proc. of the 2004 International Conference on Machine Learning. ACM, New York (2004)
13. Weinberger, K.Q., Saul, L.K.: Unsupervised learning of image manifolds by semidefinite programming. In: Proc. of the 2004 IEEE Conf. on Computer Vision and Pattern Recognition, pp. 988–995 (2004)
14. Ozay, N., Sznaier, M., Lagoa, C., Camps, O.: A sparsification approach to set membership identification of a class of affine hybrid system. In: Proc. 47th IEEE Conf. Dec. Control (CDC), pp. 123–130 (2008)
15. Sznaier, M., Lagoa, C., Mazzaro, M.C.: An algorithm for sampling balls in $\mathcal{H}_\infty$ with applications to risk–adjusted performance analysis and model (in)validation. IEEE Trans. Autom. Control **50**, 410–416 (2005)
16. Sznaier, M., Lagoa, C., Ma, W.: Risk adjusted identification of a class of nonlinear systems. In: Proc. 46 IEEE Conf. Dec. Control., pp. 5117–5122 (2007)
17. Sánchez Peña, R., Sznaier, M.: Robust Systems Theory and Applications. Wiley, New York (1998)
18. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: a factorization method. Int. J. Comput. Vis. **9**, 137–154 (1992)
19. Xiao, J., Chai, J., Kanade, T.: A closed-form solution to non-rigid shape and motion recovery. In: Proc. of the 8th European Conference on Computer Vision (ECCV 2004) (2004)
20. Vidal, R., Hartley, R.: Motion segmentation with missing data using powerfactorization and gpca. In: Proc. of the 2004 IEEE Conf. on Computer Vision and Pattern Recognition, vol. 2, pp. 310–316 (2004)
21. Fazel, M., Hindi, H., Boyd, S.P.: A rank minimization heuristic with application to minimum order system approximation. In: Proceedings of the 2001 American Control Conf. (2001), vol. 6, pp. 4734–4739. AACC, Washington (2001)
22. Lobo, M., Fazel, M., Boyd, S.: Portfolio optimization with linear and fixed transaction costs. Ann. Oper. Res. **152**, 376–394 (2007)
23. Gargi, U., Kasturi, R., Strayer, S.H.: Performance characterization of video-shot-change detection methods. IEEE Trans. Circuits Syst. Video Technol. **10**, 1–13 (2000)
24. Yuan, J., Wang, H., Xiao, L., Zheng, W., Li, J., Lin, F., Zhang, B.: A formal study of shot boundary detection. IEEE Trans. Circuits Syst. Video Technol. **17**, 168–186 (2007)
25. Osian, M., Van Gool, L.: Video shot characterization. Mach. Vis. Appl. **15**, 172–177 (2004)
26. Petrovic, N., Ivanovic, A., Jojic, N.: Recursive estimation of generative models of video. In: Proc. of the 2006 IEEE Conf. on Computer Vision and Pattern Recognition, pp. 79–86 (2006)
27. Lu, L., Vidal, R.: Combined central and subspace clustering for computer vision applications. In: Proc. of the 2006 International Conference on Machine Learning, pp. 593–600 (2006)
28. Rand, W.: Objective criteria for the evaluation of clustering methods. J. Am. Stat. Assoc. **66**, 846–850 (1971)
29. Doretto, G., Chiuso, A., Wu, Y., Soatto, S.: Dynamic textures. Int. J. Comput. Vis. **51**, 91–109 (2003)
30. Chan, A.B., Vasconcelos, N.: Mixtures of dynamic textures. In: Proc. of the 2005 International Conference on Computer Vision (ICCV), vol. 1, pp. 641–647 (2005)

31. Cooper, L., Liu, J., Huang, K.: Spatial segmentation of temporal texture using mixture linear models. In: Proceedings of the Dynamical Vision Workshop in the International Conference of Computer Vision, pp. 142–150 (2005)
32. Ghoreyshi, A., Vidal, R.: Segmenting dynamic textures with using descriptors, arx models and level sets. In: Proceedings of the Dynamical Vision Workshop in the European Conference of Computer Vision, pp. 127–141 (2006)
33. Vandenberghe, L., Boyd, S.P.: Semidefinite programming. SIAM Rev. **38**(1), 49–95 (1996)
34. Fazel, M., Hindi, H., Boyd, S.P.: Log-det heuristic for matrix rank minimization with applications to Hankel and euclidean distance matrices. In: Proceedings of the 2003 American Control Conf., vol. 3, pp. 2156–2162. AACC, Washington (2003)
35. Sznaier, M., Mazzaro, C., Camps, O.I.: Open-loop worst-case identification of nonschur plants. Automatica **39**, 1019–1025 (2003)
36. Chen, J., Gu, G.: Control Oriented System Identification, an $\mathcal{H}_\infty$ Approach. Wiley, New York (2000)
37. Donoho, D.: High-dimensional data analysis: The curses and blessings of dimensionality. In: AMS Conf. on Math Challenges of the 21st Century (2000). Plenary presentation
38. Candes, E., Romberg, J., Tao, T.: Stable signal recovery from incomplete and inaccurate measurements. Commun. Pure Appl. Math. **59**, 1207–1223 (2006)
39. Donoho, D.: Compressed sensing. IEEE Trans. Information Theory **52**, 1289–13006 (2006)
40. Carin, L., Liu, D., Guo, B.: In situ compressive sensing. Inverse Probl. **24**, 015023 (2008)

Chapter 2
Information Trajectory of Optimal Learning

Roman V. Belavkin

Summary The paper outlines some basic principles of geometric and nonasymptotic theory of learning systems. An evolution of such a system is represented by points on a statistical manifold, and a topology related to information dynamics is introduced to define trajectories continuous in information. It is shown that optimization of learning with respect to a given utility function leads to an evolution described by a continuous trajectory. Path integrals along the trajectory define the optimal utility and information bounds. Closed form expressions are derived for two important types of utility functions. The presented approach is a generalization of the use of Orlicz spaces in information geometry, and it gives a new, geometric interpretation of the classical information value theory and statistical mechanics. In addition, theoretical predictions are evaluated experimentally by comparing performance of agents learning in a nonstationary stochastic environment.

2.1 Introduction

The ability to learn and adapt the behavior with respect to changes in the environment is arguably one of the most important characteristics of intelligent systems. The study of learning algorithms has become an active area of research in artificial intelligence closely related to different areas of mathematics, cognitive science, psychology and neurobiology. The optimization and information theories are of particular importance. This paper presents a geometric approach to the evolution of a learning system that is inspired by information geometry [1, 7], and it is closely related to the information value theory of Stratonovich [23].

Learning can be considered as a process of incorporating new information to improve the performance of a system. Thus, learning by this definition assumes incomplete information. On the other hand, optimization is the main motivation for learning. This duality principle of the utility and information in learning systems is

R.V. Belavkin (✉)
Middlesex University, London NW4 4BT, UK
e-mail: R.Belavkin@mdx.ac.uk

fundamental for the theory presented [4]. We now briefly outline the main principles of the classical methods and their limitations.

Without uncertainty, optimization problems are the problems of finding the extrema (minimum or maximum) of some functions, which are called utilities, costs or fitness functions depending on the particular convention. These functions represent someone's preference relation on the underlying choice set, which can be the set of lottery prizes, errors of an estimation algorithm, space-time evolutions of a dynamical system and so on. The utility function may incorporate multiple constraints and objectives in a single Lagrange function, the extreme values of which are used to find solutions to optimization problems in variational analysis and the theory of optimal control. In particular, the maximum principle [15] defines the necessary conditions of optimality in canonical form of the Euler system of differential equations. This approach to optimal control is often referred to as the *trajectory* approach. An alternative is the dynamic programming approach [6] that is solved by partial differential equations (i.e., the Hamilton–Jacobi–Bellman equation), and it is often referred to as the *wavefront* approach.

Under uncertainty, the problem is usually formulated using methods of probability theory. The elements of a choice set are drawn stochastically as the outcomes of some lottery that is represented by a probability measure over the choice set. The idea is then to 'play' a lottery that maximizes the utility (or minimizes the risk) on average. Maximization of conditional expected utility is used in Bayesian approach to stochastic optimal control and estimation [25, 26], and sequential stochastic optimization is usually solved via dynamic programming [6]. A significant development in this area was the theory of conditional Markov processes [22] that allows one to reduce the number of variables for additive utility functions and represent the space-time evolution of the system by stochastic differential equations [21].

These methods of optimal control have been also used in the design of intelligent and adaptive systems [11, 24]. One of the main challenges, however, is that these systems operate with incomplete information, and thus optimality of the described above methods (which assume a given model of the system) is no longer guaranteed. However, often one can consider asymptotic optimality under certain assumptions. Some of these assumptions are:

1. The limits of empirical distributions exist.
2. Data is obtained from independent and identically distributed samples.
3. The 'true' distributions are stationary.

The first assumption allows one to pick some priors and then update them using empirical frequencies [16]. If these frequencies converge to the 'true' distributions, then asymptotically the system becomes optimal. The first assumption, however, depends on the second (the weak law of large numbers). Its last part (identically distributed) is equivalent to the third assumption. It is now becoming increasingly apparent that these basic assumptions may be violated in learning systems.

Indeed, the last assumption may not be valid if the agents' interaction with the environment changes the underlying distributions (i.e., there is a dependency between the agents and their environment). Dropping the stationary assumption, however, is not a problem because the Bernoulli theorem is then replaced by the Poisson theorem, where the limit of empirical frequencies is the average of the distributions. Much more important, however, is the assumption of independent trials. Note that this does not mean that the evolution of the system is a sequence of independent events. This can be a Markov or even the conditional Markov process. However, if the Markov transitional probabilities are not known, then the samples updating the empirical transitional frequencies are assumed to be independent.

To see that this assumption can be violated in a learning system, one has to consider the exchangeability concept, introduced by Bruno de Finetti [8]. Exchangeable sequences are such that their joint distributions are invariant under permutation of the sequence order. For finite sequences, there are more exchangeable distributions than independent, and they coincide only when sequences are infinite (the de Finetti's theorem). Thus, if the sequence is not exchangeable, then it is also not independent. Now, learning is a process when new information, obtained from samples, is used to adapt the system in order to improve the performance. This means that the order, in which data is sampled and used, may be important, and therefore learning sequences are generally not exchangeable and are not independent. Without this condition, the first assumption is too strong, and therefore the limit may not exist in the traditional sense (i.e., as convergence in the laws of large numbers). As an illustration of this argument, consider a cat learning the distribution of mice. What is the limit of this distribution, if the mice also learn the distribution of the cat?

The problem of incomplete prior information should not be confused with the complexity issues arising in many optimization problems, such as the 'curse of dimensionality' in sequential optimization. Given unlimited computational power, one theoretically could use the dynamic programming approach to optimize decisions even over infinite sequences, and some researchers suggested that it could resolve problems of incomplete information, such as the exploration-exploitation dilemma [24]. This idea, however, contradicts the statistical nature of information, because new information can only be obtained through measurements, and it can only be lost in transmission (such as computation). The dynamic programming method is a technique for optimization of utility over sequences, but it does not address the problem of incomplete prior information.

Problems of optimization with information constraints have been considered in information theory [12, 13, 18, 19] leading to optimal solutions in the form of exponential family of distributions. The dual problem of entropy maximization with linear constraints was considered in statistical mechanics [9, 10]. The information value theory was developed as an application of these results to cybernetics and statistical decisions [23]. It was shown also that the results of this theory hold for a wide class of the entropically and informationally stable systems. In particular, this class includes sequences of nonstationary, nonindependent random variables. These results, therefore, can be applied to a more general class of learning systems than those considered by the traditional methods.

This paper presents geometric approach to the analysis of learning systems, and describes also a simple experiment as an illustration. The theory defines nonasymptotic optimality conditions relative to available information, and defines the evolution of an optimal learning by a trajectory on the statistical manifold. The analysis has similarities with the use of Orlicz spaces and exponential statistical manifolds in non-parametric information geometry to describe systems of bounded entropy [14]. Here, however, the theory is developed for more general class of convex functionals representing information. The corresponding spaces are quasi-pseudo-metric generalizing normed spaces. The approach leads to a nonasymptotic and nonparametric theory for optimization of the evolution of a learning system by empirical constraints. Some examples are closely related to information theory and statistical mechanics. The optimal trajectory also defines the utility and information bounds of a learning system, which are given by the analogue of a gradient theorem for path integrals in conservative vector fields.

2.2 Topology and Geometry of Learning Systems

In this section, we recall some elements of the theory of optimal choice under uncertainty [25] and information value theory [23] that are relevant to our representation of the learning systems. Then we define a topology on a functional space related to information dynamics in such systems.

2.2.1 Problem Statement and Basic Concepts

Fundamental concept in the theory of rational choice is the *preference* relation on a set Ω, which is a complete and transitive binary relation $\lesssim \subseteq \Omega \times \Omega$ (i.e., total pre-order). Subset of symmetric pairs $\sim \subseteq \lesssim$ is the equivalence relation, and the set of antisymmetric pairs $< \subseteq \lesssim$ is a partial order on Ω. The quotient set $\Omega/\sim$ is totally ordered. We assume that $\Omega/\sim$ can be embedded into the extended real line $\overline{\mathbb{R}} \equiv \mathbb{R} \cup \{\pm\infty\}$. In this case, the preference relation can be represented by a *utility* function $u : \Omega \to \overline{\mathbb{R}}$ (i.e., $\omega_1 \lesssim \omega_2$ iff $u(\omega_1) \leq u(\omega_2)$). The rational choice (optimization) corresponds to maximization of the utility.

Under uncertainty, one considers probability measures $y : \mathfrak{R} \to \mathbb{R}$ on a σ-algebra $\mathfrak{R}(\Omega) \subseteq 2^{\Omega}$. Probability measures can be interpreted as lotteries over the choice set $(\Omega, \lesssim)$. For example, the Dirac δ-measures ($\delta_\omega(d\omega) = 1$ if $\omega \in d\omega$; 0 otherwise) correspond to the elements $\omega \in \Omega$ observed with certainty. Other probability measures are unique convex combinations of the δ-measures, and therefore the set $\mathcal{P}(\Omega)$ of *all* probability measures on $\mathfrak{R}(\Omega)$ is a simplex in some linear space L—a convex hull of the set Δ of all δ-measures on Ω ($\mathcal{P}(\Omega)$ is a Choquet simplex if Ω is infinite).

The set of all probability measures, $\mathcal{P}(\Omega)$, will be referred to as *statistical manifold*, as in information geometry, and it is the set of *all* lotteries. The choice problem under uncertainty requires an extension of the preference relation $(\Omega, \lesssim)$ onto $\mathcal{P}(\Omega)$. This extension should be compatible with $(\Omega, \lesssim)$ in the following sense $(\Delta, \lesssim) = (\Omega, \lesssim)$. One such extension is given by the *expected utility*: $E_y\{u\} = \int_\Omega u(\omega)\, dy(\omega)$. Thus, measure p is preferred to q if and only if $E_p\{u\} \geq E_q\{u\}$.

Furthermore, a fundamental result of game theory states that expected utility is the only representation that also satisfies two additional axioms: continuity and substitution independence [25].

The main difference problems of optimization under uncertainty, described above, and the learning problems is that the latter are concerned with incomplete information. In particular, this means that the probability measures on the choice set are not known exactly, or in other words that the learner does not know precisely which lotteries he plays. For an agent with limited resources, this presents a dilemma of collecting more information (exploration) or using already available information for optimal control (exploitation).

The traditional approach to solving this dilemma is to treat it as a statistical problem of estimating unknown parameters $\theta \in \mathbb{R}^m$ of some known family of distributions $y(d\omega \mid \theta)$. Points θ of the parameter space $\mathbb{R}^m$ define points on the statistical manifold, and the corresponding relations and metrics are the subject of information geometry [1, 7]. The approach taken in this work is similar to infinite-dimensional nonparametric information geometry [14], where probability measures are studied directly in the corresponding functional space. This allows for considering all families of measures and to derive nonasymptotic optimality conditions using the conjugate duality theory [17]. The topologies will be defined using quasi-norms and quasi-metrics related to information constraints, which is more appropriate for describing learning systems, and it is a generalization of the standard approach using normed spaces (i.e., Orlicz spaces in [14]).

Observe that optimization under uncertainty is concerned with at least two types of real functions on the choice set $(\Omega, \lesssim)$—utilities and probability measures. Moreover, for a fixed utility function, the expected utility is a linear functional on measures; for a fixed measure, the expected utility is a linear functional on utility functions. Thus, measures and utility functions can be represented by elements of dual linear spaces L and L^*, where the expected utility implements the pairing $\langle \cdot, \cdot \rangle : L^* \times L \to \mathbb{R}$:

$$\langle x, y \rangle = \int_\Omega x(\omega)\, dy(\omega) \tag{2.1}$$

Note that because we only deal with preference relations that have a utility representation, set $\Omega/\sim$ is a separable, complete, metrizable space, and therefore we only need to consider Radon measures. Such measures are finite on compact subsets $\Omega_c \subseteq \Omega$, and they are in one-to-one correspondence with linear functionals $y(f) = \langle f, y \rangle$ on the space $\mathcal{C}_c(\Omega)$ of continuous functions with compact support. Thus, we associate measures with nonnegative elements of space $L \equiv \mathcal{C}_c^*(\Omega)$, dual of $\mathcal{C}_c(\Omega)$. Utility functions are the elements of its second dual $L^* \equiv \mathcal{C}_c^{**}(\Omega)$.

The theory presented is also largely inspired by information value theory [23]. Consider two points on the statistical manifold, y_0 (prior) and y (posterior), associated with an observation of some random event. The associated change $\langle x, y - y_0 \rangle$ of the expected utility represents the value of this event, and it is different for agents with different utility functions $x \in L^*$. On the other hand, information is usually represented by some functional $F : L \to \overline{\mathbb{R}}$ as a divergence of y from fixed point

y_0 on the statistical manifold, and it does not take the utility into account. Thus, different $y \in L$ with the same divergence $F(y) = I$ may have different values for the agent. The *value of information* amount $I \in \mathbb{R}$ is defined as the maximum of the expected utility subject to information constraint $F(y) \leq I$:

$$U(I) := \sup\big\{\langle x, y\rangle : F(y) \leq I\big\} \tag{2.2}$$

Note that the original definition in [23] is more specific using Shannon information for $F(y)$. Clearly, an optimization of information dynamics in a learning system should be closely related to information value—the optimal system should adapt to learn only the most valuable information. We now define a topology related to information value to facilitate the analysis of such systems.

2.2.2 Asymmetric Topologies and Gauge Functions

Let L and L^* be a dual pair of linear spaces over the field $\mathbb{R}$ with bilinear form $\langle\cdot,\cdot\rangle : L^* \times L \to \mathbb{R}$ separating L and L^*: $\langle x, y\rangle = 0, \forall x \in L^*$ implies $y = 0 \in L$, and $\langle x, y\rangle = 0, \forall y \in L$ implies $x = 0 \in L^*$. We shall define topologies on L and L^* that are compatible with respect to the pairing $\langle\cdot,\cdot\rangle$, but subbases of these topologies will be formed by systems of neighborhoods of zero that are generally nonbalanced sets (i.e., $y \in M$ does not imply $-y \in M$). Thus, the spaces may fail to be topological vector spaces. The gauge functions will define quasi-norms and quasi-metrics (i.e., nonsymmetric generalizations of a norm and a metric). The main motivation for this asymmetry is to avoid nonmonotonic operations on functions, such as $x \mapsto |x|$.

First, we recall some properties that depend only on the pairing $\langle\cdot,\cdot\rangle$, and not on particular topologies chosen. Nonzero $x \in L^*$ are in one-to-one correspondence with hyperplanes $H := \{y \in L : \langle x, y\rangle = \alpha\} \subset L$, $0 \notin H$, and inequality $\langle x, y\rangle \leq \alpha$ defines a closed half-space. The intersection of all closed half-spaces containing set $M \subset L$ is a *convex closure* of M denoted by $\operatorname{co} M$. Set M is a *closed convex* set if $M = \operatorname{co} M$. The *polar* of M is

$$M^* := \big\{x \in L^* : \langle x, y\rangle \leq 1,\ \ y \in M\big\}$$

The polar set is always closed, convex and $0 \in M^*$. Also, $M^{**} = \operatorname{co}[M \cup \{0\}]$, and $M = M^{**}$ if and only if M is closed, convex and $0 \in M$. Without loss of generality, we shall assume $0 \in M$.

Set M is called *absorbing* if for each $y \neq 0 \in L$ there exists $\beta > 0$ such that $y \in \beta M$; set N is called *bounded* if $N \subset \beta M$ for all $\beta \geq \varepsilon$ and some $\varepsilon > 0$. Set M is absorbing if and only M^* is bounded (to see this, observe that $y \neq 0$ are in one-to-one correspondence with closed half-spaces in L^*).

Given a closed convex set $M \subset L$ absorbing with respect to $0 \in M$, the collection of sets $\mathfrak{M} := \{\beta M : \beta > 0\}$ is the subbasis of closed neighborhoods of zero uniquely defining a topology on L. If in addition M is bounded, then the polar $M^* \ni 0$ is also absorbing, and the collection $\mathfrak{M}^* := \{\beta^{-1} M^* : \beta^{-1} > 0\}$ is the subbasis of the polar topology on L^*.

Given set $M \subset L$, the *gauge* function $p_M : L \to \overline{\mathbb{R}}$ is defined as

$$p_M(y) := \inf\{\beta > 0 : y \in \beta M\}, \quad p_M(0) := 0$$

If M is absorbing with respect to $0 \in M$, then $p_M(y) < \infty$ for all $y \in L$, and if M is bounded, then $p_M(y) = 0$ only if $y = 0$. The gauge is positively homogeneous function of the first degree, $p_M(\lambda y) = \lambda p_M(y)$, $\lambda > 0$, and if M is convex, then it is also subadditive, $p_M(y_1 + y_2) \leq p_M(y_1) + p_M(y_2)$. Thus, the gauge of an absorbing convex set satisfies all axioms of a semi-norm apart from symmetry, $p_M(y) \neq p_M(-y)$, and therefore it is a *quasi-pseudonorm*. Function $d_M(y_1, y_2) = p_M(y_1 - y_2)$ is a *quasi-pseudometric* on L. If M is also bounded, then p_M is a *quasi-norm*, and d_M is a *quasi-metric* (i.e., $d_M(y_1, y_2) \neq d_M(y_2, y_1)$).

Gauge functions are closely related to support functions. The *support* of set M is function $h_M : L^* \to \overline{\mathbb{R}}$ defined as

$$h_M(x) := \sup\big\{\langle x, y\rangle : y \in M\big\}$$

Generally, $h_M(x) = p_{M^*}(x)$, and if M is convex, then $h_{M^*}(y) = p_M(y)$ (otherwise, $h_{M^*}(y) \leq p_M(y)$).

2.2.3 Trajectories Continuous in Information

Observe now that the value of information, defined by (2.2), is equal to support h_M of subset $M = \{y \in L : F(y) \leq I\}$ of the statistical manifold, defined by information constraint. It is common to represent information by a closed, convex functional, and therefore M is closed and convex. For the theory of convex functions, see [17, 20]. Here, we recall some basic concepts.

Convex functional $F : L \to \overline{\mathbb{R}}$ is called *proper* if its *effective domain* $\operatorname{dom} F := \{y : F(y) < \infty\}$ is nonempty and $F(y) > -\infty$. Proper convex functional is *closed* if sublevel sets $\{y : F(y) \leq \lambda\}$ are closed for each $\lambda \in \mathbb{R}$. The dual functional $F^* : X \to \overline{\mathbb{R}}$ is the Legendre–Fenchel transform of F:

$$F^*(x) := \sup\big\{\langle x, y\rangle - F(y)\big\}$$

It is always closed and convex. Closed convex functionals are continuous on the (algebraic) interior of dom F, and they have the property

$$x \in \partial F(y) \quad \Longleftrightarrow \quad \partial F^*(x) \ni y$$

where set $\partial F(y_0) := \{x \in L^* : \langle x, y - y_0\rangle \leq F(y) - F(y_0),\ \forall y \in L\}$ is called *subdifferential*, and its elements are called *subgradients* (a generalization of the Gâteaux differential and gradient). In particular, $0 \in \partial F(y_0)$ implies $F(y_0) \leq F(y)$ for all $y \in L$ (i.e., $\inf F = F(y_0)$). If $F(y)$ is strictly convex at y (or F^* is G-differentiable at $x \in L^*$), then $\partial F^*(x) = \{y\}$ for all $x \in \partial F(y)$. Consequently, if dual convex functionals are both strictly convex (or G-differentiable), then $\partial F : L \to L^*$ is a bijection. Below are examples of such dual convex functionals that are used often in information theory.

Example 1 (Relative information) Given positive $y_0 \in L$, let $F : L \to \overline{\mathbb{R}}$ be:

$$F(y) = \int_\Omega \ln \frac{y(\omega)}{y_0(\omega)}\, dy(\omega) - \int_\Omega d\big[y(\omega) - y_0(\omega)\big]$$

if y is positive, $F(0) := \int_\Omega dy_0(\omega)$, and $F(y) := \infty$ for negative y. This functional is closed, strictly convex, and its G-derivative is $F'_G(y) = \ln \frac{y}{y_0}$ on the interior of $\operatorname{dom} F$. Note that $F(y) \geq 0$ for all y, because $F'_G(y_0) = 0$ and $\inf F = F(y_0) = 0$. When y and y_0 are both probability measures, then relative information is equivalent to the Kullback–Leibler divergence [13]. Relative information can be used also to represent negative entropy or Shannon mutual information.

Example 2 The dual of relative information is the following functional

$$F^*(x) = \int_\Omega e^{x(\omega)}\, dy_0(\omega)$$

Indeed, $F'_G(y) = \ln \frac{y}{y_0} = x$, and therefore $y = y_0\, e^x = F^{*\prime}_G(x)$, which is the gradient of the above functional. It is also closed, strictly convex and positive for all $x \in L^*$. Normalization of functions $y = y_0\, e^{x(\omega)}$ corresponds to transformation $F^*(x) \mapsto \ln F^*(x)$.

If $\inf F = F(0)$, then the gauge and the support functions of set $\{y : F(y) \leq I\}$ can be computed as:

$$p_F(y) = \inf\big\{\beta > 0 : F\big(\beta^{-1} y\big) \leq I\big\} \tag{2.3}$$

$$h_F(x) = \sup\big\{\langle x, y\rangle : F(y) \leq I\big\} \tag{2.4}$$

The support function above is the gauge of the polar set, which can also be computed as $p_{F^*}(x) = \inf\{\beta^{-1} > 0 : F^*(\beta x) \leq I^*\}$.

Thus, information functional $F : L \to \overline{\mathbb{R}}$ can be used to define a topology on the statistical manifold as the collection of all elements $y \in L$, for which set $M = \{y : F(y) \leq I\}$ is absorbing (and therefore $p_F(y) < \infty$). The topology on the dual space (the space of utility functions) is the collection of $x \in L^*$ for which the polar set is absorbing (and therefore $h_F(x) < \infty$). We shall denote these topological spaces by L_F and L^*_F.

A topology related to information $I \in \mathbb{R}$ is useful for the analysis of learning systems and their dynamics. In particular, an evolution that is continuous in information is represented by a function $y = f(I)$ that maps closed sets $(-\infty, I] \subset \mathbb{R}$ into closed sets $M = \{y : F(y) \leq I\}$ on the statistical manifold. Note that such an evolution is also order-preserving (monotonic) between $(\mathbb{R}, \leq)$ and pre-order $\lesssim$ on L_F, defined by the gauge p_F. We shall refer to such an evolution of a learning system as a continuous *information trajectory*.

2.3 Optimal Evolution and Bounds

An evolution of a learning system, even if described by a continuous trajectory, may not be optimal. As mentioned earlier, an optimal evolution is the totality of points $\bar{y} \in L_F$ maximizing information value or the expected utility $\langle x, y\rangle$ subject to information constraints. Thus, $\bar{y}$ must satisfy the extrema of (2.2) (or the support function (2.4)) for a given utility. Optimal solutions are found by the standard method of Lagrange multipliers, which we present below for completeness of exposition.

Theorem 1 (Necessary and sufficient optimality conditions) *The least upper bound $U(I) = \sup\{\langle x, y\rangle : F(y) \leq I < \infty\}$ is achieved at $\bar{y}$ if and only if the following conditions are satisfied*

$$\bar{y} \in \partial F^*(\beta x), \quad F(\bar{y}) = I, \quad \beta^{-1} \in \partial U(I), \quad \beta^{-1} > 0$$

Proof The Lagrange function is $K(y, \beta^{-1}, I) = \langle x, y\rangle + \beta^{-1}[I - F(y)]$, where β^{-1} is the Lagrange multiplier corresponding to $F(y) \leq I$. Zero in the subdifferential of $K(y, \beta^{-1}, I)$ gives the necessary conditions of extrema:

$$\partial_y K\big(\bar{y}, \beta^{-1}, I\big) = x - \beta^{-1}\partial F(\bar{y}) \ni 0, \quad \Rightarrow \quad \beta x \in \partial F(\bar{y})$$
$$\partial_{\beta^{-1}} K\big(\bar{y}, \beta^{-1}, I\big) = I - F(\bar{y}) \ni 0, \quad \Rightarrow \quad F(\bar{y}) = I$$

Noting that $K(\bar{y}, \beta^{-1}, I) = U(I)$, gives $\partial_I K(\bar{y}, \beta^{-1}, I) = \partial U(I) \ni \beta^{-1}$.

Sufficient conditions are obtained by considering convexity. Because F is convex and $\langle x, \cdot\rangle$ is linear, the Lagrange function is concave for $\beta^{-1} > 0$ and convex for $\beta^{-1} < 0$. Therefore, $\bar{y} \in \partial F^*(\beta x)$ with $\beta^{-1} > 0$ defines the least upper bound of $U(I)$. □

Corollary 1 *The optimal trajectory $y = \bar{y}(I)$ is continuous in information.*

Proof The optimality condition $F(\bar{y}) = I$ implies that $\bar{y} \in \{y : F(y) \leq I\}$ for any $I \in \mathbb{R}$, and therefore $y = \bar{y}(I)$ cannot map any closed set $(\infty, I] \subset \mathbb{R}$ outside closed set $\{y : F(y) \leq I\}$ in L_F. □

Example 3 When F is the relative information from Example 1, the optimal solutions are in the exponential form

$$\bar{y}(\omega) = y_0(\omega) \exp\big\{\beta\, x(\omega) - \Psi(\beta)\big\}$$

where $\Psi(\beta) = \ln \int_\Omega e^{\beta x}\, dy_0(\omega)$ from the normalizing condition. If $y_0 = \text{const}$, then optimal function $\bar{y}$ is the canonical Gibbs distribution. When the utility function is $x = -|s|^2$ (i.e., negative squared deviation), then $\bar{y}$ is Gaussian with variance $\sigma^2 = (2\beta)^{-1}$ and $e^{\Psi(\beta)} = \sqrt{\pi\beta^{-1}} = \sigma\sqrt{2\pi}$.

The totality of optimal points $\bar{y}$ can be considered as one parameter family of distributions, where parameter $\beta \in \mathbb{R}$ is the gauge of $\bar{y}$ with respect to set $\{y : F(y) \leq 1\}$, and it can be determined from the information constraint $I \in \mathbb{R}$ ($F(y) \leq I$). Note, however, that β can also be determined from the expected utility $U = \langle x, \bar{y}\rangle$. Indeed, consider function $I(U) := \inf\{F(y) : U_0 \leq U \leq \langle x, y\rangle\}$, where $U_0 = \langle x, y_0\rangle$. Clearly, $I(U)$ is the inverse of information value $U(I)$. The Lagrange function for $I(U)$ is $K(y, \beta, U) = F(y) + \beta[U - \langle x, y\rangle]$, and the solutions are defined by

$$\bar{y} \in \partial F^*(\beta x), \quad \langle x, \bar{y}\rangle = U, \quad \beta \in \partial I(U), \quad \beta \geq 0$$

Thus, the optimal information trajectory can be parametrized by the information or by the expected utility constraints through the inverse of mappings $\beta \mapsto F(\bar{y}(\beta)) = I$ and $\beta \mapsto \langle x, \bar{y}(\beta)\rangle = U$. These mappings can be conveniently expressed by the *generalized characteristic potentials*:

$$\Phi\big(\beta^{-1}\big) := \inf\big\{\beta^{-1} I - U(I)\big\}, \qquad \Psi(\beta) := \sup\big\{\beta U - I(U)\big\}.$$

The potentials are real functions, and the extrema in their definitions are given by conditions $\beta^{-1} \in \partial U(I)$ and $\beta \in \partial I(U)$. One can show also that $\Phi(\beta^{-1}) = -\beta^{-1}\Psi(\beta)$. The parametrization is based on the following theorem.

Theorem 2 (Parametrization) *Parameter $\beta \in \mathbb{R}$ defining solutions $\bar{y}$ to problems $U = \sup\{\langle x, y\rangle : F(y) \leq I\}$ and $I = \inf\{F(y) : U \leq \langle x, y\rangle\}$ is related to the constraints $I \in \mathbb{R}$ or $U \in \mathbb{R}$ by the following relations*

$$I \in \partial\,\Phi\big(\beta^{-1}\big), \qquad U \in \partial\,\Psi(\beta)$$
$$I \in \beta\,\partial\,\Psi(\beta) - \Psi(\beta), \qquad U \in \beta^{-1}\,\partial\,\Phi\big(\beta^{-1}\big) - \Phi\big(\beta^{-1}\big)$$

Proof Consider the Legendre–Fenchel transforms of Φ and Ψ:

$$U(I) = \inf\big\{\beta^{-1} I - \Phi\big(\beta^{-1}\big)\big\}, \qquad I(U) = \sup\big\{\beta U - \Psi(\beta)\big\}$$

The extrema are satisfied when $I \in \partial\Phi(\beta^{-1})$ and $U \in \partial\Psi(\beta)$, which is the first pair of relations. Substituting them into the Legendre–Fenchel transforms gives the second pair. □

Subdifferentials in Theorem 2 are replaced by derivatives $\Phi'(\beta^{-1})$ and $\Psi'(\beta)$ if Ψ and Φ are differentiable. This is the case when $F(y)$ is strictly convex.

Example 4 When solutions $\bar{y}$ are in the exponential from (Example 3), one obtains $U = \langle x, \bar{y}\rangle = \int x\, e^{\beta x - \Psi(\beta)}\, dy_0$, and condition $U = \Psi'(\beta)$ gives

$$\Psi(\beta) = \ln \int_\Omega e^{\beta\, x(\omega)}\, dy_0(\omega)$$

The above is the *cumulant generating function* of measure y_0. Potential $\Phi(\beta^{-1}) = -\beta^{-1}\Psi(\beta)$ in this case is the *free energy*.

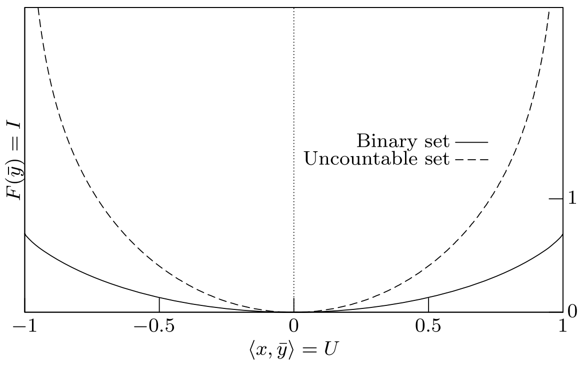


Fig. 2.1 Parametric dependencies of $I = F(\bar{y})$ on $U = \langle x, \bar{y} \rangle$ in Examples 5 and 6

Information amount is often represented by negative entropy, which corresponds to relative information F minimized at some uniform measure $y_0 = 1/|\Omega|$ (if Ω is finite) or a Lebesgue measure $dy_0 = d\omega / \int d\omega$ (if Ω is compact). Potential $\Psi(\beta)$ in these cases is

$$\Psi(\beta) = \ln \sum_{\Omega} e^{\beta x(\omega)} - \ln |\Omega| \quad \text{or} \quad \Psi(\beta) = \ln \int_{\Omega} e^{\beta x(\omega)}\, d\omega - \ln \int_{\Omega} d\omega$$

The following examples give expressions for $U(\beta)$ in two important cases.

Example 5 (Binary utility) Let $\Omega = \{\omega_1, \omega_2\}$, and $x : \Omega \to \{c - d, c + d\}$. Then using $e^{\beta(c-d)} + e^{\beta(c+d)} = 2e^{\beta c} \cosh(\beta d)$, we obtain

$$\Psi(\beta) = \beta c + \ln \cosh(\beta d), \qquad U(\beta) = c + d \tanh(\beta d)$$

Example 6 (Uncountable utility) Let Ω be compact, $x : \Omega \to [c - d, c + d] \subset \mathbb{R}$ such that $dx/d\omega = 1$. Then $\int_{\Omega} e^{\beta x(\omega)}\, d\omega = \int_{c-d}^{c+d} e^{\beta x}\, dx = 2\beta^{-1} e^{\beta c} \sinh(\beta d)$, $\int_{\Omega} d\omega = \int_{c-d}^{c+d} dx = 2d$, and we obtain

$$\Psi(\beta) = \beta c + \ln \bigl|\sinh(\beta d)\bigr| - \ln |\beta d|, \qquad U(\beta) = c + d \coth(\beta d) - \beta^{-1}$$

Functions $U = \Psi'(\beta)$ and $I = \beta \Psi'(\beta) - \Psi(\beta)$ define parametric dependency between U and I in a system evolving along the optimal information trajectory $y = \bar{y}(t)$, and it defines the following bounds on learning systems: $U(I)$ is the maximum expected utility for a given information amount; $I(U)$ is the least information amount required to achieve a given expected utility. Figure 2.1 shows $I(U)$ for functions in Examples 5 and 6 with $c = 0$ and $d = 1$.

Continuity in information, introduced earlier, allows us to consider path integrals of expected utility and information along a continuous trajectory. The upper and lower bounds on these quantities can be expressed in the following convenient form [5]. Here, we assume that Ψ and Φ are differentiable.

Theorem 3 (Optimal bounds) *Let $y = y(t)$, $t \in [t_1, t_2]$ be a continuous information trajectory of a learning system such that information $F(y) = I(t)$ and expected utility $\langle x, y\rangle = U(t)$ are increasing functions. Then*

$$\int_{y_1}^{y_2} \langle x, y\rangle \, dy \leq \Psi(\beta_2) - \Psi(\beta_1)$$

$$\int_{y_1}^{y_2} F(y)\, dy \geq \Phi\big(\beta_1^{-1}\big) - \Phi\big(\beta_2^{-1}\big)$$

where $y_1 = y(t_1)$, $y_2 = y(t_2)$, and β_1, β_2 are determined from $I(t_1)$, $I(t_2)$ or $U(t_1)$, $U(t_2)$ using functions $\beta^{-1} = (\Phi')^{-1}(I)$ or $\beta = (\Psi')^{-1}(U)$, respectively.

Proof The first path integral is bounded above by a path integral along the optimal information trajectory $y = \bar{y}(t)$. Similarly, the second integral is bounded below. These path integrals exist, because the optimal trajectory is continuous in topology L_F (Corollary 1). The expected utility, $\langle x, \bar{y}\rangle = U$, in the optimal system is given by $U = \Psi'(\beta)$, where $\beta^{-1} = (\Phi')^{-1}(I)$ (Theorem 2). Similarly, the information amount, $F(\bar{y}) = I$, in the optimal system is given is $I = \Phi'(\beta^{-1})$, where $\beta = (\Psi')^{-1}(U)$. Because $I = I(t)$ and $U = U(t)$ are monotonic, the integrals do not change if the trajectory is parametrized by $\beta \in [\beta_1, \beta_2]$. Thus, path integrals along the optimal trajectory are equal to Riemann integrals $\int_{\beta_1}^{\beta_2} d\Psi(\beta)$ and $\int_{\beta_2^{-1}}^{\beta_1^{-1}} d\Phi(\beta^{-1})$. The final expressions are obtained by applying the Newton–Leibniz formula. □

2.4 Empirical Evaluation on Learning Agents

The optimal learning trajectory is not an algorithm for optimal learning. It, however, describes the equivalence class of evolutions of learning systems that is optimal with respect to a utility function x and some measure of information F. Subdifferential $\partial F^*(x)$ of its dual defines the family of optimal distributions, which depends also on the prior corresponding to the minimum of information. The points on the optimal trajectory are then computed using the amount of empirical information $I \in \mathbb{R}$ or empirical expected utility $U \in \mathbb{R}$. Moreover, because I or U are local constraints, the optimality is not asymptotic. Thus, an algorithm for nonasymptotic optimal learning in the described above sense should be such that the evolution of the system were as close as possible to the optimal information trajectory.

Here, we evaluate this idea in an experiment using an architecture for comparing different action-selection strategies in agents, described in [3]. The architecture consists of an agent placed in a virtual environment, and the main goal of the agent is to find and collect as many rewards as possible. The rewards appear in the environment stochastically according to some probability law that is unknown to the agent. The probabilities of rewards depend on some predefined initial pattern of the environment and also on the previous actions of the agent (recall the cat and mice problem). Thus, the probability law defining the rewards is nonstationary.

The experiments, reported here, compare the performance of three agents in an environment with five states $\mathcal{Y} = \{y_1, \dots, y_5\}$ and rewards with a binary utility $x(y) \in \{0, 1\}$. The results are reported for rewards distributed according to two initial patterns $\{p, 0, p, 0, p\}$ and $\{p, 0, 0, 0, p\}$, where $p \in [0, 1]$ is the probability $P(x = 1 \mid y_i, x = 0)$ of a reward appearing at state $y_i \in \mathcal{Y}$ with no current reward. Thus, p defines the average *reward frequency* in a state. The agent has three actions $\mathcal{Z} = \{z_1, z_2, z_3\}$—moving left, right or do nothing.

The agent selected actions based on estimates $\tilde{x}(y, z)$ of receiving a reward by taking action $z \in \mathcal{Z}$ in state $y \in \mathcal{Y}$ (i.e., $z(y) = \arg\max_z \tilde{x}(y, z)$). These estimates were computed using empirical probability $P_e(x \mid y, z)$ based on joint empirical distribution $P_e(x, y, z)$ stored in the agent's memory. Using different methods to compute $\tilde{x}(y, z)$ may result in the agent selecting different actions in the same states leading to differences in performance and empirical distributions $P_e(x, y, z)$. The empirical distribution $\bar{P}_e(x, y, z)$ of an optimal system should evolve along the optimal learning trajectory.

Three agents were compared using the following estimation methods:

$$\tilde{x}(y, z) = E\{x \mid y, z\} \tag{2.5}$$

$$\tilde{x}(y, z) = E\{x \mid y, z\} + \xi, \quad \xi \in \mathcal{N}\big(0, \sigma^2\big), \quad \sigma^2 = \mathrm{Var}\{x \mid y, z\} \tag{2.6}$$

$$\tilde{x}(y, z) = \bar{F}^{-1}(\xi), \quad \xi \in \mathrm{Rand}(0, 1), \quad \bar{F}(x) = \int_{-\infty}^{x} d\bar{P}(t \mid y, z) \tag{2.7}$$

The first agent, referred to as 'max $E\{u\}$' (max expected utility), estimates the utilities by their empirical expectations. This strategy is known to be suboptimal in some problems, and is often referred to as a *greedy* strategy. Note that max $E\{u\}$ corresponds to optimization without information constraints. Indeed, the maximum of information gives $\beta^{-1} = 0$ in Theorem 1, and the Lagrange function reduces to the expected utility. Thus, the greedy strategy 'overestimates' the amount of empirical information.

The second agent, referred to as 'Noisy $E\{u\}$', uses stochastic strategy, where the conditional expectation is randomized by ξ, sampled from zero-mean normal distribution with empirical variance. Thus, this method does not use statistics of order higher than two. Generally, this corresponds to using less information than the empirical distribution contains.

The third agent, referred to as 'Rand $ML(u)$' (for 'random maximum likelihood'), uses stochastic estimates sampled from probability measure $\bar{P}(x \mid y, z)$ that is optimal with respect to empirical information constraints. Sampling is performed using the inverse distribution function method. Note that $\bar{P}$ can be also parametrized by the empirical expected utility $U \in \mathbb{R}$, and for binary utility function $x \in \{0, 1\}$ there is only one distribution such that $E\{x\} = U$. Thus, for binary utility $\bar{P} = P_e$, and $\tilde{x}(y, z)$ are sampled directly from $P_e(x \mid y, z)$.

The results are reported on Figs. 2.2, 2.3 and 2.4. Charts on the left are for pattern $\{p, 0, p, 0, p\}$ and on the right for $\{p, 0, 0, 0, p\}$. All the points on the charts are

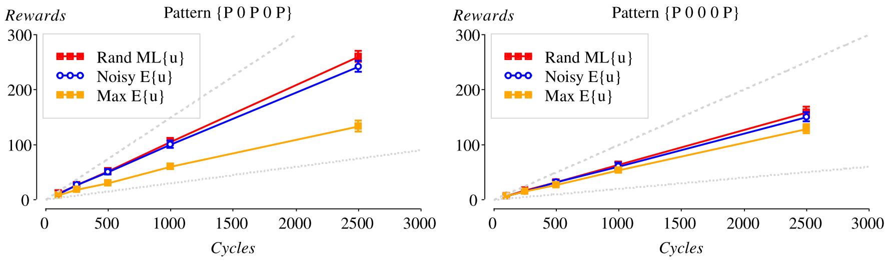


Fig. 2.2 Average numbers of rewards collected (ordinates) as a functions of cycles (abscissae) for three strategies

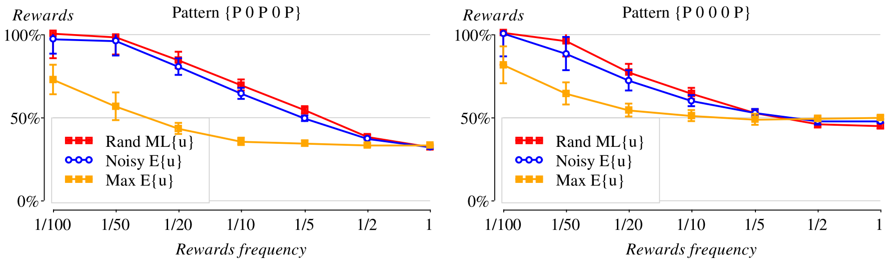


Fig. 2.3 Percentage of rewards collected as a function of rewards' frequency

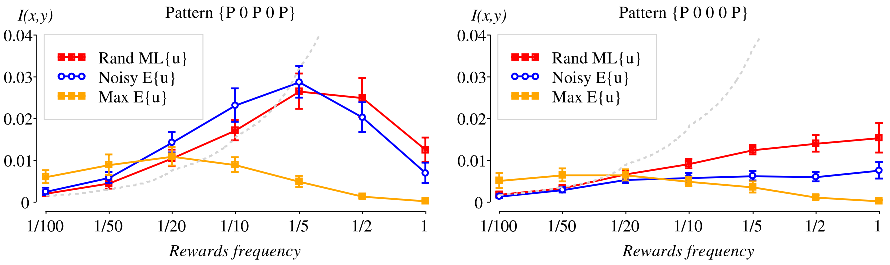


Fig. 2.4 Posterior information amount as a function of rewards' frequency

the average values from 30 experiments. The error bars on all charts are standard deviations.

Figure 2.2 shows the numbers of rewards against the number of cycles in the experiments with $p = .1$. One can see that the best performance was achieved by the Rand $ML(u)$ agent, the second is the Noisy $E\{u\}$ agent, and the least number of reward was collected by the max $E\{u\}$ agent, as expected.

Figure 2.3 shows the percentage of rewards collected by the agents after 1000 cycles in different experiments with the control probability of rewards $p \in [.01, 1]$, shown on the horizontal axis. Figure 2.4 shows, for the same experiments, the amount of Shannon information $I_{x,y}$ between rewards and states computed from the empirical distribution $P_e(x, y) = \sum_z P_e(x, y, z)$. One can see that the agent collecting the greatest number of rewards also often requires the least amounts of information (particularly for $p \in [.01, .05]$). These empirical results agree with the theory, presented in previous sections.

2.5 Conclusion

This paper presented geometric representation of evolution of learning systems. The representation is related to the use of Orlicz spaces in infinite-dimensional nonparametric information geometry, but the topology considered here is based on more general convex functions on linear spaces. The duality plays a very important role. In particular, subdifferentials of dual convex functionals are (generally multi-valued) monotone operators between the dual spaces, and they set up Galois connection preserving pre-orders on the topological spaces. Monotone transformations are very desirable in our theory, because when applied to utility functions, they also preserve the preference relation (complete pre-order) on the space of outcomes. Note that pre-order (order) is not symmetric (antisymmetric) binary relation, and preserving this property was our main motivation for considering asymmetric topologies on the statistical manifold.

The topology related to information allows for the definition of continuous trajectories representing the evolution of a learning system. Optimality conditions have been formulated using the information value theory, and generalized characteristic potentials have been defined to parametrize the optimal information trajectory by empirical constraints. Path integrals along the optimal trajectory define theoretical bounds for a learning system that can be computed as a difference of the potentials at the end points of the trajectory. This result has some similarity to the gradient theorem about path independence of the integral in a conservative vector field.

The theory was illustrated not only on several theoretical examples, but also evaluated in an experiment. The results suggest that the theory can be very useful in many applications of machine learning, such as nonasymptotic optimization of systems with dynamic information, optimization of communication networks based on information value and optimization of the ‘exploration-exploitation’ balance in statistical decisions. The latter problem has been often approached using stochastic methods based on Gibbs distributions with unknown parameter β^{-1} (temperature). Optimality conditions $\beta^{-1} \in \partial U(I)$ or $\beta \in \partial I(U)$ define the parameter from empirical constraints, and with it the optimal level of exploration. Previously, the author applied the relation between parameter β^{-1} and information to cognitive models of human and animals’ learning behavior [2], and it improved significantly the correspondence between the models and experimental data. Further development of the theory and its applications to machine learning problems is the subject of ongoing research.

Acknowledgement This work was supported in part by EPSRC grant EP/DO59720.

References

1. Amari, S.I.: Differential-geometrical methods of statistics. Lecture Notes in Statistics, vol. 25. Springer, Berlin (1985)
2. Belavkin, R.V.: On emotion, learning and uncertainty: A cognitive modelling approach. PhD thesis, The University of Nottingham, Nottingham, UK (2003)
3. Belavkin, R.V.: Acting irrationally to improve performance in stochastic worlds. In: Bramer, M., Coenen, F., Allen, T. (eds.) Proceedings of AI–2005, the 25th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence. Research and Development in Intelligent Systems vol. XXII, pp. 305–316. Springer, Cambridge (2005). BCS
4. Belavkin, R.V.: The duality of utility and information in optimally learning systems. In: 7th IEEE International Conference on 'Cybernetic Intelligent Systems'. IEEE Press, London (2008)
5. Belavkin, R.V.: Bounds of optimal learning. In: 2009 IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning, pp. 199–204. IEEE Press, Nashville (2009)
6. Bellman, R.E.: Dynamic Programming. Princeton University Press, Princeton (1957)
7. Chentsov, N.N.: Statistical Decision Rules and Optimal Inference. Nauka, Moscow (1972). In Russian, English translation: Am. Math. Soc., Providence (1982)
8. de Finetti, B.: La prévision: ses lois logiques, ses sources subjectives. Ann. Inst. Henri Poincaré **7**, 1–68 (1937). In French
9. Jaynes, E.T.: Information theory and statistical mechanics. Phys. Rev. **106**, 620–630 (1957)
10. Jaynes, E.T.: Information theory and statistical mechanics. Phys. Rev. **108**, 171–190 (1957)
11. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. J. Artif. Intell. Res. **4**, 237–285 (1996)
12. Kolmogorov, A.N.: The theory of information transmission. In: Meeting of the USSR Academy of Sciences on Scientific Problems of Production Automatisation, 1956, pp. 66–99. Akad. Nauk USSR, Moscow (1957). In Russian
13. Kullback, S.: Information Theory and Statistics. Wiley, New York (1959)
14. Pistone, G., Sempi, C.: An infinite-dimensional geometric structure on the space of all the probability measures equivalent to a given one. Ann. Stat. **23**(5), 1543–1561 (1995)
15. Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V., Mishchenko, E.F.: The Mathematical Theory of Optimal Processes. Wiley, New York (1962). Translated from Russian
16. Robbins, H.: An empirical Bayes approach to statistics. In: Third Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 157–163 (1956)
17. Rockafellar, R.T.: Conjugate Duality and Optimization. CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 16. SIAM, Philadelphia (1974)
18. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Techn. J. **27**, 379–423 (1948)
19. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J. **27**, 623–656 (1948)
20. Showalter, R.E.: Monotone Operators in Banach Space and Nonlinear Partial Differential Equations. Mathematical Surveys and Monographs, vol. 49. Am. Math. Soc., Providence (1997)
21. Stratonovich, R.L.: Optimum nonlinear systems which bring about a separation of a signal with constant parameters from noise. Radiofizika **2**(6), 892–901 (1959)
22. Stratonovich, R.L.: Conditional Markov processes. Theory Probab. Appl. **5**(2), 156–178 (1960)
23. Stratonovich, R.L.: On value of information. Izv. USSR Acad. Sci. Techn. Cybern. **5**, 3–12 (1965). In Russian
24. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. Adaptive Computation and Machine Learning. MIT Press, Cambridge (1998)
25. von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behavior, 1st edn. Princeton University Press, Princeton (1944)
26. Wald, A.: Statistical Decision Functions. Wiley, New York (1950)

Chapter 3
Performance-Information Analysis and Distributed Feedback Stabilization in Large-Scale Interconnected Systems

Khanh D. Pham

Summary Large-scale interconnected systems are characterized as large and complex systems divided into several smaller autonomous systems that have certain autonomy in local optimization and decision-making. As an example, a class of interconnected linear stochastic systems, where no constituent systems need to have global information and distributed decision making enables autonomous systems to dynamically reconfigure risk-value aware performance indices for uncertain environmental conditions, is considered in the subject research. Among the many challenges in distributed and intelligent control of interconnected autonomous systems is performance uncertainty analysis and decentralized feedback stabilization. The theme of the proposed research is the interplay between performance-information dynamics and decentralized feedback stabilization, both providing the foundations for distributed and autonomous decision making. First, recent work by the author in which performance information availability was used to assess limits of achievable performance will be extended to give insight into how different aggregation structures and probabilistic knowledge of random decision processes between networks of autonomous systems are exploited to derive a distributed computation of complete distributions of performance for interconnected autonomous systems. Second, the resulting information statistics on performance of interconnected autonomous systems will be leveraged in the design of decentralized output-feedback stabilization, thus enabling distributed autonomous systems to operate resiliently in uncertain environments with performance guarantees that are now more robust than the traditional performance average.

3.1 Introduction

The research under investigation is adaptive control decisions of interconnected systems in stochastic and dynamic environments. The central subject matter of the

K.D. Pham (✉)
Space Vehicles Directorate, Air Force Research Laboratory, Kirtland Air Force Base, NM 87117, USA
e-mail: Khanh.Pham@kirtland.af.mil

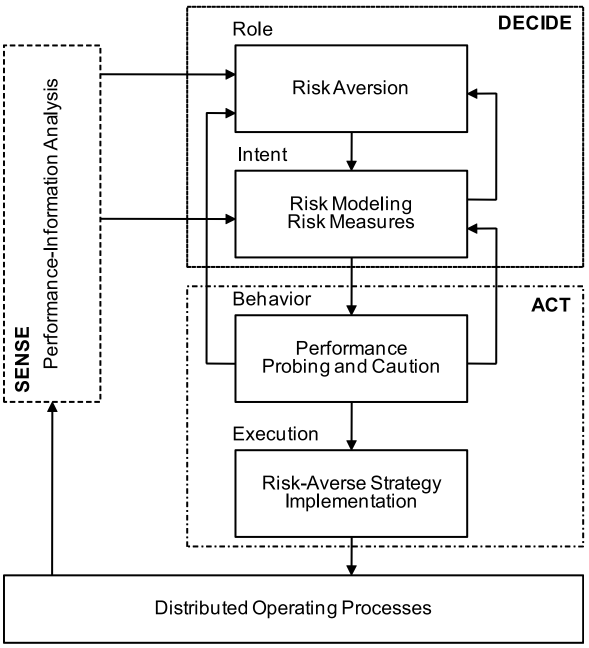


Fig. 3.1 An architecture for risk-averse based large-scale interconnected systems

chapter is focused on two related perspectives: the performance-information analysis perspective and the distributed feedback stabilization perspective. These seem to be the two dominant perspectives in stochastic control of large-scale interconnected systems. Specifically, the following two problems are associated with adaptive control decision. The first is the problem of designing a performance-information system that can motivate and screen appropriate adaptive control decision by an interconnected system in charge of local operation in a stochastic and dynamic environment. The second is the problem of emphasizing the importance of adaptive control decision in stochastic control under environmental uncertainty and exploring the characteristics of an interconnected system conducive to adaptive control decision by the autonomous system to whom operating authority is delegated.

What framework can be used to view that process in distributed stabilization for large-scale interconnected systems? To be useful, perhaps, such a framework must be simple, but at the same time it must be comprehensive enough that most of the essential elements of a distributed control process can be discussed. The control process envisaged here is a dynamic process surrounding performance assessment, feedback and corrective action. Diagrammatically, the entire relationship may be depicted as shown in Fig. 3.1. Each autonomous system has some goal in terms of the performance riskiness of the operating process it has to control. The performance uncertainty is affected not only by the control decision or action but also by the environment in which the system (the local control decision and the local operating process) must operate. The basic reason for the existence of control decision is uncertainty or incomplete knowledge on the part of the decision policy about (1) the mechanism of the operating process and (2) the environmental conditions that should prevail at any point in time. In this diagram, each interconnected system recognizes the need for action in congruence with the expected standard of performance, e.g., risk aversion for performance against all random realizations from environmental disturbances is considered herein. The autonomous system will use the standard as a guideline to determine whether its action is called for or not. Its recognition comes as a result of filtering feedback information through some criteria such as risk modeling and risk measures for making judgment.

From this point of view, all respective measures of performance for interconnected systems are viewed as random variables with mixed random realizations from their own uncertain environments. Information about the states of environments is assumed to be common knowledge. Furthermore, it is assumed here that each interconnected system will choose control decisions which will be best for its own utility, whether it is an internalized goal of either (1) performance probing via an effective knowledge construct that can be able to extract the knowledge of higher-order characteristics of performance distribution; (2) performance caution that mitigates performance riskiness with multiple attributes beyond performance averaging as such variance, skewness, flatness, etc., just to name a few; or (3) both. The final process execution has only one category of factors. It is the implementation of a risk-averse decision strategy where explicit recognition of performance uncertainty brings to what constitutes a "good" decision by the interconnected system. To the best of the author's knowledge, the problem of performance risk congruence has not attracted much academic or practical attention in stochastic multi-agent systems until very recently [8, 9] and [10]. Failure to recognize the problem may not be harmful in many operating decision situations when alternative courses of action may not differ very much in their risk characteristics. But in cases of important and indispensable consideration in reliability-based design [3] and incorporation of aversion to specification uncertainty [5], which may involve much uncertainty and alternatives whose risk characteristics are wide ranged, consideration of the interconnected system's goal toward risk may be a crucial factor in designing a high performance system.

The chapter is organized as follows. Section 3.2 puts the distributed control of adaptive control decisions for interconnected systems in uncertain environments into a consistent mathematical framework. In Sect. 3.3, the performance-information analysis is a focal point for adaptive control decisions of interconnected systems. The methodological characteristics of moment and cumulant generating models are used to characterize the uncertainty of performance information with respect to uncertain environments. Some interesting insights into how the value of performance information are affected by changes in the attributes of a performance-information system. Section 3.4 will develop a distributed feedback stabilization for a large-scale interconnected system with adaptive control decision using the recent statistical control development. Another main feature of this section is the exploration of risk and preference for a stochastic control system. A generalization of performance evaluation is suggested. The risk of a performance measure is expressible as a linear combination of the associated higher-order statistics. From Sect. 3.5, construction of a candidate function for the value function and the calculation of decentralized efficient risk-averse decision strategies accounting for multiple attributes of performance robustness are discussed. Finally, some concluding remarks are drawn in Sect. 3.6. These final remarks will help put the novelty of this research in perspectives of distributed control and performance-information analysis.

3.2 Problem Formulation

Before going into a formal presentation, it is necessary to consider some conceptual notations. To be specific, for a given Hilbert space X with norm $\|\cdot\|_X$, $1 \le p \le \infty$ and $a, b \in \mathbb{R}$ such that $a \le b$, a Banach space is defined as follows

$$\begin{aligned} &L^p_{\mathcal{F}}(a,b;X) \\ &\quad \triangleq \Bigg\{ \phi(\cdot) = \big\{\phi(t,\omega) : a \le t \le b\big\} \text{ such that } \phi(\cdot) \text{ is an } X\text{-valued} \\ &\qquad \mathcal{F}_t\text{-measurable process on } [a,b] \text{ with } E\left\{ \int_a^b \|\phi(t,\omega)\|^p_X \, dt \right\} < \infty \Bigg\} \end{aligned} \tag{3.1}$$

with norm

$$\|\phi(\cdot)\|_{\mathcal{F},p} \triangleq \left(E\left\{ \int_a^b \|\phi(t,\omega)\|^p_X \, dt \right\} \right)^{1/p} \tag{3.2}$$

where the elements ω of the filtered sigma field $\mathcal{F}_t$ of a sample description space Ω that is adapted for the time horizon $[a,b]$ are random outcomes or events. Also, the Banach space of X-valued continuous functionals on $[a,b]$ with the max-norm induced by $\|\cdot\|_X$ is denoted by $C(a,b;X)$. The deterministic version of (3.1) and its associated norm (3.2) is written as $L^p(a,b;X)$ and $\|\cdot\|_p$.

More specifically, consider a network of N interconnected systems, or equivalently, agents numbered from 1 through N. Each agent i, for $i \in \overline{N} \triangleq \{1,2,\ldots,N\}$ operates within its local environment modeled by the corresponding filtered probability space $(\Omega_i, \mathcal{F}_i, \{\mathcal{F}_i\}_{t \ge t_0 > 0}, \mathcal{P}_i)$ that is defined with a stationary p_i-dimensional Wiener process $w_i(t) \triangleq w_i(t,\omega_i) : [t_0,t_f] \times \Omega_i \mapsto \mathbb{R}^{p_i}$ on a finite horizon $[t_0,t_f]$ and the correlation of independent increments

$$E\big\{\big[w_i(\tau) - w_i(\xi)\big]\big[w_i(\tau) - w_i(\xi)\big]^T\big\} = W_i|\tau - \xi|, \quad W_i > 0$$

Assume that agent i generates and maintains its own states based on information concerning its neighboring agents and local environment. The update rule of agent i evolves according to a simple nearest-neighbor model

$$\begin{aligned} dx_i(t) &= \big(A_i(t)x_i(t) + B_i(t)u_i(t) + C_i(t)z_i(t) + E_i(t)d_i(t)\big)\,dt \\ &\quad + G_i(t)\,dw_i(t) \end{aligned} \tag{3.3}$$

$$x_i(t_0) = x_i^0$$

where all the coefficients $A_i \in C(t_0, t_f; \mathbb{R}^{n_i \times n_i})$, $B_i \in C(t_0, t_f; \mathbb{R}^{n_i \times m_i})$, $C_i \in C(t_0, t_f; \mathbb{R}^{n_i \times q_i})$, $E_i \in C(t_0, t_f; \mathbb{R}^{n_i \times r_i})$, and $G_i \in C(t_0, t_f; \mathbb{R}^{n_i \times p_i})$ are deterministic matrix-valued functions. Furthermore, it should be noted that $x_i \in L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{n_i})$ is the n_i-dimensional state of agent i with the initial state $x_i^0 \in \mathbb{R}^{n_i}$ fixed, $u_i \in L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{m_i})$ is the m_i-dimensional control decision, and $d_i \in L^2(t_0, t_f; \mathbb{R}^{r_i})$ is the r_i-dimensional known disturbance.

In the formulation the actual interaction, $z_i \in L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{q_i})$ represented by the q_i-dimensional process is of interest to agent i where L_{ij} represent time-varying interaction gains associated with its neighboring agents

$$z_i(t) = \sum_{j=1, j \neq i}^{N} L_{ij}(t) x_{ij}(t), \quad i \in \overline{N} \tag{3.4}$$

For practical purposes, the process z_i is however not available at the level of agent i. It is then desired to have decentralized decision making without intensive communication exchanges. An approach of interaction prediction is therefore proposed to resolve the communication difficulty by means of a crude model of reduced order for the interactions among the neighboring agents that can exchange information with agent i. In this work, the actual interaction $z_i(\cdot)$ is now approximated by an explicit model of the type

$$dz_{mi}(t) = \big(A_{zi}(t) z_{mi}(t) + E_{zi}(t) d_{mi}(t)\big)\, dt + G_{zi}(t)\, dw_{mi}(t), \quad z_{mi}(t_0) = 0 \tag{3.5}$$

where $A_{zi} \in C(t_0, t_f; \mathbb{R}^{q_i \times q_i})$ is an arbitrary deterministic matrix-valued function which describes a crude model for the actual interaction $z_i(\cdot)$. In particular, A_{zi} can be chosen to be the off-diagonal block of the global matrix coefficient A corresponding the partition vector $z_i(\cdot)$. Other coefficients $E_{zi} \in C(t_0, t_f; \mathbb{R}^{q_i \times r_{mi}})$ and $G_{zi} \in C(t_0, t_f; \mathbb{R}^{q_i \times p_{mi}})$ are deterministic matrix-valued functions of the stochastic differential equation (3.5). The approximate interaction $z_{mi}(\cdot)$ produced by the model is affected not only by the known disturbance $d_{mi} \in L^2(t_0, t_f; \mathbb{R}^{r_{mi}})$ but also by another local uncertainty $w_{mi}(t) \triangleq w_{mi}(t, \omega_{mi}) : [t_0, t_f] \times \Omega_{mi} \mapsto \mathbb{R}^{p_{mi}}$ which is an p_{mi}-dimensional stationary Wiener process defined on a complete filtered probability space $(\Omega_{mi}, \mathcal{F}_{mi}, \{\mathcal{F}_{mi}\}_{t \geq t_0 > 0}, \mathcal{P}_{mi})$ over $[t_0, t_f]$ with the correlation of independent increments

$$E\left\{\big[w_{mi}(\tau) - w_{mi}(\xi)\big]\big[w_{mi}(\tau) - w_{mi}(\xi)\big]^T\right\} = W_{mi}|\tau - \xi|, \quad W_{mi} > 0$$

With the approach considered here, there is a need to treat the actual interaction $z_i(\cdot)$ as a control process that is supposed to follow the approximate interaction process $z_{mi}(\cdot)$. Thus, this requirement leads to certain classes of admissible control laws associated with (3.3) to be denoted by $U_i \times Z_i \subset L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{m_i}) \times L^2_{\mathcal{F}_{mi}}(t_0, t_f; \mathbb{R}^{q_i})$, e.g., given $(u_i(\cdot), z_i(\cdot)) \in U_i \times Z_i$, the 3-tuple $(x_i(\cdot), u_i(\cdot), z_i(\cdot))$ shall be referred to as an admissible 3-tuple if $x_i(\cdot) \in L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{n_i})$ is a solution of the stochastic differential equation (3.3) associated with $u_i(\cdot) \in U_i$ and $z_i(\cdot) \in Z_i$. Furthermore, the nearest neighbor model of local dynamics (3.3) in the absence of its known disturbances and local environment is assumed to be uniformly exponentially stable. That is, there exist positive constants η_1 and η_2 such that the pointwise matrix norm of the closed-loop state transition matrix associated with the local dynamical model (3.3) satisfies the inequality

$$\left\|\Phi_i(t,\tau)\right\| \le \eta_1 e^{-\eta_2(t-\tau)} \quad \forall t \ge \tau \ge t_0$$

The pair $(A_i(t), [B_i(t), C_i(t)])$ is pointwise stabilizable if there exist bounded matrix-valued functions $K_i(t)$ and $K_{zi}(t)$ so that the closed-loop system $dx(t) = (A_i(t) + B_i(t)K_i(t) + C_i(t)K_{zi}(t))x_i(t)\,dt$ is uniformly exponentially stable.

Under the assumptions of $x_i(t_0) \in \mathbb{R}^{n_i}$, $z_{mi}(t_0) \in \mathbb{R}^{q_i}$, $u_i(\cdot) \in U_i$, and $z_i(\cdot) \in Z_i$, there are tradeoffs among the closeness of the local states from desired states, the size of the local control levels, and the size of local interaction approximates. Therefore, agent i has to carefully balance the three to achieve its local performance. Stated mathematically, there exists an integral-quadratic form (IQF) performance measure $J_i : \mathbb{R}^{n_i} \times \mathbb{R}^{q_i} \times U_i \times Z_i \mapsto \mathbb{R}$

$$\begin{aligned} J_i&\big(x_i^0, z_{mi}^0; u_i(\cdot), z_i(\cdot)\big) \\ &= x_i^T(t_f)Q_i^f x_i(t_f) + \int_{t_0}^{t_f} \big[x_i^T(\tau)Q_i(\tau)x_i(\tau) + u_i^T(\tau)R_i(\tau)u_i(\tau) \\ &\quad + \big(z_i(\tau) - z_{mi}(\tau)\big)^T R_{zi}(\tau)\big(z_i(\tau) - z_{mi}(\tau)\big)\big]\,d\tau \end{aligned} \tag{3.6}$$

associated with agent i wherein $Q_i^f \in \mathbb{R}^{n_i \times n_i}$, $Q_i \in C(t_0, t_f; \mathbb{R}^{n_i \times n_i})$, $R_i \in C(t_0, t_f; \mathbb{R}^{m_i \times m_i})$, and $R_{zi} \in C(t_0, t_f; \mathbb{R}^{q_i \times q_i})$ representing relative weightings for terminal states, transient states, decision levels, and interaction mismatches are deterministic and positive semidefinite with $R_i(t)$ and $R_{zi}(t)$ invertible.

The description of the framework continues with yet another collection of random processes $\{y_i(t)\}_{i=1}^N$ that carry critical information about, for instance, the agents' states and their underlying dynamic structures, e.g.,

$$dy_i(t) = H_i(t)x_i(t)\,dt + dv_i(t), \quad i \in \overline{N} \tag{3.7}$$

which is a causal function of the local process $x_i(t)$ corrupted by another stationary s_i-dimensional Wiener process $v_i(t) \triangleq v_i(t, \omega_i) : [t_0, t_f] \times \Omega_i \mapsto \mathbb{R}^{s_i}$ on a finite horizon $[t_0, t_f]$ and the correlation of independent increments

$$E\big\{\big[v_i(\tau) - v_i(\xi)\big]\big[v_i(\tau) - v_i(\xi)\big]^T\big\} = V_i|\tau - \xi|, \quad V_i > 0$$

Within the chosen framework for decentralized decision making, each agent is further supposed to estimate its evolving process that depends on both its own local measurements and interaction approximates from other agents. Given observations $y_i(\tau)$, $t_0 \le \tau \le t$, the state estimate at agent i is denoted by $\hat{x}_i(t)$. Further, the state estimate error covariance matrix of the state estimation error $\tilde{x}_i(t) \triangleq x_i(t) - \hat{x}_i(t)$

is described by $\Sigma_i(t) \triangleq E\{[x_i(t) - \hat{x}_i(t)][x_i(t) - \hat{x}_i(t)]^T\}$. With the additional assumption of (A_i, H_i) uniformly detectable, it can then be shown that the estimate $\hat{x}_i(t)$ for each agent i is readily generated by the solution of the stochastic differential equation (Kalman-type filter)

$$\begin{aligned} d\hat{x}_i(t) &= \big(A_i(t)x_i(t) + B_i(t)u_i(t) + C_i(t)z_i(t) + E_i(t)d_i(t)\big)\,dt \\ &\quad + L_i(t)\big(dy_i(t) - H_i(t)\hat{x}_i(t)\,dt\big), \quad \hat{x}_i(t_0) = x_i^0 \end{aligned} \tag{3.8}$$

where $\tilde{x}_i(t)$ is the state estimation error obtained from the error process

$$\begin{aligned} d\tilde{x}_i(t) &= \big(A_i(t) - L_i(t)H_i(t)\big)\tilde{x}_i(t)\,dt + G_i(t)\,dw_i(t) - L_i(t)\,dv_i(t) \\ \tilde{x}_i(t_0) &= 0 \end{aligned} \tag{3.9}$$

and for each agent i it is remarked that the Kalman gain is given by

$$L_i(t) = \Sigma_i(t)H_i^T(t)V_i^{-1} \tag{3.10}$$

The error covariance matrix $\Sigma_i(t)$ is the solution of the matrix Riccati differential equation

$$\begin{aligned} \frac{d}{dt}\Sigma_i(t) &= A_i(t)\Sigma_i(t) + \Sigma_i(t)A_i(t) + G_i(t)W_iG_i^T(t) \\ &\quad - \Sigma_i(t)H_i^T(t)V_i^{-1}H_i(t)\Sigma_i(t), \quad \Sigma_i(t_0) = 0 \end{aligned} \tag{3.11}$$

The aggregate autonomy-interaction arrangement that locally depicts the aspirations and interaction details of each agent i then requires the following augmented subsystem variables and parameters

$$x_{ai}(t) \triangleq \begin{bmatrix} \hat{x}_i(t) \\ \tilde{x}_i(t) \\ z_{mi}(t) \end{bmatrix}; \qquad x_{ai}^0 \triangleq \begin{bmatrix} x_i^0 \\ 0 \\ 0 \end{bmatrix}; \qquad w_{ai}(t) \triangleq \begin{bmatrix} w_i(t) \\ v_i(t) \\ w_{mi}(t) \end{bmatrix} \tag{3.12}$$

The respective tradeoff interaction levels and local incentive (3.6) for agent i is rewritten as follows

$$\begin{aligned} &J_i\big(x_{ai}^0; u(\cdot), z_i(\cdot)\big) \\ &= x_{ai}^T(t_f)Q_{ai}^f x_{ai}(t_f) + \int_{t_0}^{t_f} \big[x_{ai}^T(\tau)Q_{ai}(\tau)x_{ai}(\tau) + u_i^T(\tau)R_i(\tau)u_i(\tau) \\ &\quad + z_i^T(\tau)R_{zi}(\tau)z_i(\tau) - 2x_{ai}^T(\tau)S_{ai}(\tau)z_i(\tau)\big]\,d\tau \end{aligned} \tag{3.13}$$

where some of the local weightings are given by

$$Q_{ai}^f \triangleq \begin{bmatrix} Q_i^f & Q_i^f & 0 \\ Q_i^f & Q_i^f & 0 \\ 0 & 0 & 0 \end{bmatrix}; \qquad Q_{ai} \triangleq \begin{bmatrix} Q_i & Q_i & 0 \\ Q_i & Q_i & 0 \\ 0 & 0 & R_{zi} \end{bmatrix}; \qquad S_{ai} \triangleq \begin{bmatrix} 0 \\ 0 \\ R_{zi} \end{bmatrix} \tag{3.14}$$

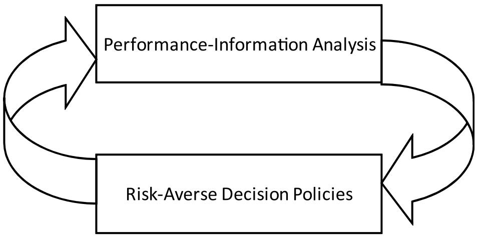

Performance-Information Analysis
Risk-Averse Decision Policies

Fig. 3.2 Risk-averse control process: performance-information analysis and risk-averse decision policies

subject to the local dynamics for strategic relations

$$\begin{aligned} dx_{ai}(t) &= \big(A_{ai}(t)x_{ai}(t) + B_{ai}(t)u_i(t) + C_{ai}(t)z_i(t) + D_{ai}(t)\big)\,dt \\ &\quad + G_{ai}(t)dw_{ai}(t) \qquad (3.15) \\ x_{ai}(t_0) &= x_{ai}^0 \end{aligned}$$

with the corresponding system parameters

$$A_{ai} \triangleq \begin{bmatrix} A_i & L_iH_i & 0 \\ 0 & A_i - L_iH_i & 0 \\ 0 & 0 & A_{zi} \end{bmatrix}; \quad B_{ai} \triangleq \begin{bmatrix} B_i \\ 0 \\ 0 \end{bmatrix}; \quad C_{ai} \triangleq \begin{bmatrix} C_i \\ 0 \\ 0 \end{bmatrix} \qquad (3.16)$$

$$D_{ai} \triangleq \begin{bmatrix} E_id_i \\ 0 \\ E_{zi}d_{mi} \end{bmatrix}; \quad G_{ai} \triangleq \begin{bmatrix} 0 & L_i & 0 \\ G_i & -L_i & 0 \\ 0 & 0 & G_{zi} \end{bmatrix}; \quad W_{ai} \triangleq \begin{bmatrix} W_i & 0 & 0 \\ 0 & V_i & 0 \\ 0 & 0 & W_{mi} \end{bmatrix} \qquad (3.17)$$

Thus, far only the structure and the circumstantial setting of a large-scale interconnected system have been made clear. The next task is a discussion of a risk-averse control process which can be divided into two phases: (1) performance-information analysis and (2) risk-averse decision policy. The essence of these phases is, in a sense, the setting of a standard. Standards in performance-information analysis provide a yardstick of performance evaluation and standards in risk-averse decision policy phase communicate and coordinate acceptable levels of risk aversion efforts to the interconnected systems. Essentially, Fig. 3.2 shows a model of the standards in a risk-averse control system serving two purposes: (1) as targets for the interconnected systems to strive for and (2) as an evaluation yardstick.

3.3 Performance-Information Analysis

Performance-information analysis affects the consequence factor to agent i at two levels. First, it specifies the variable of performance measure to be looked at and thus determines an important element in the consequence of control decision at agent i. Second, performance risk analysis implies the existence of standards of comparison for the performance variables specified. This implies that some notions of risk modeling and measures for which standards have great significance may be in need.

As a basic analytical construct, it is important to recognize that, within the view of the linear-quadratic structure (3.15) and (3.13), the performance measure (3.13) for agent i is clearly a random variable with Chi-squared type. Hence, the degree of uncertainty of (3.13) must be assessed via a complete set of higher-order performance statistics beyond the statistical averaging. The essence of information about the states of performance uncertainty regards as a source of information flow which will affect agent i's perception of the problem and the environment. More specifically, for agent i, first and second characteristic functions of the Chi-squared random variable (3.13), denoted respectively, by $\varphi_i(s, x_{ai}^s)$ and $\psi_i(s, x_{ai}^s)$ which are utilized to provide a conceptual framework for evaluation of performance information, are given by

$$\varphi_i\big(s, x_{ai}^s; \theta_i\big) \triangleq E\big\{e^{\theta_i J_i(s, x_{ai}^s)}\big\} \tag{3.18}$$

$$\psi_i\big(s, x_{ai}^s; \theta_i\big) \triangleq \ln\big\{\varphi_i\big(s, x_{ai}^s; \theta_i\big)\big\} \tag{3.19}$$

where small parameter θ_i is in an open interval about 0 for the information systems $\varphi_i(s, x_{ai}^s)$ and $\psi_i(s, x_{ai}^s)$ whereas the initial condition, (t_0, x_{ai}^0) is now parameterized as the running condition, (s, x_{ai}^s) with $s \in [t_0, t_f]$ and $x_{ai}^s \triangleq x_{ai}(s)$. From the definition, there are three factors as the determinants of the value of performance information: (i) the characteristics of the set of actions; (ii) the characteristics of consequence of alternatives; and (iii) the characteristics of the utility function.

The first factor relates to the action set, denoted by U_i via state-estimate feedback strategies to maintain a fair degree of accuracy on each agent behaviors and the corresponding utility. Since the significance of (3.15) and (3.13) is the linear-quadratic nature, the actions of agent i should therefore be causal functions of the local process $\hat{x}_i(t)$. The restriction of decision strategy spaces can be justified by the assumption that agent i participates in the large-scale decision making where it only has access to the current time and local state estimates of the interaction. Thus, it amounts to considering only those feedback strategies which permit linear feedback syntheses $\hat{\gamma}_i : [t_0, t_f] \times L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{n_i}) \mapsto L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{m_i})$ and $\hat{\gamma}_i^z : [t_0, t_f] \times L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{n_i}) \mapsto L^2_{\mathcal{F}_i}(t_0, t_f; \mathbb{R}^{q_i})$

$$u_i(t) = \hat{\gamma}_i\big(t, \hat{x}_i(t)\big) \triangleq K_i(t)\hat{x}_i(t) + p_i(t) \tag{3.20}$$

$$z_i(t) = \hat{\gamma}_i^z\big(t, \hat{x}_i(t)\big) \triangleq K_{zi}(t)\hat{x}_i(t) + p_{zi}(t) \tag{3.21}$$

where matrix-valued gains $K_i \in C(t_0, t_f; \mathbb{R}^{m_i \times n_i})$ and $K_{zi} \in C(t_0, t_f; \mathbb{R}^{q_i \times n_i})$ together with vector-valued affine $p_i \in C(t_0, t_f; \mathbb{R}^{m_i})$ and $p_{zi} \in C(t_0, t_f; \mathbb{R}^{q_i})$ will be appropriately defined, respectively.

The importance of the second factor comes from the consequence of actions, denoted by the outcome function which maps the triplets of outcomes, actions and random realizations of the underlying stochastic process into outcomes, e.g.,

$$\begin{aligned} dx_{ai}(t) &= \big(F_{ai}(t)x_{ai}(t) + B_{ai}(t)p_i(t) + C_{ai}(t)p_{zi}(t) + D_{ai}(t)\big)\,dt \\ &\quad + G_{ai}(t)\,dw_{ai}(t) \qquad (3.22) \\ x_{ai}(s) &= x_{ai}^s \end{aligned}$$

where the time-continuous composite state matrix, F_{ai} is given by

$$F_{ai} \triangleq \begin{bmatrix} A_i + B_iK_i + C_iK_{zi} & L_iH_i & 0 \\ 0 & A_i - L_iH_i & 0 \\ 0 & 0 & A_{zi} \end{bmatrix}$$

The characteristics of the utility function $J_i(s, x_{ai}^s)$ which maps outcomes into utility levels can be influenced the value of performance information in various ways, e.g.,

$$\begin{aligned} J_i(s, x_{ai}^s) &= x_{ai}^T(t_f)Q_{ai}^f x_{ai}(t_f) + \int_s^{t_f} \big[x_{ai}^T(\tau)N_{ai}(\tau)x_{ai}(\tau) + 2x_{ai}^T(\tau)O_{ai}(\tau)\big]\,d\tau \\ &\quad + \int_s^{t_f} \big[p_i^T(\tau)R_i(\tau)p_i(\tau) + p_{zi}^T(\tau)R_{zi}(\tau)p_{zi}(\tau)\big]\,d\tau \qquad (3.23) \end{aligned}$$

where

$$N_{ai} \triangleq \begin{bmatrix} Q_i + K_i^TR_iK_i + K_{zi}^TR_{zi}K_{zi} & Q_i & 0 \\ Q_i & Q_i & 0 \\ -2R_{zi}K_{zi} & 0 & R_{zi} \end{bmatrix}$$

$$O_{ai} \triangleq \begin{bmatrix} K_i^TR_ip_i + K_{zi}^TR_{zi}p_{zi} \\ 0 \\ -R_{zi}p_{zi} \end{bmatrix}$$

The next result investigates the perception of performance riskiness from the standpoint of higher-order characteristics pertaining to probability distributions with respect to all random realizations from the underlying stochastic environment. It provides a quantitative explication of the concept of performance risk which is expressible as a linear combination of its higher-order statistics, i.e., performance-measure statistics whose mathematical descriptions are given below.

Theorem 1 (Performance-measure statistics) *For each agent i, let the pairs (A_i, B_i) and (A_i, C_i) be uniformly stabilizable and the pair (A_i, H_i) be uniformly detectable on $[t_0, t_f]$ in the autonomous system governed by* (3.22) *and* (3.23). *Then, for any given $k_i \in \mathbb{Z}^+$, the k_i-th cumulant associated with* (3.23) *is given by*

$$\kappa_{k_i}^i = \big(x_{ai}^0\big)^T H^i(t_0, k_i)x_{ai}^0 + 2\big(x_{ai}^0\big)^T \breve{D}^i(t_0, k_i) + D^i(t_0, k_i) \qquad (3.24)$$

where the cumulant variables $\{H^i(s, r)\}_{r=1}^{k_i}$, $\{\breve{D}^i(s, r)\}_{r=1}^{k_i}$ and $\{D^i(s, r)\}_{r=1}^{k_i}$ evaluated at $s = t_0$ satisfy the supporting matrix-valued differential equations (*with the* *dependence of* $H^i(s,r)$, $\breve{D}^i(s,r)$, *and* $D^i(s,r)$ *upon the admissible* K_i, K_{zi}, p_i, *and* p_{zi} *suppressed*)

$$\frac{d}{ds}H^i(s,1) = -F_{ai}^T(s)H^i(s,1) - H^i(s,1)F_{ai}(s) - N_{ai}(s) \tag{3.25}$$

$$\frac{d}{ds}H^i(s,r) = -F_{ai}^T(s)H^i(s,r) - H^i(s,r)F_{ai}(s) - \sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}H^i(s,v)G_{ai}(s)W_{ai}G_{ai}^T(s)H^i(s,r-v), \quad 2 \le r \le k_i \tag{3.26}$$

$$\frac{d}{ds}\breve{D}^i(s,1) = -F_{ai}^T(s)\breve{D}^i(s,1) - H^i(s,1)\big[B_{ai}(s)p_i(s) + C_{ai}(s)p_{zi}(s) + D_{ai}(s)\big] - O_{ai}(s) \tag{3.27}$$

$$\frac{d}{ds}\breve{D}^i(s,r) = -F_{ai}^T(s)\breve{D}^i(s,r) - H^i(s,r)\big[B_{ai}(s)p_i(s) + C_{ai}(s)p_{zi}(s) + D_{ai}(s)\big], \quad 2 \le r \le k_i \tag{3.28}$$

$$\frac{d}{ds}D^i(s,1) = -2(\breve{D}^i)^T(s,1)\big[B_{ai}(s)p_i(s) + C_{ai}(s)p_{zi}(s) + D_{ai}(s)\big] - Tr\big\{H^i(s,1)G_{ai}(s)W_{ai}G_{ai}^T(s)\big\} - p_i^T(s)R_i(s)p_i(s) - p_{zi}^T(s)R_{zi}(s)p_{zi}(s) \tag{3.29}$$

$$\frac{d}{ds}D^i(s,r) = -2(\breve{D}^i)^T(s,r)\big[B_{ai}(s)p_i(s) + C_{ai}(s)p_{zi}(s) + D_{ai}(s)\big] - Tr\big\{H^i(s,r)G_{ai}(s)W_{ai}G_{ai}^T(s)\big\}, \quad 2 \le r \le k_i \tag{3.30}$$

where the terminal-value conditions $H^i(t_f,1) = Q_{ai}^f$, $H^i(t_f,r) = 0$ *for* $2 \le r \le k_i$; $\breve{D}^i(t_f,r) = 0$; *and* $D^i(t_f,r) = 0$ *for* $1 \le r \le k_i$.

To anticipate for a well-posed optimization problem that follows, some sufficient conditions for the existence of solutions to the cumulant-generating equations in the calculation of performance-measure statistics are now presented in the sequel.

Theorem 2 (Existence of solutions for performance-measure statistics) *Let* (A_i, B_i) *and* (A_i, C_i) *be uniformly stabilizable. Also let* (A_i, H_i) *be uniformly detectable. Then, any given* $k_i \in \mathbb{Z}^+$, *the time-backward matrix differential equations* (3.25)–(3.30) *admit unique and bounded solutions* $\{H^i(s,r)\}_{r=1}^{k_i}$, $\{\breve{D}^i(s,r)\}_{r=1}^{k_i}$ *and* $\{D^i(s,r)\}_{r=1}^{k_i}$ *on* $[t_0, t_f]$.

Proof Under the assumptions of stabilizability and detectability, there always exist some feedback control and filter gains, $K_i(s)$, $K_{zi}(s)$ and $L_i(s)$ such that the continuous-time composite state matrix $F_{ai}(s)$ is exponentially stable on $[t_0, t_f]$. According to the results in [1], the state transition matrix, $\Phi_{ai}(s, t_0)$ associated with the continuous-time composite state matrix $F_{ai}(s)$ has the following properties

$$\frac{d}{ds}\Phi_{ai}(s,t_0) = F_{ai}(s)\Phi_{ai}(s,t_0); \quad \Phi_{ai}(t_0,t_0) = I$$

$$\lim_{t_f\to\infty}\left\|\Phi_{ai}(t_f,\sigma)\right\| = 0; \qquad \lim_{t_f\to\infty}\int_{t_0}^{t_f}\left\|\Phi_{ai}(t_f,\sigma)\right\|^2 d\sigma < \infty.$$

By the matrix variation of constant formula, the unique and time-continuous solutions to the time-backward matrix differential equations (3.25)–(3.26) together with the terminal-value conditions are then written as follows

$$H^i(s,1) = \Phi_{ai}^T(t_f,s)Q_{ai}^f\Phi_{ai}(t_f,s) + \int_s^{t_f}\Phi_{ai}^T(\sigma,s)N_{ai}(\sigma)\Phi_{ai}(\sigma,s)\,d\sigma$$

$$H^i(s,r) = \int_s^{t_f}\Phi_{ai}^T(\sigma,s)\sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}H^i(\sigma,v)G_{ai}(\sigma)W_{ai}G_{ai}^T(\sigma) \times H^i(\sigma,r-v)\Phi_{ai}(\sigma,s)\,d\sigma$$

for any $s \in [t_0, t_f]$ and $2 \le r \le k_i$. It is observed that as long as the growth rate of the integrals is not faster than the exponentially decreasing rate of two factors of $\Phi_{ai}(\cdot,\cdot)$, it is therefore concluded that there exist upper bounds on the unique and time-continuous solutions $\{H^i(s,r)\}_{r=1}^{k_i}$ for any time interval $[t_0, t_f]$. With the existence and boundedness of the solutions of the equations (3.25) and (3.26), it is reasonable to conclude that the unique and time-continuous solutions $\{\breve{D}^i(s,r)\}_{r=1}^{k_i}$ and $\{D^i(s,r)\}_{r=1}^{k_i}$ of the remaining linear vector- and scalar-valued equations (3.28)–(3.30) also exist and are bounded on the time interval $[t_0, t_f]$. □

As for the problem statements of the control decision optimization, the results (3.25)–(3.30) are now interpreted in terms of variables and matrices of the local dynamical system by letting $H^i(s,r)$, $\breve{D}^i(s,r)$ and $G_{ai}(s)W_{ai}G_{ai}^T(s)$ be partitioned as follows

$$H^i(s,r) \triangleq \begin{bmatrix} H_{00}^i(s,r) & H_{01}^i(s,r) & H_{02}^i(s,r) \\ H_{10}^i(s,r) & H_{11}^i(s,r) & H_{12}^i(s,r) \\ H_{20}^i(s,r) & H_{21}^i(s,r) & H_{22}^i(s,r) \end{bmatrix}; \qquad \breve{D}^i(s,r) \triangleq \begin{bmatrix} \breve{D}_0^i(s,r) \\ \breve{D}_1^i(s,r) \\ \breve{D}_2^i(s,r) \end{bmatrix}$$

$$G_{ai}(s)W_{ai}G_{ai}^T(s) \triangleq \begin{bmatrix} \Pi_{00}^i(s) & \Pi_{01}^i(s) & \Pi_{02}^i(s) \\ \Pi_{10}^i(s) & \Pi_{11}^i(s) & \Pi_{12}^i(s) \\ \Pi_{20}^i(s) & \Pi_{21}^i(s) & \Pi_{22}^i(s) \end{bmatrix}$$

$$= \begin{bmatrix} L_i(s)V_iL_i^T(s) & -L_i(s)V_iL_i^T(s) & 0 \\ -L_i(s)V_iL_i^T(s) & L_i(s)V_iL_i^T(s) + G_i(s)W_iG_i^T(s) & 0 \\ 0 & 0 & G_{zi}(s)W_{mi}G_{zi}^T(s) \end{bmatrix}$$

$$H^i(s,r)G_{ai}(s)W_{ai}G_{ai}^T(s) \triangleq \begin{bmatrix} P_{00}^i(s) & P_{01}^i(s) & P_{02}^i(s) \\ P_{10}^i(s) & P_{11}^i(s) & P_{12}^i(s) \\ P_{20}^i(s) & P_{21}^i(s) & P_{22}^i(s) \end{bmatrix}$$

whose the matrix components for agent i are defined by

$$P_{00}^i(s) \triangleq H_{00}^i(s,r)\Pi_{00}^i(s) + H_{01}^i(s,r)\Pi_{10}^i(s) + H_{02}^i(s,r)\Pi_{20}^i(s)$$
$$P_{01}^i(s) \triangleq H_{00}^i(s,r)\Pi_{01}^i(s) + H_{01}^i(s,r)\Pi_{11}^i(s) + H_{02}^i(s,r)\Pi_{21}^i(s)$$
$$P_{02}^i(s) \triangleq H_{00}^i(s,r)\Pi_{02}^i(s) + H_{01}^i(s,r)\Pi_{12}^i(s) + H_{02}^i(s,r)\Pi_{22}^i(s)$$
$$P_{10}^i(s) \triangleq H_{10}^i(s,r)\Pi_{00}^i(s) + H_{11}^i(s,r)\Pi_{10}^i(s) + H_{12}^i(s,r)\Pi_{20}^i(s)$$
$$P_{11}^i(s) \triangleq H_{10}^i(s,r)\Pi_{01}^i(s) + H_{11}^i(s,r)\Pi_{11}^i(s) + H_{12}^i(s,r)\Pi_{21}^i(s)$$
$$P_{12}^i(s) \triangleq H_{10}^i(s,r)\Pi_{02}^i(s) + H_{11}^i(s,r)\Pi_{12}^i(s) + H_{12}^i(s,r)\Pi_{22}^i(s)$$
$$P_{20}^i(s) \triangleq H_{20}^i(s,r)\Pi_{00}^i(s) + H_{21}^i(s,r)\Pi_{10}^i(s) + H_{22}^i(s,r)\Pi_{20}^i(s)$$
$$P_{21}^i(s) \triangleq H_{20}^i(s,r)\Pi_{01}^i(s) + H_{21}^i(s,r)\Pi_{11}^i(s) + H_{22}^i(s,r)\Pi_{21}^i(s)$$
$$P_{22}^i(s) \triangleq H_{20}^i(s,r)\Pi_{02}^i(s) + H_{21}^i(s,r)\Pi_{12}^i(s) + H_{22}^i(s,r)\Pi_{22}^i(s)$$

Then the previous result can be expanded as follows.

Theorem 3 (Performance-measure statistics) *For each agent i, let the pairs (A_i, B_i) and (A_i, C_i) be uniformly stabilizable and the pair (A_i, H_i) be uniformly detectable on $[t_0, t_f]$ in the autonomous system governed by (3.3) through (3.7). Then, for any given $k_i \in \mathbb{Z}^+$, the k_i-th cumulant associated with (3.23) is given by*

$$\kappa_{k_i}^i = \left(x_i^0\right)^T H_{00}^i(t_0, k_i)x_i^0 + 2\left(x_i^0\right)^T \breve{D}_0^i(t_0, k_i) + D^i(t_0, k_i) \tag{3.31}$$

where the cumulant components $\{H_{00}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{01}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{02}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{10}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{11}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{12}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{20}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{21}^i(s,r)\}_{r=1}^{k_i}$, $\{H_{22}^i(s,r)\}_{r=1}^{k_i}$, $\{\breve{D}_0^i(s,r)\}_{r=1}^{k_i}$, $\{\breve{D}_1^i(s,r)\}_{r=1}^{k_i}$, $\{\breve{D}_2^i(s,r)\}_{r=1}^{k_i}$ and $\{D^i(s,r)\}_{r=1}^{k_i}$ evaluated at $s = t_0$ satisfy the supporting matrix-valued differential equations

$$\begin{aligned}\frac{d}{ds}H_{00}^i(s,1) = &-\left[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\right]^T H_{00}^i(s,1) \\ &- H_{00}^i(s,1)\left[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\right] \\ &- K_i^T(s)R_i(s)K_i(s) - Q_i(s) - K_{zi}^T(s)R_{zi}(s)K_{zi}(s),\end{aligned}$$

$$H_{00}^i(t_f,1) = Q_i^f \tag{3.32}$$

$$\begin{aligned}\frac{d}{ds}H_{01}^i(s,1) &= -\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T H_{01}^i(s,1) - Q_i(s)\\ &\quad - H_{00}^i(s,1)L_i(s)H_i(s) - H_{01}^i(s,1)\big[A_i(s)-L_i(s)H_i(s)\big],\\ H_{01}^i(t_f,1) &= Q_i^f\end{aligned} \tag{3.33}$$

$$\begin{aligned}\frac{d}{ds}H_{02}^i(s,1) &= -\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T H_{02}^i(s,1)\\ &\quad - H_{02}^i(s,1)A_{zi}(s), \quad H_{02}^i(t_f,1)=0\end{aligned} \tag{3.34}$$

$$\begin{aligned}\frac{d}{ds}H_{10}^i(s,1) &= -\big[L_i(s)H_i(s)\big]^T H_{00}^i(s,1) - \big[A_i(s)-L_i(s)H_i(s)\big]^T H_{10}^i(s,1)\\ &\quad - H_{10}^i(s,1)\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big] - Q_i(s),\\ H_{10}^i(t_f,1) &= Q_i^f\end{aligned} \tag{3.35}$$

$$\begin{aligned}\frac{d}{ds}H_{11}^i(s,1) &= -\big[L_i(s)H_i(s)\big]^T H_{01}^i(s,1) - \big[A_i(s)-L_i(s)H_i(s)\big]^T H_{11}^i(s,1)\\ &\quad - H_{10}^i(s,1)L_i(s)H_i(s) - H_{11}^i(s,1)\big[A_i(s)-L_i(s)H_i(s)\big],\\ H_{11}^i(t_f,1) &= Q_i^f\end{aligned} \tag{3.36}$$

$$\begin{aligned}\frac{d}{ds}H_{12}^i(s,1) &= -\big[L_i(s)H_i(s)\big]^T H_{02}^i(s,1) - \big[A_i(s)-L_i(s)H_i(s)\big]^T H_{12}^i(s,1)\\ &\quad - H_{12}^i(s,1)A_{zi}(s), \quad H_{12}^i(t_f,1)=0\end{aligned} \tag{3.37}$$

$$\begin{aligned}\frac{d}{ds}H_{20}^i(s,1) &= -H_{20}^i(s,1)\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]\\ &\quad - A_{zi}^T(s)H_{20}^i(s,1) + 2R_{zi}(s)K_{zi}(s), \quad H_{20}^i(t_f,1)=0\end{aligned} \tag{3.38}$$

$$\begin{aligned}\frac{d}{ds}H_{21}^i(s,1) &= -A_{zi}^T(s)H_{21}^i(s,1) - H_{20}^i(s,1)\big[L_i(s)H_i(s)\big]\\ &\quad - H_{21}^i(s,1)\big[A_i(s)-L_i(s)H_i(s)\big], \quad H_{21}^i(t_f,1)=0\end{aligned} \tag{3.39}$$

$$\frac{d}{ds}H_{22}^i(s,1) = -A_{zi}^T(s)H_{22}^i(s,1) - H_{22}^i(s,1)A_{zi}(s) - R_{zi}(s), \quad H_{22}^i(t_f,1)=0 \tag{3.40}$$

$$\begin{aligned}\frac{d}{ds}H_{00}^i(s,r) &= -\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T H_{00}^i(s,r)\\ &\quad - H_{00}^i(s,r)\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]\\ &\quad - \sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}\big[P_{00}^i(s)H_{00}^i(s,r-v) + P_{01}^i(s)H_{01}^i(s,r-v)\end{aligned}$$

$$+ P_{02}^i(s)H_{20}^i(s,r-v)\big], \quad H_{00}^i(t_f,r)=0, \quad 2 \le r \le k_i \tag{3.41}$$

$$\begin{aligned}\frac{d}{ds}H_{01}^i(s,r) &= -\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T H_{01}^i(s,r)\\ &\quad - H_{00}^i(s,r)\big[L_i(s)H_i(s)\big] - H_{01}^i(s,r)\big[A_i(s)-L_i(s)H_i(s)\big]\\ &\quad -\sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}\big[P_{00}^i(s)H_{01}^i(s,r-v)+P_{01}^i(s)H_{11}^i(s,r-v)\\ &\quad + P_{02}^i(s)H_{21}^i(s,r-v)\big], \quad H_{01}^i(t_f,r)=0, \quad 2\le r\le k_i\end{aligned} \tag{3.42}$$

$$\begin{aligned}\frac{d}{ds}H_{02}^i(s,r) &= -\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T H_{02}^i(s,r)\\ &\quad -\sum_{r=1}^{r-1}\frac{2r!}{v!(r-v)!}\big[P_{00}^i(s)H_{02}^i(s,r-v)+P_{01}^i(s)H_{12}^i(s,r-v)\\ &\quad + P_{02}^i(s)H_{22}^i(s,r-v)\big] - H_{02}^i(s,r)A_{zi}(s), \quad H_{02}^i(t_f,r)=0,\\ &2\le r\le k_i\end{aligned} \tag{3.43}$$

$$\begin{aligned}\frac{d}{ds}H_{10}^i(s,r) &= -\big[L_i(s)H_i(s)\big]^T H_{00}^i(s,r) - \big[A_i(s)-L_i(s)H_i(s)\big]^T H_{10}^i(s,r)\\ &\quad - H_{10}^i(s,r)\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]\\ &\quad -\sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}\big[P_{10}^i(s)H_{00}^i(s,r-v)+P_{11}^i(s)H_{10}^i(s,r-v)\\ &\quad + P_{12}^i(s)H_{20}^i(s,r-v)\big], \quad H_{10}^i(t_f,r)=0, \quad 2\le r\le k_i\end{aligned} \tag{3.44}$$

$$\begin{aligned}\frac{d}{ds}H_{11}^i(s,r) &= -\big[L_i(s)H_i(s)\big]^T H_{01}^i(s,r) - \big[A_i(s)-L_i(s)H_i(s)\big]^T H_{11}^i(s,r)\\ &\quad - H_{10}^i(s,r)\big[L_i(s)H_i(s)\big] - H_{11}^i(s,r)\big[A_i(s)-L_i(s)H_i(s)\big]\\ &\quad -\sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}\big[P_{10}^i(s)H_{01}^i(s,r-v)+P_{11}^i(s)H_{11}^i(s,r-v)\\ &\quad + P_{12}^i(s)H_{21}^i(s,r-v)\big], \quad H_{11}^i(t_f,r)=0, \quad 2\le r\le k_i\end{aligned} \tag{3.45}$$

$$\begin{aligned}\frac{d}{ds}H_{12}^i(s,r) &= -\big[L_i(s)H_i(s)\big]^T H_{02}^i(s,r) - \big[A_i(s)-L_i(s)H_i(s)\big]^T H_{12}^i(s,r)\\ &\quad -\sum_{v=1}^{r-1}\frac{2r!}{v!(r-v)!}\big[P_{10}^i(s)H_{02}^i(s,r-v)+P_{11}^i(s)H_{12}^i(s,r-v)\\ &\quad + P_{12}^i(s)H_{22}^i(s,r-v)\big] - H_{12}^i(s,r)A_{zi}(s), \quad H_{12}^i(t_f,r)=0,\\ &2\le r\le k_i\end{aligned} \tag{3.46}$$

$$\frac{d}{ds}H_{20}^i(s,r) = -A_{zi}^T(s)H_{20}^i(s,r) - H_{20}^i(s,r)\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]$$
$$- \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{20}^i(s)H_{00}^i(s,r-v) + P_{21}^i(s)H_{10}^i(s,r-v)$$
$$+ P_{22}^i(s)H_{20}^i(s,r-v)\big], \quad H_{20}^i(t_f,r) = 0, \quad 2 \le r \le k_i \tag{3.47}$$

$$\frac{d}{ds}H_{21}^i(s,r) = -A_{zi}^T(s)H_{21}^i(s,r) - H_{20}^i(s,r)\big[L_i(s)H_i(s)\big]$$
$$- \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{20}^i(s)H_{01}^i(s,r-v) + P_{21}^i(s)H_{11}^i(s,r-v)$$
$$+ P_{22}^i(s)H_{21}^i(s,r-v)\big] - H_{21}^i(s,r)\big[A_i(s) - L_i(s)H_i(s)\big],$$
$$H_{21}^i(t_f,r) = 0, \quad 2 \le r \le k_i \tag{3.48}$$

$$\frac{d}{ds}H_{22}^i(s,r) = -A_{zi}^T(s)H_{22}^i(s,r) - H_{22}^i(s,r)A_{zi}(s)$$
$$- \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{20}^i(s)H_{02}^i(s,r-v) + P_{21}^i(s)H_{12}^i(s,r-v)$$
$$+ P_{22}^i(s)H_{22}^i(s,r-v)\big], \quad H_{22}^i(t_f,r) = 0, \quad 2 \le r \le k_i \tag{3.49}$$

$$\frac{d}{ds}\breve{D}_0^i(s,1) = -\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]^T \breve{D}_0^i(s,1)$$
$$- H_{00}^i(s,1)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big]$$
$$- H_{02}^i(s,1)E_{zi}(s)d_{mi}(s) - K_i^T(s)R_i(s)p_i(s)$$
$$- K_{zi}^T(s)R_{zi}(s)p_{zi}(s), \quad \breve{D}_0^i(t_f,1) = 0 \tag{3.50}$$

$$\frac{d}{ds}\breve{D}_1^i(s,1) = -\big[L_i(s)H_i(s)\big]^T \breve{D}_0^i(s,1) - \big[A_i(s) - L_i(s)H_i(s)\big]^T \breve{D}_1^i(s,1)$$
$$- H_{10}^i(s,1)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big]$$
$$- H_{12}^i(s,1)E_{zi}(s)d_{mi}(s), \quad \breve{D}_1^i(t_f,1) = 0 \tag{3.51}$$

$$\frac{d}{ds}\breve{D}_2^i(s,1) = -H_{20}^i(s,1)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big]$$
$$- A_{zi}^T(s)\breve{D}_2^i(s,1) - H_{22}^i(s,1)E_{zi}(s)d_{mi}(s) + R_{zi}(s)p_{zi}(s),$$
$$\breve{D}_2^i(t_f,1) = 0 \tag{3.52}$$

$$\frac{d}{ds}\breve{D}_0^i(s,r) = -\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]^T \breve{D}_0^i(s,r)$$

$$
\begin{aligned}
&- H^i_{00}(s,r)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big] \\
&- H^i_{02}(s,r)E_{zi}(s)d_{mi}(s), \quad \breve{D}^i_0(t_f,r) = 0, \quad 2 \le r \le k_i
\end{aligned} \tag{3.53}
$$

$$
\begin{aligned}
\frac{d}{ds}\breve{D}^i_1(s,r) &= -\big[L_i(s)H_i(s)\big]^T \breve{D}^i_0(s,r) - \big[A_i(s) - L_i(s)H_i(s)\big]^T \breve{D}^i_1(s,r) \\
&\quad - H^i_{10}(s,r)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big] \\
&\quad - H^i_{12}(s,r)E_{zi}(s)d_{mi}(s), \quad \breve{D}^i_1(t_f,r) = 0, \quad 2 \le r \le k_i
\end{aligned} \tag{3.54}
$$

$$
\begin{aligned}
\frac{d}{ds}\breve{D}^i_2(s,r) &= -H^i_{20}(s,r)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big] \\
&\quad - A^T_{zi}(s)\breve{D}^i_2(s,r) - H^i_{22}(s,r)E_{zi}(s)d_{mi}(s), \quad \breve{D}^i_2(t_f,r) = 0, \\
&2 \le r \le k_i
\end{aligned} \tag{3.55}
$$

$$
\begin{aligned}
\frac{d}{ds}D^i(s,1) &= -2\big(\breve{D}^i_0\big)^T(s,1)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big] \\
&\quad - 2\big(\breve{D}^i_2\big)^T(s,1)E_{zi}(s)d_{mi}(s) \\
&\quad - Tr\big\{H^i_{00}(s,1)\Pi^i_{00}(s) + H^i_{01}(s,1)\Pi^i_{10}(s) + H^i_{02}(s,1)\Pi^i_{20}(s)\big\} \\
&\quad - Tr\big\{H^i_{10}(s,1)\Pi^i_{01}(s) + H^i_{11}(s,1)\Pi^i_{11}(s) + H^i_{12}(s,1)\Pi^i_{21}(s)\big\} \\
&\quad - Tr\big\{H^i_{20}(s,1)\Pi^i_{02}(s) + H^i_{21}(s,1)\Pi^i_{12}(s) + H^i_{22}(s,1)\Pi^i_{22}(s)\big\} \\
&\quad - p^T_i(s)R_i(s)p_i(s) - p^T_{zi}(s)R_{zi}(s)p_{zi}(s), \quad D^i(t_f,1) = 0
\end{aligned} \tag{3.56}
$$

$$
\begin{aligned}
\frac{d}{ds}D^i(s,r) &= -2\big(\breve{D}^i_0\big)^T(s,r)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big] \\
&\quad - 2\big(\breve{D}^i_2\big)^T(s,r)E_{zi}(s)d_{mi}(s) \\
&\quad - Tr\big\{H^i_{00}(s,r)\Pi^i_{00}(s) + H^i_{01}(s,r)\Pi^i_{10}(s)\big\} \\
&\quad - Tr\big\{H^i_{02}(s,r)\Pi^i_{20}(s)\big\} - Tr\big\{H^i_{10}(s,r)\Pi^i_{01}(s) + H^i_{11}(s,r)\Pi^i_{11}(s)\big\} \\
&\quad - Tr\big\{H^i_{12}(s,r)\Pi^i_{21}(s)\big\} - Tr\big\{H^i_{20}(s,r)\Pi^i_{02}(s) + H^i_{21}(s,r)\Pi^i_{12}(s)\big\} \\
&\quad - Tr\big\{H^i_{22}(s,r)\Pi^i_{22}(s)\big\}, \quad D^i(t_f,r) = 0, \quad 2 \le r \le k_i
\end{aligned} \tag{3.57}
$$

Clearly, then, the compactness offered by logic from the state-space model description (3.3) through (3.7) has been successfully combined with the quantitativity from the a priori knowledge about probabilistic descriptions of uncertain environments. Thus, the uncertainty of performance (3.23) at agent i can now be represented in a compact and robust way. Subsequently, the time-backward differential equations (3.32)–(3.57) not only offer a tractable procedure for the calculation of (3.31) but also allow the incorporation of a class of linear feedback syntheses so that agent i is actively mitigating its performance uncertainty. Such performance-

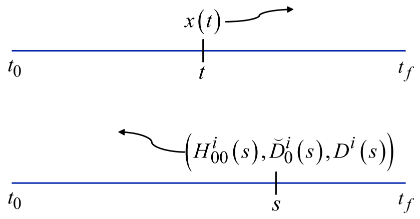


Fig. 3.3 The initial cost problem (*top*) vs. the terminal cost problem (*bottom*)

measure statistics are therefore, referred as "information" statistics which are extremely valuable for shaping the local performance distribution.

3.4 Problem Statements

Suffice it to say here that all the performance-measure statistics (3.31) depend in part on the known initial condition $x_i(t_0)$. Although different states $x_i(t)$ will result in different values for "performance-to-come", the values of (3.31) are, however, functions of time-backward evolutions of the cumulant-generating variables $H^i_{00}(s,r)$, $\breve{D}^i_0(s,r)$ and $D^i(s,r)$ that totally ignore all the intermediate values of $x_i(t)$ except at the a priori initial system state $x_i(t_0)$. This fact therefore makes this new optimization problem in statistical control particularly unique as compared with the more traditional dynamic programming class of investigations. In other words, the time-backward trajectories (3.32)–(3.57) should be considered as the "new" dynamical equations for which the resulting Mayer optimization and associated value function in the framework of dynamic programming [4] thus depend on these "new" state variables $H^i_{00}(s,r)$, $\breve{D}^i_0(s,r)$ and $D^i(s,r)$. See Fig. 3.3 for further illustrations of the initial cost problem in statistical control as opposed to the terminal cost problem in classical control.

For notational simplicity, it is now convenient to denote the right members of the cumulant-generating equations (3.32)–(3.57) as the mappings on the finite horizon $[t_0, t_f]$ with the rules of action

$$\begin{aligned}
F^{i,1}_{00}\left(s, Y^i_{00}; K_i, K_{zi}\right) = &-\left[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\right]^T Y^{i,1}_{00}(s) \\
&- Y^{i,1}_{00}(s)\left[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\right] \\
&- K_i^T(s)R_i(s)K_i(s) - Q_i(s) - K_{zi}^T(s)R_{zi}(s)K_{zi}(s)
\end{aligned}$$

$$\begin{aligned}
&F^{i,1}_{01}\left(s, Y^i_{00}, Y^i_{01}; K_i, K_{zi}\right) \\
&\quad = -\left[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\right]^T Y^{i,1}_{01}(s) \\
&\qquad - Y^{i,1}_{00}(s)\left[L_i(s)H_i(s)\right] - Y^{i,1}_{01}(s)\left[A_i(s) - L_i(s)H_i(s)\right] - Q_i(s)
\end{aligned}$$

$$\begin{aligned}
&F^{i,1}_{02}\left(s, Y^i_{02}; K_i, K_{zi}\right) \\
&\quad = -Y^{i,1}_{02}(s)A_{zi}(s) - \left[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\right]^T Y^{i,1}_{02}(s)
\end{aligned}$$

$$F_{10}^{i,1}\big(s, Y_{00}^i, Y_{10}^i; K_i, K_{zi}\big)$$
$$= -\big[A_i(s) - L_i(s)H_i(s)\big]^T Y_{10}^{i,1}(s) - \big[L_i(s)H_i(s)\big]^T Y_{00}^{i,1}(s)$$
$$- Y_{10}^{i,1}(s)\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big] - Q_i(s)$$
$$F_{11}^{i,1}\big(s, Y_{01}^i, Y_{10}^i, Y_{11}^i\big) = -Y_{10}^{i,1}(s)\big[L_i(s)H_i(s)\big] - Y_{11}^{i,1}(s)\big[A_i(s) - L_i(s)H_i(s)\big]$$
$$- \big[L_i(s)H_i(s)\big]^T Y_{01}^{i,1}(s) - \big[A_i(s) - L_i(s)H_i(s)\big]^T Y_{11}^{i,1}(s)$$
$$F_{12}^{i,1}\big(s, Y_{02}^i, Y_{12}^i\big) = -Y_{12}^{i,1}(s)A_{zi}(s) - \big[L_i(s)H_i(s)\big]^T Y_{02}^{i,1}(s)$$
$$- \big[A_i(s) - L_i(s)H_i(s)\big]^T Y_{12}^{i,1}(s)$$
$$F_{20}^{i,1}\big(s, Y_{20}^i; K_i, K_{zi}\big) = -Y_{20}^{i,1}(s)\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]$$
$$- A_{zi}^T(s)Y_{20}^{i,1}(s) + 2R_{zi}(s)K_{zi}(s)$$
$$F_{21}^{i,1}\big(s, Y_{20}^i, Y_{21}^i\big) = -A_{zi}^T(s)Y_{21}^{i,1}(s) - Y_{20}^{i,1}(s)\big[L_i(s)H_i(s)\big]$$
$$- Y_{21}^{i,1}(s)\big[A_i(s) - L_i(s)H_i(s)\big]$$
$$F_{22}^{i,1}\big(s, Y_{22}^i\big) = -A_{zi}^T(s)Y_{22}^{i,1}(s) - Y_{22}^{i,1}(s)A_{zi}(s) - R_{zi}(s)$$
$$F_{00}^{i,r}\big(s, Y_{00}^i, Y_{01}^i, Y_{20}^i; K_i, K_{zi}\big)$$
$$= -\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]^T Y_{00}^{i,r}(s)$$
$$- Y_{00}^{i,r}(s)\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]$$
$$- \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{00}^i(s)Y_{00}^{i,r-v}(s) + P_{01}^i(s)Y_{01}^{i,r-v}(s) + P_{02}^i(s)Y_{20}^{i,r-v}(s)\big]$$
$$F_{01}^{i,r}\big(s, Y_{00}^i, Y_{01}^i, Y_{11}^i, Y_{21}^i; K_i, K_{zi}\big)$$
$$= -Y_{01}^{i,r}(s)\big[A_i(s) - L_i(s)H_i(s)\big]$$
$$- Y_{00}^{i,r}(s)\big[L_i(s)H_i(s)\big] - \big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]^T Y_{01}^{i,r}(s)$$
$$- \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{00}^i(s)Y_{01}^{i,r-v}(s) + P_{01}^i(s)Y_{11}^{i,r-v}(s) + P_{02}^i(s)Y_{21}^{i,r-v}(s)\big]$$
$$F_{02}^{i,r}\big(s, Y_{02}^i, Y_{12}^i, Y_{22}^i; K_i, K_{zi}\big)$$
$$= -Y_{02}^{i,r}(s)A_{zi}(s) - \big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]^T Y_{02}^{i,r}(s)$$
$$- \sum_{r=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{00}^i(s)Y_{02}^{i,r-v}(s) + P_{01}^i(s)Y_{12}^{i,r-v}(s) + P_{02}^i(s)Y_{22}^{i,r-v}(s)\big]$$

$$
\begin{aligned}
&F_{10}^{i,r}\big(s, Y_{00}^i, Y_{10}^i, Y_{20}^i; K_i, K_{zi}\big)\\
&\quad = -\big[A_i(s) - L_i(s)H_i(s)\big]^T Y_{10}^{i,r}(s)\\
&\qquad - \big[L_i(s)H_i(s)\big]^T Y_{00}^{i,r}(s) - Y_{10}^{i,r}(s)\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]\\
&\qquad - \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{10}^i(s)Y_{00}^{i,r-v}(s) + P_{11}^i(s)Y_{10}^{i,r-v}(s) + P_{12}^i(s)Y_{20}^{i,r-v}(s)\big]\\
&F_{11}^{i,r}\big(s, Y_{01}^i, Y_{10}^i, Y_{11}^i, Y_{21}^i\big)\\
&\quad = -\big[A_i(s) - L_i(s)H_i(s)\big]^T Y_{11}^{i,r}(s) - \big[L_i(s)H_i(s)\big]^T Y_{01}^{i,r}(s)\\
&\qquad - Y_{10}^{i,r}(s)\big[L_i(s)H_i(s)\big] - Y_{11}^{i,r}(s)\big[A_i(s) - L_i(s)H_i(s)\big]\\
&\qquad - \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{10}^i(s)Y_{01}^{i,r-v}(s) + P_{11}^i(s)Y_{11}^{i,r-v}(s) + P_{12}^i(s)Y_{21}^{i,r-v}(s)\big]\\
&F_{12}^{i,r}\big(s, Y_{02}^i, Y_{12}^i, Y_{22}^i\big)\\
&\quad = -Y_{12}^{i,r}(s)A_{zi}(s) - \big[L_i(s)H_i(s)\big]^T Y_{02}^{i,r}(s) - \big[A_i(s) - L_i(s)H_i(s)\big]^T Y_{12}^{i,r}(s)\\
&\qquad - \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{10}^i(s)Y_{02}^{i,r-v}(s) + P_{11}^i(s)Y_{12}^{i,r-v}(s) + P_{12}^i(s)Y_{22}^{i,r-v}(s)\big]\\
&F_{20}^{i,r}(s) = -A_{zi}^T(s)Y_{20}^{i,r}(s) - Y_{20}^{i,r}(s)\big[A_i(s) + B_i(s)K_i(s) + C_i(s)K_{zi}(s)\big]\\
&\qquad\qquad - \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{20}^i(s)Y_{00}^{i,r-v}(s) + P_{21}^i(s)Y_{10}^{i,r-v}(s)\\
&\qquad\qquad + P_{22}^i(s)Y_{20}^{i,r-v}(s)\big]\\
&F_{21}^{i,r}\big(s, Y_{01}^i, Y_{11}^i, Y_{20}^i, Y_{21}^i\big)\\
&\quad = -A_{zi}^T(s)Y_{21}^{i,r}(s) - Y_{20}^{i,r}(s)\big[L_i(s)H_i(s)\big]\\
&\qquad - \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{20}^i(s)Y_{01}^{i,r-v}(s) + P_{21}^i(s)Y_{11}^{i,r-v}(s) + P_{22}^i(s)Y_{21}^{i,r-v}(s)\big]\\
&\qquad - Y_{21}^{i,r}(s)\big[A_i(s) - L_i(s)H_i(s)\big]\\
&F_{22}^{i,r}\big(s, Y_{02}^i, Y_{12}^i, Y_{22}^i\big)\\
&\quad = -A_{zi}^T(s)Y_{22}^{i,r}(s) - Y_{22}^{i,r}(s)A_{zi}(s) - \sum_{v=1}^{r-1} \frac{2r!}{v!(r-v)!}\big[P_{20}^i(s)Y_{02}^{i,r-v}(s)\\
&\qquad + P_{21}^i(s)Y_{12}^{i,r-v}(s) + P_{22}^i(s)Y_{22}^{i,r-v}(s)\big]
\end{aligned}
$$

$$\begin{aligned}
&\breve{G}_0^{i,1}\big(s,\breve{Z}_0^i,Y_{00}^i,Y_{02}^i;K_i,K_{zi};p_i,p_{zi}\big)\\
&\quad=-Y_{02}^{i,1}(s)E_{zi}(s)d_{mi}(s)-Y_{00}^{i,1}(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&\qquad-K_{zi}^T(s)R_{zi}(s)p_{zi}(s)-\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T\breve{Z}_0^{i,1}(s)\\
&\qquad-K_i^T(s)R_i(s)p_i(s)\\
&\breve{G}_1^{i,1}\big(s,Y_{10}^i,Y_{12}^i,\breve{Z}_0^i,\breve{Z}_1^i;p_i,p_{zi}\big)\\
&\quad=-\big[A_i(s)-L_i(s)H_i(s)\big]^T\breve{Z}_1^{i,1}(s)\\
&\qquad-Y_{10}^{i,1}(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&\qquad-\big[L_i(s)H_i(s)\big]^T\breve{Z}_0^{i,1}(s)-Y_{12}^{i,1}(s)E_{zi}(s)d_{mi}(s)\\
&\breve{G}_2^{i,1}\big(s,Y_{20}^i,Y_{22}^i,\breve{Z}_2^i;p_i,p_{zi}\big)\\
&\quad=-Y_{20}^{i,1}(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&\qquad-A_{zi}^T(s)\breve{Z}_2^{i,1}(s)-Y_{22}^{i,1}(s)E_{zi}(s)d_{mi}(s)+R_{zi}(s)p_{zi}(s)\\
&\breve{G}_0^{i,r}\big(s,Y_{00}^i,Y_{02}^i,\breve{Z}_0^i;K_i,K_{zi};p_i,p_{zi}\big)\\
&\quad=-Y_{02}^{i,r}(s)E_{zi}(s)d_{mi}(s)-Y_{00}^{i,r}(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&\qquad-\big[A_i(s)+B_i(s)K_i(s)+C_i(s)K_{zi}(s)\big]^T\breve{Z}_0^{i,r}(s)\\
&\breve{G}_1^{i,r}\big(s,Y_{10}^i,Y_{12}^i,\breve{Z}_0^i,\breve{Z}_1^i;p_i,p_{zi}\big)\\
&\quad=-\big[A_i(s)-L_i(s)H_i(s)\big]^T\breve{Z}_1^{i,r}(s)\\
&\qquad-Y_{10}^{i,r}(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&\qquad-\big[L_i(s)H_i(s)\big]^T\breve{Z}_0^{i,r}(s)-Y_{12}^{i,r}(s)E_{zi}(s)d_{mi}(s)\\
&\breve{G}_2^{i,r}\big(s,Y_{20}^i,Y_{22}^i,\breve{Z}_2^i;p_i,p_{zi}\big)\\
&\quad=-A_{zi}^T(s)\breve{Z}_2^{i,r}(s)-Y_{22}^{i,r}(s)E_{zi}(s)d_{mi}(s)\\
&\qquad-Y_{20}^{i,r}(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&G^{i,1}\big(s,Y_{00}^i,Y_{01}^i,Y_{02}^i,Y_{10}^i,Y_{11}^i,Y_{12}^i,Y_{20}^i,Y_{21}^i,Y_{22}^i,\breve{Z}_0^i,\breve{Z}_2^i;p_i,p_{zi}\big)\\
&\quad=-2\big(\breve{Z}_0^{i,1}\big)^T(s)\big[B_i(s)p_i(s)+C_i(s)p_{zi}(s)+E_i(s)d_i(s)\big]\\
&\qquad-2\big(\breve{Z}_2^{i,1}\big)^T(s)E_{zi}(s)d_{mi}(s)\\
&\qquad-Tr\big\{Y_{00}^{i,1}(s)\Pi_{00}^i(s)+Y_{01}^{i,1}(s)\Pi_{10}^i(s)+Y_{02}^{i,1}(s)\Pi_{20}^i(s)\big\}\\
&\qquad-Tr\big\{Y_{10}^{i,1}(s)\Pi_{01}^i(s)+Y_{11}^{i,1}(s)\Pi_{11}^i(s)+Y_{12}^{i,1}(s)\Pi_{21}^i(s)\big\}
\end{aligned}$$

$$- Tr\{Y_{20}^{i,1}(s)\Pi_{02}^{i}(s) + Y_{21}^{i,1}(s)\Pi_{12}^{i}(s) + Y_{22}^{i,1}(s)\Pi_{22}^{i}(s)\}$$
$$- p_i^T(s)R_i(s)p_i(s) - p_{zi}^T(s)R_{zi}(s)p_{zi}(s)$$
$$G^{i,r}\big(s, Y_{00}^i, Y_{01}^i, Y_{02}^i, Y_{10}^i, Y_{11}^i, Y_{12}^i, Y_{20}^i, Y_{21}^i, Y_{22}^i, \breve{Z}_0^i, \breve{Z}_2^i; p_i, p_{zi}\big)$$
$$= -2\big(\breve{Z}_0^{i,r}\big)^T(s)\big[B_i(s)p_i(s) + C_i(s)p_{zi}(s) + E_i(s)d_i(s)\big]$$
$$- 2\big(\breve{Z}_2^{i,r}\big)^T(s)E_{zi}(s)d_{mi}(s)$$
$$- Tr\{Y_{00}^{i,r}(s)\Pi_{00}^{i}(s) + Y_{01}^{i,r}(s)\Pi_{10}^{i}(s) + Y_{02}^{i,r}(s)\Pi_{20}^{i}(s)\}$$
$$- Tr\{Y_{10}^{i,r}(s)\Pi_{01}^{i}(s) + Y_{11}^{i,r}(s)\Pi_{11}^{i}(s) + Y_{12}^{i,r}(s)\Pi_{21}^{i}(s)\}$$
$$- Tr\{Y_{20}^{i,r}(s)\Pi_{02}^{i}(s) + Y_{21}^{i,r}(s)\Pi_{12}^{i}(s) + Y_{22}^{i,r}(s)\Pi_{22}^{i}(s)\}$$

provided that all the k_i-tuple variables are given by

$$Y_{00}^i(\cdot) \triangleq \big(Y_{00}^{i,1}(\cdot), \ldots, Y_{00}^{i,k_i}(\cdot)\big) \equiv \big(H_{00}^i(\cdot,1), \ldots, H_{00}^i(\cdot,k_i)\big)$$
$$Y_{01}^i(\cdot) \triangleq \big(Y_{01}^{i,1}(\cdot), \ldots, Y_{01}^{i,k_i}(\cdot)\big) \equiv \big(H_{01}^i(\cdot,1), \ldots, H_{01}^i(\cdot,k_i)\big)$$
$$Y_{02}^i(\cdot) \triangleq \big(Y_{02}^{i,1}(\cdot), \ldots, Y_{02}^{i,k_i}(\cdot)\big) \equiv \big(H_{02}^i(\cdot,1), \ldots, H_{02}^i(\cdot,k_i)\big)$$
$$Y_{10}^i(\cdot) \triangleq \big(Y_{10}^{i,1}(\cdot), \ldots, Y_{10}^{i,k_i}(\cdot)\big) \equiv \big(H_{10}^i(\cdot,1), \ldots, H_{10}^i(\cdot,k_i)\big)$$
$$Y_{11}^i(\cdot) \triangleq \big(Y_{11}^{i,1}(\cdot), \ldots, Y_{11}^{i,k_i}(\cdot)\big) \equiv \big(H_{11}^i(\cdot,1), \ldots, H_{11}^i(\cdot,k_i)\big)$$
$$Y_{12}^i(\cdot) \triangleq \big(Y_{12}^{i,1}(\cdot), \ldots, Y_{12}^{i,k_i}(\cdot)\big) \equiv \big(H_{12}^i(\cdot,1), \ldots, H_{12}^i(\cdot,k_i)\big)$$
$$Y_{20}^i(\cdot) \triangleq \big(Y_{20}^{i,1}(\cdot), \ldots, Y_{20}^{i,k_i}(\cdot)\big) \equiv \big(H_{20}^i(\cdot,1), \ldots, H_{20}^i(\cdot,k_i)\big)$$
$$Y_{21}^i(\cdot) \triangleq \big(Y_{21}^{i,1}(\cdot), \ldots, Y_{21}^{i,k_i}(\cdot)\big) \equiv \big(H_{21}^i(\cdot,1), \ldots, H_{21}^i(\cdot,k_i)\big)$$
$$Y_{22}^i(\cdot) \triangleq \big(Y_{22}^{i,1}(\cdot), \ldots, Y_{22}^{i,k_i}(\cdot)\big) \equiv \big(H_{22}^i(\cdot,1), \ldots, H_{22}^i(\cdot,k_i)\big)$$
$$\breve{Z}_0^i(\cdot) \triangleq \big(\breve{Z}_0^{i,1}(\cdot), \ldots, \breve{Z}_0^{i,k_i}(\cdot)\big) \equiv \big(\breve{D}_0^i(\cdot,1), \ldots, \breve{D}_0^i(\cdot,k_i)\big)$$
$$\breve{Z}_1^i(\cdot) \triangleq \big(\breve{Z}_1^{i,1}(\cdot), \ldots, \breve{Z}_1^{i,k_i}(\cdot)\big) \equiv \big(\breve{D}_1^i(\cdot,1), \ldots, \breve{D}_1^i(\cdot,k_i)\big)$$
$$\breve{Z}_2^i(\cdot) \triangleq \big(\breve{Z}_2^{i,1}(\cdot), \ldots, \breve{Z}_2^{i,k_i}(\cdot)\big) \equiv \big(\breve{D}_2^i(\cdot,1), \ldots, \breve{D}_2^i(\cdot,k_i)\big)$$
$$Z_0^i(\cdot) \triangleq \big(Z_0^{i,1}(\cdot), \ldots, Z_0^{i,k_i}(\cdot)\big) \equiv \big(D_0^i(\cdot,1), \ldots, D_0^i(\cdot,k_i)\big)$$

Now it is straightforward to establish the product mappings

$$F_{00}^i \triangleq F_{00}^{i,1} \times \cdots \times F_{00}^{i,k_i} \qquad F_{01}^i \triangleq F_{01}^{i,1} \times \cdots \times F_{01}^{i,k_i}$$
$$F_{02}^i \triangleq F_{02}^{i,1} \times \cdots \times F_{02}^{i,k_i} \qquad F_{10}^i \triangleq F_{10}^{i,1} \times \cdots \times F_{10}^{i,k_i}$$
$$F_{11}^i \triangleq F_{11}^{i,1} \times \cdots \times F_{11}^{i,k_i} \qquad F_{12}^i \triangleq F_{12}^{i,1} \times \cdots \times F_{12}^{i,k_i}$$

$$F^i_{20} \triangleq F^{i,1}_{20} \times \cdots \times F^{i,k_i}_{20} \qquad F^i_{21} \triangleq F^{i,1}_{21} \times \cdots \times F^{i,k_i}_{21}$$
$$F^i_{22} \triangleq F^{i,1}_{22} \times \cdots \times F^{i,k_i}_{22} \qquad \breve{G}^i_0 \triangleq \breve{G}^{i,1}_0 \times \cdots \times \breve{G}^{i,k_i}_0$$
$$\breve{G}^i_1 \triangleq \breve{G}^{i,1}_1 \times \cdots \times \breve{G}^{i,k_i}_1 \qquad \breve{G}^i_2 \triangleq \breve{G}^{i,1}_2 \times \cdots \times \breve{G}^{i,k_i}_2$$
$$G^i \triangleq G^{i,1} \times \cdots \times G^{i,k_i}$$

Thus, the dynamic equations (3.32)–(3.57) can be rewritten compactly as follows

$$\frac{d}{ds}Y^i_{00}(s) = F^i_{00}\big(s, Y^i_{00}(s), Y^i_{01}(s), Y^i_{20}(s); K_i(s), K_{zi}(s)\big), \quad Y^i_{00}(t_f) \tag{3.58}$$

$$\frac{d}{ds}Y^i_{01}(s) = F^i_{01}\big(s, Y^i_{00}(s), Y^i_{01}, Y^i_{11}(s), Y^i_{21}(s); K_i(s), K_{zi}(s)\big), \quad Y^i_{01}(t_f) \tag{3.59}$$

$$\frac{d}{ds}Y^i_{02}(s) = F^i_{02}\big(s, Y^i_{02}, Y^i_{12}(s), Y^i_{22}(s); K_i(s), K_{zi}(s)\big), \quad Y^i_{02}(t_f) \tag{3.60}$$

$$\frac{d}{ds}Y^i_{10}(s) = F^i_{10}\big(s, Y^i_{00}(s), Y^i_{10}(s), Y^i_{20}(s); K_i(s), K_{zi}(s)\big), \quad Y^i_{10}(t_f) \tag{3.61}$$

$$\frac{d}{ds}Y^i_{11}(s) = F^i_{11}\big(s, Y^i_{01}(s), Y^i_{10}(s), Y^i_{11}(s), Y^i_{21}(s)\big), \quad Y^i_{11}(t_f) \tag{3.62}$$

$$\frac{d}{ds}Y^i_{12}(s) = F^i_{12}\big(s, Y^i_{02}(s), Y^i_{12}(s), Y^i_{22}(s)\big), \quad Y^i_{12}(t_f) \tag{3.63}$$

$$\frac{d}{ds}Y^i_{20}(s) = F^i_{20}\big(s, Y^i_{00}(s), Y^i_{10}(s), Y^i_{20}(s); K_i(s), K_{zi}(s)\big), \quad Y^i_{20}(t_f) \tag{3.64}$$

$$\frac{d}{ds}Y^i_{21}(s) = F^i_{21}\big(s, Y^i_{01}(s), Y^i_{11}(s), Y^i_{20}(s), Y^i_{21}(s)\big), \quad Y^i_{21}(t_f) \tag{3.65}$$

$$\frac{d}{ds}Y^i_{22}(s) = F^i_{22}\big(s, Y^i_{02}(s), Y^i_{12}(s), Y^i_{22}(s)\big), \quad Y^i_{22}(t_f) \tag{3.66}$$

$$\frac{d}{ds}\breve{Z}^i_0(s) = \breve{G}^i_0\big(s, Y^i_{00}(s), Y^i_{02}(s), \breve{Z}^i_0(s); K_i(s), K_{zi}(s); p_i(s), p_{zi}(s)\big),$$
$$\breve{Z}^i_0(t_f) \tag{3.67}$$

$$\frac{d}{ds}\breve{Z}^i_1(s) = \breve{G}^i_1\big(s, Y^i_{10}(s), Y^i_{12}(s), \breve{Z}^i_0(s), \breve{Z}^i_1(s); p_i(s), p_{zi}(s)\big), \quad \breve{Z}^i_1(t_f) \tag{3.68}$$

$$\frac{d}{ds}\breve{Z}^i_2(s) = \breve{G}^i_2\big(s, Y^i_{20}, Y^i_{22}(s), \breve{Z}^i_2(s); p_i(s), p_{zi}(s)\big), \quad \breve{Z}^i_2(t_f) \tag{3.69}$$

$$\frac{d}{ds}Z^i(s) = G^i\big(s, Y^i_{00}(s), Y^i_{01}(s), Y^i_{02}(s), Y^i_{10}(s), Y^i_{11}(s), Y^i_{12}(s), Y^i_{20}(s), \ldots$$
$$Y^i_{21}(s), Y^i_{22}(s), \breve{Z}^i_0(s), \breve{Z}^i_2(s); p_i(s), p_{zi}(s)\big), \quad Z^i(t_f) \tag{3.70}$$

where the terminal-value conditions are defined by

$$Y^i_{00}(t_f) \triangleq Q^f_i \times 0 \times \cdots \times 0 \qquad Y^i_{01}(t_f) \triangleq Q^f_i \times 0 \times \cdots \times 0$$

$$Y_{02}^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0 \qquad Y_{10}^i(t_f) \triangleq Q_i^f \times 0 \times \cdots \times 0$$

$$Y_{11}^i(t_f) \triangleq Q_i^f \times 0 \times \cdots \times 0 \qquad Y_{12}^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0$$

$$Y_{20}^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0 \qquad Y_{21}^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0$$

$$Y_{22}^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0 \qquad \breve{Z}_0^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0$$

$$\breve{Z}_1^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0 \qquad \breve{Z}_2^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0$$

$$Z^i(t_f) \triangleq 0 \times 0 \times \cdots \times 0$$

Note that for each agent i the product system (3.58)–(3.70) uniquely determines Y_{00}^i, Y_{01}^i, Y_{02}^i, Y_{10}^i, Y_{11}^i, Y_{12}^i, Y_{20}^i, Y_{21}^i, Y_{22}^i, $\breve{Z}_0^i$, $\breve{Z}_1^i$, $\breve{Z}_2^i$, and Z^i once the admissible 4-tuple $(K_i, K_{zi}, p_i, p_{zi})$ is specified. Thus, Y_{00}^i, Y_{01}^i, Y_{02}^i, Y_{10}^i, Y_{11}^i, Y_{12}^i, Y_{20}^i, Y_{21}^i, Y_{22}^i, $\breve{Z}_0^i$, $\breve{Z}_1^i$, $\breve{Z}_2^i$, and Z^i are considered as the functions of K_i, K_{zi}, p_i, and p_{zi}. The performance index for the interconnected system can therefore be formulated in terms of K_i, K_{zi}, p_i, and p_{zi} for agent i.

The subject of risk taking has been of great interest not only to control system designers of engineered systems but also to decision makers of financial systems. One approach to study risk in stochastic control system is exemplified in the ubiquitous theory of linear-quadratic Gaussian (LQG) control whose preference of expected value of performance measure associated with a class of stochastic systems is minimized against all random realizations of the uncertain environment. Other aspects of performance distributions that do not appear in the classical theory of LQG are variance, skewness, kurtosis, etc. For instance, it may nevertheless be true that some performance with negative skewness appears riskier than performance with positive skewness when expectation and variance are held constant. If skewness does, indeed, play an essential role in determining the perception of risk, then the range of applicability of the present theory should be restricted, for example, to symmetric or equally skewed performance measures.

There have been several studies that attempt to generalize the present LQG theory to account for the effects of variance [11] and [6] or of other description of probability density [12] on the perceived riskiness of performance measures. The contribution of this research is to directly address the perception of risk via a selective set of performance distribution characteristics of its outcomes governed by either dispersion, skewness, flatness, etc. or a combination thereof. Figure 3.4 depicts some possible interpretations on measures of performance risk for control decision under uncertainty.

Definition 1 (Risk-value aware performance index) Associate with agent i the $k_i \in \mathbb{Z}^+$ and the sequence $\mu^i = \{\mu_r^i \ge 0\}_{r=1}^{k_i}$ with $\mu_1^i > 0$. Then, for (t_0, x_i^0) given, the risk-value aware performance index

$$\phi_0^i : \{t_0\} \times \left(\mathbb{R}^{n_i \times n_i}\right)^{k_i} \times \left(\mathbb{R}^{n_i}\right)^{k_i} \times \mathbb{R}^{k_i} \mapsto \mathbb{R}^+$$

over a finite optimization horizon is defined by a risk-value model to reflect the tradeoff between value and riskiness of the Chi-squared type performance mea-

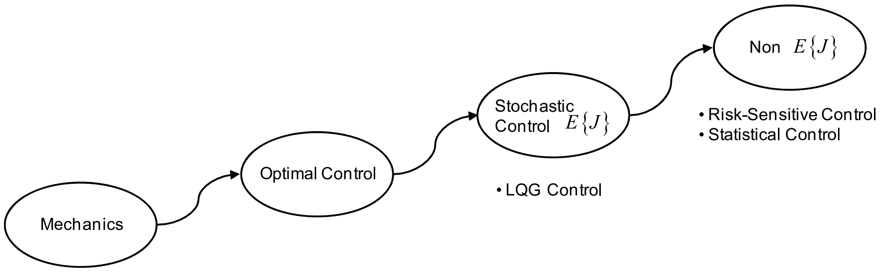


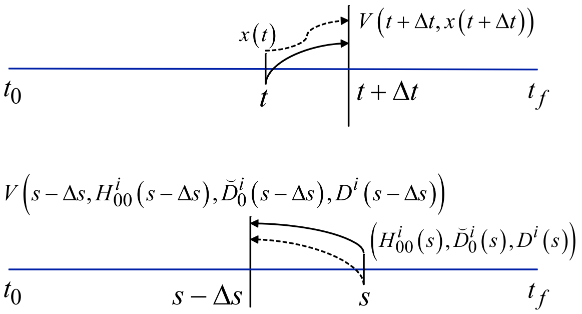
Fig. 3.4 Measures of performance risk for control decision under uncertainty

sure (3.31)

$$
\begin{aligned}
&\phi_0^i\big(t_0, Y_{00}^i(\cdot, K_i, K_{zi}, p_i, p_{zi}), \breve{Z}_0^i(\cdot, K_i, K_{zi}, p_i, p_{zi}), Z^i(\cdot, K_i, K_{zi}, p_i, p_{zi})\big) \\
&\quad \triangleq \underbrace{\mu_1^i \kappa_1^i(K_i, K_{zi}, p_i, p_{zi})}_{\text{Value Measure}} \\
&\qquad + \underbrace{\mu_2^i \kappa_2^i(K_i, K_{zi}, p_i, p_{zi}) + \cdots + \mu_{k_i}^i \kappa_{k_i}^i(K_i, K_{zi}, p_i, p_{zi})}_{\text{Risk Measure}} \\
&\quad = \mu_1^i\big[\big(x_i^0\big)^T Y_{00}^{i,1}(t_0, K_i, K_{zi}, p_i, p_{zi})x_i^0 + 2\big(x_i^0\big)^T \breve{Z}_0^{i,1}(t_0, K_i, K_{zi}, p_i, p_{zi}) \\
&\qquad + Z_0^{i,1}(t_0, K_i, K_{zi}, p_i, p_{zi})\big] + \mu_2^i\big[\big(x_i^0\big)^T Y_{00}^{i,2}(t_0, K_i, K_{zi}, p_i, p_{zi})x_i^0 \\
&\qquad + 2\big(x_i^0\big)^T \breve{Z}_0^{i,2}(t_0, K_i, K_{zi}, p_i, p_{zi}) + Z_0^{i,2}(t_0, K_i, K_{zi}, p_i, p_{zi})\big] \\
&\qquad + \cdots + \mu_{k_i}^i\big[\big(x_i^0\big)^T Y_{00}^{i,k_i}(t_0, K_i, K_{zi}, p_i, p_{zi})x_i^0 \\
&\qquad + 2\big(x_i^0\big)^T \breve{Z}_0^{i,k_i}(t_0, K_i, K_{zi}, p_i, p_{zi}) + Z_0^{i,k_i}(t_0, K_i, K_{zi}, p_i, p_{zi})\big]
\end{aligned}
\tag{3.71}
$$

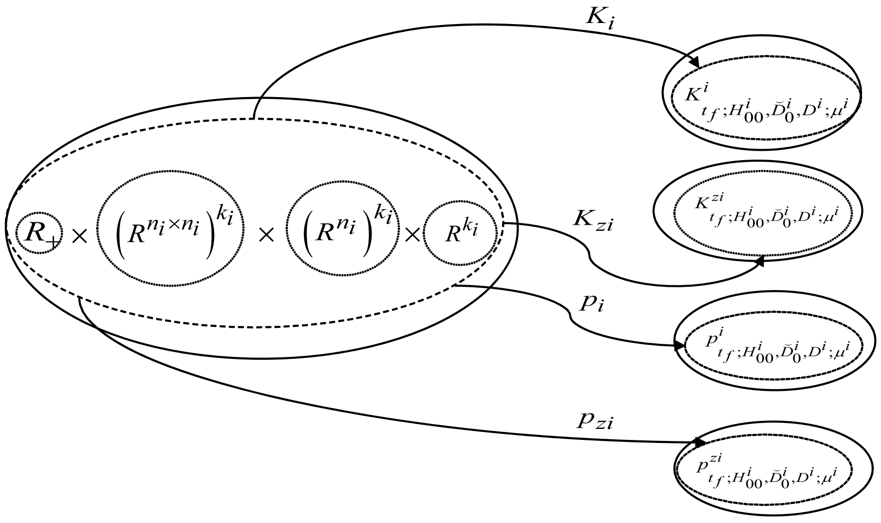
where all the parametric design measures μ_r^i considered here by agent i, represent for different emphases on higher-order statistics and agent prioritization toward performance robustness. Solutions $\{Y_{00}^{i,r}(s, K_i, K_{zi}, p_i, p_{zi})\}_{r=1}^{k_i}$, $\{\breve{Z}_0^{i,r}(s, K_i, K_{zi}, p_i, p_{zi})\}_{r=1}^{k_i}$ and $\{Z^{i,r}(s, K_i, K_{zi}, p_i, p_{zi})\}_{r=1}^{k_i}$ when evaluated at $s = t_0$ satisfy the time-backward differential equations (3.32)–(3.57) together with the terminal-value conditions as aforementioned.

From the above definition, the statistical problem is shown to be an initial cost problem, in contrast with the more traditional terminal cost class of investigations. One may address an initial cost problem by introducing changes of variables which convert it to a terminal cost problem. However, this modifies the natural context of statistical control, which it is preferable to retain. Instead, one may take a more direct dynamic programming approach to the initial cost problem. Such an approach is illustrative of the more general concept of the principle of optimality, an idea tracing its roots back to the 17th century and depicted in Fig. 3.5.



$x(t)$
$V(t+\Delta t, x(t+\Delta t))$
t_0
t
$t+\Delta t$
t_f
$V(s-\Delta s, H_{00}^i(s-\Delta s), \breve{D}_0^i(s-\Delta s), D^i(s-\Delta s))$
$(H_{00}^i(s), \breve{D}_0^i(s), D^i(s))$
t_0
$s-\Delta s$
s
t_f

Fig. 3.5 Value functions in the terminal cost problem (*top*) and the initial cost problem (*bottom*)



K_i
$K^i_{t_f; H_{00}^i, \breve{D}_0^i, D^i; \mu^i}$
$R_+ \times (R^{n_i \times n_i})^{k_i} \times (R^{n_i})^{k_i} \times R^{k_i}$
K_{zi}
$K^{zi}_{t_f; H_{00}^i, \breve{D}_0^i, D^i; \mu^i}$
p_i
$P^i_{t_f; H_{00}^i, \breve{D}_0^i, D^i; \mu^i}$
p_{zi}
$P^{zi}_{t_f; H_{00}^i, \breve{D}_0^i, D^i; \mu^i}$

Fig. 3.6 Admissible feedback gains and interaction recovery inputs

For the given $(t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f))$, classes of $\mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i}$, $\mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i}$, $\mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i}$ and $\mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i}$ of admissible 4-tuple $(K_i, K_{zi}, p_i, p_{zi})$ are then defined.

Definition 2 (Admissible feedback gains and affine inputs) For agent i, the compact subsets $\overline{K}_i \subset \mathbb{R}^{n_i \times n_i}$, $\overline{K}_{zi}$, $\overline{P}_i \subset \mathbb{R}^{m_i}$, and $\overline{P}_{zi} \subset \mathbb{R}^{m_{zi}}$ be denoted by the sets of allowable matrices and vectors as in Fig. 3.6.

Then, for $k_i \in \mathbb{Z}^+$, $\mu^i = \{\mu_r^i \geq 0\}_{r=1}^{k_i}$ with $\mu_1^i > 0$, the matrix-valued sets of $\mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i} \in C(t_0, t_f; \mathbb{R}^{m_i \times n_i})$ and $\mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i} \in C(t_0, t_f; \mathbb{R}^{m_{zi} \times n_i})$ and the vector-valued sets of $\mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f); \mu^i} \in C(t_0, t_f;$

$\mathbb{R}^{m_i}$) and $\mathcal{P}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i} \in C(t_0, t_f; \mathbb{R}^{m_{zi}})$ with respective values $K_i(\cdot) \in \overline{K}_i$, $K_{zi}(\cdot) \in \overline{K}_{zi}$, $p_i(\cdot) \in \overline{P}_i$, and $p_{zi}(\cdot) \in \overline{P}_{zi}$ are admissible if the resulting solutions to the time-backward differential equations (3.32)–(3.57) exist on the finite horizon $[t_0, t_f]$.

The optimization problem for agent i, where instantiations are aimed at reducing performance robustness and constituent strategies are robust to uncertain environment's stochastic variations, is subsequently stated.

Definition 3 (Optimization of Mayer problem) Suppose that $k_i \in \mathbb{Z}^+$ and the sequence $\mu^i = \{\mu^i_r \geq 0\}_{r=1}^{k_i}$ with $\mu^i_1 > 0$ are fixed. Then, the optimization problem over $[t_0, t_f]$ is given by the minimization of agent i's performance index (3.71) over all $K_i(\cdot) \in \mathcal{K}^i_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, $K_{zi}(\cdot) \in \mathcal{K}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, $p_i(\cdot) \in \mathcal{P}^i_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$ and $p_{zi}(\cdot) \in \mathcal{P}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$ subject to the time-backward dynamical constraints (3.58)–(3.70) for $s \in [t_0, t_f]$.

It is important to recognize that the optimization considered here is in "Mayer form" and can be solved by applying an adaptation of the Mayer form verification theorem of dynamic programming given in [4]. In the framework of dynamic programming where the subject optimization is embedded into a family of optimization based on different starting points, there is therefore a need to parameterize the terminal time and new states by $(\varepsilon, Y^i_{00}, \breve{Z}^i_0, Z^i)$ rather than $(t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f))$. Thus, the values of the corresponding optimization problems depend on the terminal-value conditions which lead to the definition of a value function.

Definition 4 (Value function) The value function $\mathcal{V}^i : [t_0, t_f] \times (\mathbb{R}^{n_i \times n_i})^{k_i} \times (\mathbb{R}^{n_i})^{k_i} \times \mathbb{R}^{k_i} \mapsto \mathbb{R}^+$ associated with the Mayer problem defined as $\mathcal{V}^i(\varepsilon, Y^i_{00}, \breve{Z}^i_0, Z^i)$ is the minimization of $\phi^i_0(t_0, Y^i_{00}(\cdot, K_i, K_{zi}, p_i, p_{zi}), \breve{Z}^i_0(\cdot, K_i, K_{zi}, p_i, p_{zi}), Z^i(\cdot, K_i, K_{zi}, p_i, p_{zi}))$ over all $K_i(\cdot) \in \mathcal{K}^i_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, $K_{zi}(\cdot) \in \mathcal{K}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, $p_i(\cdot) \in \mathcal{P}^i_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, and $p_{zi}(\cdot) \in \mathcal{P}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$ subject to some $(\varepsilon, Y^i_{00}, \breve{Z}^i_0, Z^i) \in [t_0, t_f] \times (\mathbb{R}^{n_i \times n_i})^{k_i} \times (\mathbb{R}^{n_i})^{k_i} \times \mathbb{R}^{k_i}$.

It is conventional to let $\mathcal{V}^i(\varepsilon, Y^i_{00}, \breve{Z}^i_0, Z^i) = \infty$ when $\mathcal{K}^i_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, $\mathcal{K}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$, $\mathcal{P}^i_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$ and $\mathcal{P}^{zi}_{t_f, Y^i_{00}(t_f), \breve{Z}^i_0(t_f), Z^i(t_f); \mu^i}$ are empty. To avoid cumbersome notation, the dependence of trajectory solutions on K_i, K_{zi}, p_i and p_{zi} is now suppressed. Next, some candidates for the value function can be constructed with the help of a reachable set.

Definition 5 (Reachable set) At agent i, let the reachable set $\mathcal{Q}^i$ be defined as follows

$$\mathcal{Q}^i \triangleq \Big\{ \big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \in [t_0, t_f] \times \big(\mathbb{R}^{n_i \times n_i}\big)^{k_i} \times \big(\mathbb{R}^{n_i}\big)^{k_i} \times \mathbb{R}^{k_i} \text{ such that}$$
$$\mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times$$
$$\mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \text{ not empty} \Big\}$$

Moreover, it can be shown that the value function is satisfying a partial differential equation at each interior point of $\mathcal{Q}^i$ at which it is differentiable.

Theorem 4 (Hamilton–Jacobi–Bellman (HJB) equation for Mayer problem) *For agent i, let $(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ be any interior point of the reachable set $\mathcal{Q}^i$ at which the value function $\mathcal{V}^i(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ is differentiable. If there exists an optimal 4-tuple control strategy $(K_i^*, K_{zi}^*, p_i^*, p_{zi}^*) \in \mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i}$, the partial differential equation associated with agent i*

$$\begin{aligned}
0 = \min_{K_i \in \overline{K}_i, K_{zi} \in \overline{K}_{zi}, p_i \in \overline{P}_i, p_{zi} \in \overline{P}_{zi}} \Big\{ & \frac{\partial}{\partial \varepsilon} \mathcal{V}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \\
& + \frac{\partial}{\partial \operatorname{vec}(Y_{00}^i)} \mathcal{V}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \cdot \operatorname{vec}\big(F_{00}^i\big(\varepsilon; Y_{00}^i, Y_{01}^i, Y_{20}^i; K_i, K_{zi}\big)\big) \\
& + \frac{\partial}{\partial \operatorname{vec}(\breve{Z}_0^i)} \mathcal{V}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \cdot \operatorname{vec}\big(\breve{G}_0^i\big(\varepsilon; Y_{00}^i, Y_{02}^i, \breve{Z}_0^i; K_i, K_{zi}; p_i, p_{zi}\big)\big) \\
& + \frac{\partial}{\partial \operatorname{vec}(Z^i)} \mathcal{V}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \\
& \cdot \operatorname{vec}\big(G^i\big(\varepsilon; Y_{00}^i, Y_{00}^i, Y_{01}^i, Y_{02}^i, Y_{10}^i, Y_{11}^i, Y_{12}^i, Y_{20}^i, Y_{21}^i, Y_{22}^i, \breve{Z}_0^i, \breve{Z}_2^i; p_i, p_{zi}\big)\big) \Big\}
\end{aligned} \tag{3.72}$$

is satisfied together with $\mathcal{V}^i(t_0, Y_{00}^i, \breve{Z}_0^i, Z^i) = \phi_0^i(t_0, Y_{00}^i, \breve{Z}_0^i, Z^i)$ and vec$(\cdot)$ the vectorizing operator of enclosed entities. The optimum in (3.72) is achieved by the 4-tuple control decision strategy $(K_i^*(\varepsilon), K_{zi}^*(\varepsilon), p_i^*(\varepsilon), p_{zi}^*(\varepsilon))$ of the optimal decision strategy at ε.

Proof Detail discussions are in [4] and the modification of the original proofs adapted to the framework of statistical control together with other formal analysis can be found in [7] which is available via http://etd.nd.edu/ETD-db/theses/available/etd-04152004-121926/unrestricted/PhamKD052004.pdf. □

Finally, the next theorem gives the sufficient condition used to verify optimal decisions for the interconnected system or agent i.

Theorem 5 (Verification theorem) *Fix $k_i \in \mathbb{Z}^+$. Let $\mathcal{W}^i(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ be a continuously differentiable solution of the HJB equation (3.72) and satisfy the boundary condition*

$$\mathcal{W}^i\big(t_0, Y_{00}^i, \breve{Z}_0^i, Z^i\big) = \phi_0^i\big(t_0, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \tag{3.73}$$

Let $(t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f))$ be a point of $\mathcal{Q}^i$; 4-tuple $(K_i, K_{zi}, p_i, p_{zi})$ in $\mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i}$; and Y_{00}^i, $\breve{Z}_0^i$, and Z^i the corresponding solutions of (3.58)–(3.70). Then, $\mathcal{W}^i(s, Y_{00}^i(s), \breve{Z}_0^i(s), Z^i(s))$ is time-backward increasing in s. If $(K_i^*, K_{zi}^*, p_i^*, p_{zi}^*)$ is a 4-tuple in $\mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i}$ defined on $[t_0, t_f]$ with corresponding solutions, Y_{00}^{i*}, $\breve{Z}_0^{i*}$ and Z^{i*} of the dynamical equations such that

$$\begin{aligned}
0 &= \frac{\partial}{\partial \varepsilon}\mathcal{W}^i\big(s, Y_{00}^{i*}(s), \breve{Z}_0^{i*}(s), Z^{i*}(s)\big) + \frac{\partial}{\partial\,\mathrm{vec}(Y_{00}^i)}\mathcal{W}^i\big(s, Y_{00}^{i*}(s), \breve{Z}_0^{i*}(s), Z^{i*}(s)\big)\\
&\quad \cdot \mathrm{vec}\big(F_{00}^i\big(s; Y_{00}^{i*}(s), Y_{01}^{i*}(s), Y_{20}^{i*}(s); K_i^*(s), K_{zi}^*(s)\big)\big)\\
&\quad + \frac{\partial}{\partial\,\mathrm{vec}(\breve{Z}_0^i)}\mathcal{W}^i\big(s, Y_{00}^{i*}(s), \breve{Z}_0^{i*}(s), Z^{i*}(s)\big)\\
&\quad \cdot \mathrm{vec}\big(\breve{G}_0^i\big(s; Y_{00}^{i*}(s), Y_{02}^{i*}(s), \breve{Z}_0^{i*}(s); K_i^*(s), K_{zi}^*(s); p_i^*(s), p_{zi}^*(s)\big)\big)\\
&\quad + \frac{\partial}{\partial\,\mathrm{vec}(Z^i)}\mathcal{W}^i\big(s, Y_{00}^{i*}(s), \breve{Z}_0^{i*}(s), Z^{i*}(s)\big)\\
&\quad \cdot \mathrm{vec}\big(G^i\big(\varepsilon; Y_{00}^{i*}(s), Y_{00}^{i*}(s), Y_{01}^{i*}(s), Y_{02}^{i*}(s), Y_{10}^{i*}(s), Y_{11}^{i*}(s), Y_{12}^{i*}(s),\\
&\qquad Y_{20}^{i*}(s), Y_{21}^{i*}(s), Y_{22}^{i*}(s), \breve{Z}_0^{i*}(s), \breve{Z}_2^{i*}(s); p_i^*(s), p_{zi}^*(s)\big)\big)
\end{aligned} \tag{3.74}$$

then $(K_i^*, K_{zi}^*, p_i^*, p_{zi}^*)$ in $\mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i}$ is optimal and

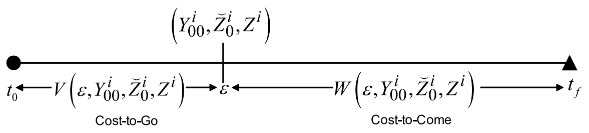
$$\mathcal{W}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) = \mathcal{V}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \tag{3.75}$$

where $\mathcal{V}^i(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ is the value function.

Proof Due to the length limitation, the interested reader is referred to the work by the author [7] which can be found via http://etd.nd.edu/ETD-db/theses/available/etd-04152004-121926/unrestricted/PhamKD052004.pdf for the detail proof and other relevant analysis. □

Note that, to have a solution $(K_i^*, K_{zi}^*, p_i^*, p_{zi}^*) \in \mathcal{K}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{K}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^i_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i} \times \mathcal{P}^{zi}_{t_f, Y_{00}^i(t_f), \breve{Z}_0^i(t_f), Z^i(t_f);\mu^i}$ well

Fig. 3.7 Cost-to-go in the sufficient condition to Hamilton–Jacobi–Bellman equation


$\left(Y_{00}^i, \breve{Z}_0^i, Z^i\right)$
t_0
$V\left(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\right)$
ε
$W\left(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\right)$
t_f
Cost-to-Go
Cost-to-Come

defined and continuous for all $s \in [t_0, t_f]$, the trajectory solutions $Y_{00}^i(s)$, $\breve{Z}_0^i(s)$ and $Z^i(s)$ to the dynamical equations (3.58)–(3.70) when evaluated at $s = t_0$ must then exist. Therefore, it is necessary that $Y_{00}^i(s)$, $\breve{Z}_0^i(s)$ and $Z^i(s)$ are finite for all $s \in [t_0, t_f)$. Moreover, the solutions of the dynamical equations (3.58)–(3.70) exist and are continuously differentiable in a neighborhood of t_f. Applying the results from [2], these trajectory solutions can further be extended to the left of t_f as long as $Y_{00}^i(s)$, $\breve{Z}_0^i(s)$ and $Z^i(s)$ remain finite. Hence, the existence of unique and continuously differentiable solution $Y_{00}^i(s)$, $\breve{Z}_0^i(s)$ and $Z^i(s)$ are bounded for all $s \in [t_0, t_f)$. As a result, the candidate value function $\mathcal{W}^i(s, Y_{00}^i, \breve{Z}_0^i, Z^i)$ is continuously differentiable as well.

3.5 Distributed Risk-Averse Feedback Stabilization

Note that the optimization problem is in "Mayer form" and can be solved by applying an adaptation of the Mayer form verification theorem of dynamic programming as described in the previous section. In particular, the terminal time and states of a family of optimization problems are now parameterized as $(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ rather than $(t_f, H_{00}^i(t_f), \breve{D}_0^i(t_f), D^i(t_f))$. Precisely stated, for $\varepsilon \in [t_0, t_f]$ and $1 \le r \le k_i$, the states of the performance robustness (3.58)–(3.70) defined on the interval $[t_0, \varepsilon]$ have the terminal values denoted by $H_{00}^i(\varepsilon) \equiv Y_{00}^i$, $\breve{D}_0^i(\varepsilon) \equiv \breve{Z}_0^i$ and $D^i(\varepsilon) \equiv Z^i$. Figure 3.7 further illustrates the cost-to-go and cost-to-come functions in statistical control.

Since the performance index (3.71) is quadratic affine in terms of arbitrarily fixed x_i^0, this observation suggests a candidate solution to the HJB equation (3.72) may be of the form as follows. For instance, it is assumed that $(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ is an interior point of the reachable set $\mathcal{Q}^i$ at which the real-valued function

$$\begin{aligned}\mathcal{W}^i\left(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\right) = \left(x_i^0\right)^T \sum_{r=1}^{k_i} \mu_r^i \left(Y_{00}^{i,r} + E_r^i(\varepsilon)\right) x_i^0 \\ + 2\left(x_i^0\right)^T \sum_{r=1}^{k_i} \mu_r^i \left(\breve{Z}_0^{i,r} + \breve{T}_r^i(\varepsilon)\right) \\ + \sum_{r=1}^{k_i} \mu_r^i \left(Z^{i,r} + T_r^i(\varepsilon)\right) \qquad (3.76)\end{aligned}$$

is differentiable. The parametric functions of time $E_r^i \in C^1(t_0, t_f; \mathbb{R}^{n_i \times n_i})$, $\breve{T}_r^i \in C^1(t_0, t_f; \mathbb{R}^{n_i})$ and $T_r^i \in C^1(t_0, t_f; \mathbb{R})$ are yet to be determined. Furthermore, the time derivative of $\mathcal{W}^i(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i)$ can be shown as below

$$\begin{aligned}
&\frac{d}{d\varepsilon}\mathcal{W}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \\
&\quad = \big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \Big[F_{00}^{i,r}\big(\varepsilon; Y_{00}^i, Y_{01}^i, Y_{20}^i; K_i, K_{zi}\big) + \frac{d}{d\varepsilon} E_r^i(\varepsilon) \Big] x_i^0 \\
&\qquad + 2\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \Big[\breve{G}_0^{i,r}(\varepsilon; Y_{00}^i, Y_{02}^i, \breve{Z}_0^i; K_i, K_{zi}; p_i, p_{zi}) + \frac{d}{d\varepsilon} \breve{T}_r^i(\varepsilon) \Big] \\
&\qquad + \sum_{r=1}^{k_i} \mu_r^i \Big[G^{i,r}\big(\varepsilon; Y_{00}^i, Y_{00}^i, Y_{01}^i, Y_{02}^i, Y_{10}^i, Y_{11}^i, Y_{12}^i, Y_{20}^i, Y_{21}^i, Y_{22}^i, \breve{Z}_0^i, \breve{Z}_2^i; \\
&\qquad\qquad\qquad p_i, p_{zi}\big) \\
&\qquad + \frac{d}{d\varepsilon} T_r^i(\varepsilon) \Big]
\end{aligned} \tag{3.77}$$

The substitution of this hypothesized solution (3.76) into the HJB equation (3.72) and making use of (3.77) results in

$$\begin{aligned}
0 \equiv \min_{K_i \in \overline{K}_i, K_{zi} \in \overline{K}_{zi}, p_i \in \overline{P}_i, p_{zi} \in \overline{P}_{zi}} &\Bigg\{ \big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \frac{d}{d\varepsilon} E_r^i(\varepsilon) x_i^0 + \sum_{r=1}^{k_i} \mu_r^i \frac{d}{d\varepsilon} T_r^i(\varepsilon) \\
&+ 2\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \frac{d}{d\varepsilon} \breve{T}_r^i(\varepsilon) + \big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i F_{00}^{i,r}\big(\varepsilon; Y_{00}^i, Y_{01}^i, Y_{20}^i; K_i, K_{zi}\big) x_i^0 \\
&+ 2\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \breve{G}_0^{i,r}\big(\varepsilon; Y_{00}^i, Y_{02}^i, \breve{Z}_0^i; K_i, K_{zi}; p_i, p_{zi}\big) \\
&+ \sum_{r=1}^{k_i} \mu_r^i G^{i,r}\big(\varepsilon; Y_{00}^i, Y_{00}^i, \ldots, Y_{01}^i, Y_{02}^i, Y_{10}^i, Y_{11}^i, Y_{12}^i, Y_{20}^i, Y_{21}^i, Y_{22}^i, \breve{Z}_0^i, \breve{Z}_2^i; \\
&\qquad p_i, p_{zi}\big) \Bigg\}
\end{aligned} \tag{3.78}$$

In the light of arbitrary x_i^0 with the rank of one, the differentiation of the expression within the bracket of (3.78) with respect to K_i, K_{zi}, p_i, and p_{zi} yields the necessary conditions for an extremum of (3.71) on $[t_0, \varepsilon]$

$$K_i = -\big(\mu_1^i R_i(\varepsilon)\big)^{-1} B_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i Y_{00}^{i,r} \tag{3.79}$$

$$K_{zi} = -\big(\mu_1^i R_{zi}(\varepsilon)\big)^{-1} C_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i Y_{00}^{i,r} \tag{3.80}$$

$$p_i = -\big(\mu_1^i R_i(\varepsilon)\big)^{-1} B_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i \breve{Z}_0^{i,r} \tag{3.81}$$

$$p_{zi} = -\big(\mu_1^i R_{zi}(\varepsilon)\big)^{-1} C_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i \breve{Z}_0^{i,r} \tag{3.82}$$

Replacing these results (3.79)–(3.82) into the right member of the HJB equation (3.72) yields the value of the minimum. It is now necessary to exhibit $\{E_r^i(\cdot)\}_{r=1}^{k_i}$, $\{\breve{T}_r^i(\cdot)\}_{r=1}^{k_i}$, and $\{T_r^i(\cdot)\}_{r=1}^{k_i}$ which render the left side of the HJB equation (3.72) equal to zero for $\varepsilon \in [t_0, t_f]$, when $\{Y_{00}^{i,r}\}_{r=1}^{k_i}$, $\{\breve{Z}_0^{i,r}\}_{r=1}^{k_i}$, and $\{Z^{i,r}\}_{r=1}^{k_i}$ are evaluated along the solution trajectories of the dynamical equations (3.58) through (3.70). Due to the space limitation, the closed forms for $\{E_r^i(\cdot)\}_{r=1}^{k_i}$, $\{\breve{T}_r^i(\cdot)\}_{r=1}^{k_i}$, and $\{T_r^i(\cdot)\}_{r=1}^{k_i}$ are however, omitted here. Furthermore, the boundary condition (3.73) requires that

$$\begin{aligned}
&\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \big(Y_{00}^{i,r}(t_0) + E_r^i(t_0)\big) x_i^0 + 2\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \big(\breve{Z}_0^{i,r}(t_0) + \breve{T}_r^i(t_0)\big) \\
&\quad + \sum_{r=1}^{k_i} \mu_r^i \big(Z^{i,r}(t_0) + T_r^i(t_0)\big) \\
&= \big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i Y_{00}^i(t_0)\, x_i^0 + 2\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \breve{Z}_0^{i,r}(t_0) + \sum_{r=1}^{k_i} \mu_r^i Z^{i,r}(t_0)
\end{aligned}$$

Thus, matching the boundary condition yields the initial value conditions $E_r^i(t_0) = 0$, $\breve{T}_r^i(t_0) = 0$ and $T_r^i(t_0) = 0$.

Applying the 4-tuple decision strategy specified in (3.79)–(3.82) along the solution trajectories of the time-backward differential equations (3.58)–(3.70), these equations become the time-backward Riccati-type differential equations. Enforcing the initial value conditions of $E_r^i(t_0) = 0$, $\breve{T}_r^i(t_0) = 0$ and $T_r^i(t_0) = 0$ uniquely implies that $E_r^i(\varepsilon) = Y_{00}^{i,r}(t_0) - Y_{00}^{i,r}(\varepsilon)$, $\breve{T}_r^i(\varepsilon) = \breve{Z}_0^{i,r}(t_0) - \breve{Z}_0^{i,r}(\varepsilon)$, and $T_r^i(\varepsilon) = Z^{i,r}(t_0) - Z^{i,r}(\varepsilon)$ for all $\varepsilon \in [t_0, t_f]$ and yields a value function

$$\begin{aligned}
&\mathcal{W}^i\big(\varepsilon, Y_{00}^i, \breve{Z}_0^i, Z^i\big) \\
&\quad = \big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i Y_{00}^{i,r}(t_0)\, x_i^0 + 2\big(x_i^0\big)^T \sum_{r=1}^{k_i} \mu_r^i \breve{Z}_0^{i,r}(t_0) + \sum_{r=1}^{k_i} \mu_r^i Z^{i,r}(t_0)
\end{aligned}$$

for which the sufficient condition (3.74) of the verification theorem is satisfied. Therefore, the extremal risk-averse control laws (3.79)–(3.82) minimizing (3.71) become optimal

$$K_i^*(\varepsilon) = -\big(\mu_1 R_i(\varepsilon)\big)^{-1} B_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i Y_{00}^{i,r*}(\varepsilon) \tag{3.83}$$

$$K_{zi}^*(\varepsilon) = -\big(\mu_1 R_{zi}(\varepsilon)\big)^{-1} C_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i Y_{00}^{i,r*}(\varepsilon) \tag{3.84}$$

$$p_i^*(\varepsilon) = -\big(\mu_1 R_i(\varepsilon)\big)^{-1} B_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i \breve{Z}_0^{i,r*}(\varepsilon) \tag{3.85}$$

$$p_{zi}^*(\varepsilon) = -\big(\mu_1 R_{zi}(\varepsilon)\big)^{-1} C_i^T(\varepsilon) \sum_{r=1}^{k_i} \mu_r^i \breve{Z}_0^{i,r*}(\varepsilon) \tag{3.86}$$

In closing the chapter, it is beneficial to summarize the methodological characteristics of the performance robustness analysis and distributed feedback stabilization and reflect on them from a larger perspective, e.g., management control of a large-scale interconnected system.

Theorem 6 (Decentralized and risk-averse control decisions) *Let the pairs (A_i, B_i) and (A_i, C_i) be uniformly stabilizable and the pair (A_i, H_i) be uniformly detectable on $[t_0, t_f]$. The corresponding interconnected system controlled by agent i is then considered with the initial condition $x^*(t_0)$*

$$\begin{aligned}
&dx_i^*(t) = \big(A_i(t)x_i^*(t) + B_i(t)u_i^*(t) + C_i(t)z_i^*(t) + E_i(t)d_i(t)\big)\,dt + G_i(t)\,dw_i(t)\\
&dy_i^*(t) = H_i(t)x_i^*(t)\,dt + dv_i(t),\\
&\quad i \in \overline{N}
\end{aligned}$$

whose local interactions with the nearest neighbors are given by

$$z_i^*(t) = \sum_{j=1, j\neq i}^{N} L_{ij}(t)x_{ij}^*(t), \quad i \in \overline{N}$$

which is then approximated by an explicit model-following of the type

$$dz_{mi}(t) = \big(A_{zi}(t)z_{mi}(t) + E_{zi}(t)d_{mi}(t)\big)\,dt + G_{zi}(t)\,dw_{mi}(t), \quad z_{mi}(t_0) = 0.$$

Then, the local state estimate $\hat{x}_i^*(t)$ for each agent i is generated by

$$\begin{aligned}
d\hat{x}_i^*(t) &= \big(A_i(t)x_i^*(t) + B_i(t)u_i^*(t) + C_i(t)z_i^*(t) + E_i(t)d_i(t)\big)\,dt\\
&\quad + L_i(t)\big(dy_i^*(t) - H_i(t)\hat{x}_i^*(t)\,dt\big), \quad \hat{x}_i^*(t_0) = x_i^0
\end{aligned}$$

and the Kalman-type gain is given by

$$L_i(t) = \Sigma_i(t) H_i^T(t) V_i^{-1}$$

The covariance for estimation error, $\Sigma_i(t)$ is the solution of the time-forward matrix differential equation

$$\frac{d}{dt}\Sigma_i(t) = A_i(t)\Sigma_i(t) + \Sigma_i(t)A_i(t) + G_i(t)W_iG_i^T(t) \\ - \Sigma_i(t)H_i^T(t)V_i^{-1}H_i(t)\Sigma_i(t), \quad \Sigma_i(t_0) = 0$$

Further, let $k_i \in \mathbb{Z}^+$ and the sequence $\mu^i = \{\mu_r^i \geq 0\}_{r=1}^{k_i}$ with $\mu_1^i > 0$. The optimal decentralized and risk-averse control decision for agent i is given by

$$u_i^*(t) = K_i^*(t)\hat{x}_i^*(t) + p_i^*(t); \qquad z_i^*(t) = K_{zi}^*(t)\hat{x}_i^*(t) + p_{zi}^*(t)$$

provided that

$$K_i^*(s) = -R_i^{-1}(s)B_i^T(s)\sum_{r=1}^{k_i}\hat{\mu}_r^i Y_{00}^{i,r*}(s) \tag{3.87}$$

$$K_{zi}^*(s) = -R_{zi}^{-1}(s)C_i^T(s)\sum_{r=1}^{k_i}\hat{\mu}_r^i Y_{00}^{i,r*}(s) \tag{3.88}$$

$$p_i^*(s) = -R_i^{-1}(s)B_i^T(s)\sum_{r=1}^{k_i}\hat{\mu}_r^i \breve{Z}_0^{i,r*}(s) \tag{3.89}$$

$$p_{zi}^*(s) = -R_{zi}^{-1}(s)C_i^T(s)\sum_{r=1}^{k_i}\hat{\mu}_r^i \breve{Z}_0^{i,r*}(s) \tag{3.90}$$

where normalized weighting preferences $\hat{\mu}_r^i \triangleq \mu_r^i/\mu_1^i$'s are chosen by agent i toward strategic design of its performance robustness. The supporting solutions $\{Y_{00}^{i,r*}(s)\}_{r=1}^{k_i}$, $\{\breve{Z}_0^{i,r*}(s)\}_{r=1}^{k_i}$, $\{Z^{i,r*}(s)\}_{r=1}^{k_i}$ optimally satisfy the time-backward matrix-valued and vector-valued differential equations

$$\frac{d}{ds}Y_{00}^{i*}(s) = F_{00}^{i*}\big(s, Y_{00}^{i*}(s), Y_{01}^{i*}(s), Y_{20}^{i*}(s); K_i^*(s), K_{zi}^*(s)\big), \quad Y_{00}^{i*}(t_f) \tag{3.91}$$

$$\frac{d}{ds}Y_{01}^{i*}(s) = F_{01}^{i*}\big(s, Y_{00}^{i*}(s), Y_{01}^{i*}, Y_{11}^{i*}(s), Y_{21}^{i*}(s); K_i^*(s), K_{zi}^*(s)\big), \quad Y_{01}^{i*}(t_f) \tag{3.92}$$

$$\frac{d}{ds}Y_{02}^{i*}(s) = F_{02}^{i*}\big(s, Y_{02}^{i*}, Y_{12}^{i*}(s), Y_{22}^{i*}(s); K_i^*(s), K_{zi}^*(s)\big), \quad Y_{02}^{i*}(t_f) \tag{3.93}$$

$$\frac{d}{ds}Y_{10}^{i*}(s) = F_{10}^{i*}\big(s, Y_{00}^{i*}(s), Y_{10}^{i*}(s), Y_{20}^{i*}(s); K_i^*(s), K_{zi}^*(s)\big), \quad Y_{10}^{i*}(t_f) \tag{3.94}$$

$$\frac{d}{ds}Y_{11}^{i*}(s) = F_{11}^{i*}\big(s, Y_{01}^{i*}(s), Y_{10}^{i*}(s), Y_{11}^{i*}(s), Y_{21}^{i*}(s)\big), \quad Y_{11}^{i*}(t_f) \tag{3.95}$$

$$\frac{d}{ds}Y_{12}^{i*}(s) = F_{12}^{i*}\big(s, Y_{02}^{i*}(s), Y_{12}^{i*}(s), Y_{22}^{i*}(s)\big), \quad Y_{12}^{i*}(t_f) \tag{3.96}$$

$$\frac{d}{ds}Y_{20}^{i*}(s) = F_{20}^{i*}\big(s, Y_{00}^{i*}(s), Y_{10}^{i*}(s), Y_{20}^{i*}(s); K_i^*(s), K_{zi}^*(s)\big), \quad Y_{20}^{i*}(t_f) \tag{3.97}$$

$$\frac{d}{ds}Y_{21}^{i*}(s) = F_{21}^{i*}\big(s, Y_{01}^{i}(s), Y_{11}^{i}(s), Y_{20}^{i}(s), Y_{21}^{i}(s)\big), \quad Y_{21}^{i*}(t_f) \tag{3.98}$$

$$\frac{d}{ds}Y_{22}^{i*}(s) = F_{22}^{i*}\big(s, Y_{02}^{i*}(s), Y_{12}^{i*}(s), Y_{22}^{i*}(s)\big), \quad Y_{22}^{i*}(t_f) \tag{3.99}$$

$$\frac{d}{ds}\breve{Z}_0^{i*}(s) = \breve{G}_0^{i*}\big(s, Y_{00}^{i*}(s), Y_{02}^{i*}(s), \breve{Z}_0^{i*}(s); K_i^*(s), K_{zi}^*(s); \ldots p_i^*(s), p_{zi}^*(s)\big),$$
$$\breve{Z}_0^{i*}(t_f) \tag{3.100}$$

$$\frac{d}{ds}\breve{Z}_1^{i*}(s) = \breve{G}_1^{i*}\big(s, Y_{10}^{i*}(s), Y_{12}^{i*}(s), \breve{Z}_0^{i*}(s), \breve{Z}_1^{i*}(s); p_i^*(s), p_{zi}^*(s)\big),$$
$$\breve{Z}_1^{i*}(t_f) \tag{3.101}$$

$$\frac{d}{ds}\breve{Z}_2^{i*}(s) = \breve{G}_2^{i*}\big(s, Y_{20}^{i*}, Y_{22}^{i*}(s), \breve{Z}_2^{i*}(s); p_i^*(s), p_{zi}^*(s)\big), \quad \breve{Z}_2^{i*}(t_f) \tag{3.102}$$

$$\begin{aligned}\frac{d}{ds}Z^{i*}(s) = G^{i*}\big(&s, Y_{00}^{i*}(s), Y_{01}^{i*}(s), Y_{02}^{i*}(s), Y_{10}^{i*}(s), Y_{11}^{i*}(s), Y_{12}^{i*}(s), Y_{20}^{i*}(s),\\ &Y_{21}^{i*}(s), Y_{22}^{i*}(s), \breve{Z}_0^{i*}(s), \breve{Z}_2^{i*}(s); p_i^*(s), p_{zi}^*(s)\big),\end{aligned}$$
$$Z^{i*}(t_f) \tag{3.103}$$

where the terminal-value conditions are defined by

$$\begin{aligned}
&Y_{00}^{i*}(t_f) \triangleq Q_i^f \times 0 \times \cdots \times 0 && Y_{01}^{i*}(t_f) \triangleq Q_i^f \times 0 \times \cdots \times 0\\
&Y_{02}^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0 && Y_{10}^{i*}(t_f) \triangleq Q_i^f \times 0 \times \cdots \times 0\\
&Y_{11}^{i*}(t_f) \triangleq Q_i^f \times 0 \times \cdots \times 0 && Y_{12}^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0\\
&Y_{20}^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0 && Y_{21}^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0\\
&Y_{22}^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0 && \breve{Z}_0^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0\\
&\breve{Z}_1^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0 && \breve{Z}_2^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0\\
&Z^{i*}(t_f) \triangleq 0 \times 0 \times \cdots \times 0
\end{aligned}$$

As pictorial illustrations, Fig. 3.8 shows how local interaction predictions between agent i and its neighbor agents as well as local perceptions of performance risk associated with the Chi-squared performance random variable are integrated

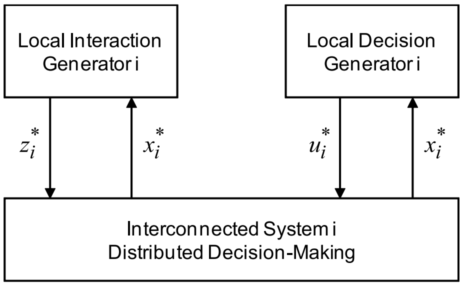

Local Interaction Generator i
Local Decision Generator i
z_i^*
x_i^*
u_i^*
x_i^*
Interconnected System i Distributed Decision-Making

Fig. 3.8 Local control and interaction coordination for interconnected autonomous systems

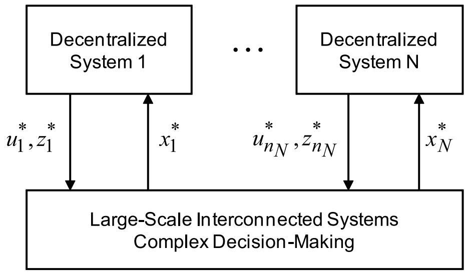


Fig. 3.9 Multi-level control and coordination for large-scale interconnected systems

to yield a class of distributed feedback decision strategies. Such decentralized and risk-averse decision laws shown in Fig. 3.8, are therefore robust and adaptable to uncertain environments without reliance on intensive communication exchanges. Consequently, these interconnected systems controlled by autonomous agent i for $i = 1, \ldots N$ take active roles to mitigate performance variations caused by the underlying stochastic disturbances via means of mixed random realizations and thus the large-scale interconnected system is capable of shaping the performance distribution and robustness whose multi-level control and coordination among interconnected autonomous systems is illustrated in Fig. 3.9.

3.6 Conclusions

The chapter presents a new innovative solution concept to analytically address performance robustness which is widely recognized as a pressing need in stochastic control of large-scale interconnected systems. In the most basic framework of performance-information analysis, a local performance-information system transmits messages about higher-order characteristics of performance uncertainty to the local decision maker for his use in future adaptive risk-averse decisions. The messages of performance-measure statistics transmitted are then influenced by the attributes of decentralized decision settings and local decision makers. The risk of a performance measure expressed as a linear combination of higher-order aspects (beyond statistical averaging) of performance distribution in distributed control, not only works as feedback information for future risk-averse decisions via local information about the states of nature but also serves as an influence mechanism for the local decision maker's current control behavior. This work contributes toward the construction of a theory of performance robustness for stochastic control of a class of large-scale systems in terms of both substantial results and research methodology.

References

1. Brockett, R.W.: Finite Dimensional Linear Systems. Wiley, New York (1970)
2. Dieudonne, J.: Foundations of Modern Analysis. Academic Press, New York (1960)
3. Field, R.V., Bergman, L.A.: Reliability based approach to linear covariance control design. J. Eng. Mech. **124**, 193–199 (1998)
4. Fleming, W.H., Rishel, R.W.: Deterministic and Stochastic Optimal Control. Springer, New York (1975)
5. Hansen, L.P., Sargent, T.J., Tallarini, T.D. Jr.: Robust permanent income and pricing. Rev. Econ. Stud. **66**, 873–907 (1999)
6. Liberty, S.R.: Performance analysis of stochastic linear control systems: a new viewpoint. In: Proceedings International Forum on Alternatives for Linear Multivariable Control, pp. 79–86 (1977)
7. Pham, K.D.: Statistical control paradigms for structural vibration suppression. In: PhD Dissertation (2004). Available via http://etd.nd.edu/ETD-db/theses/available/etd-04152004-121926/unrestricted/PhamKD052004.pdf. Cited 13 August 2009
8. Pham, K.D.: Cooperative solutions in multi-person quadratic decision problems: performance-measure statistics and cost-cumulant control paradigm. In: The 46th Conference on Decisions and Control, pp. 2484–2490 (2007)
9. Pham, K.D.: Non-cooperative outcomes for stochastic Nash games: decision strategies towards multi-attribute performance robustness. In: The 17th World Congress International Federation on Automatic Control, pp. 11750–11756 (2008)
10. Pham, K.D.: New results in stochastic cooperative games: strategic coordination for multi-resolution performance robustness. In: Hirsch, M.J., Pardalos, P.M., Murphey, R., Grundel, D. (eds.) Optimization and Cooperative Control Strategies. Lecture Notes in Control and Information Sciences, vol. 381, pp. 257–285. Springer, Berlin (2008)
11. Sain, M.K.: Control of linear systems according to the minimal variance criterion-a new approach to the disturbance problem. IEEE Trans. Autom. Control **11**, 118–122 (1966)
12. Whittle, P.: Risk Sensitive Optimal Control. Wiley, New York (1990)

Chapter 4
A General Approach for Modules Identification in Evolving Networks

Thang N. Dinh, Incheol Shin, Nhi K. Thai, My T. Thai, and Taieb Znati

Summary Most complex networks exhibit a network modular property that is nodes within a network module are more densely connected among each other than with the rest of the network. Identifying network modules can help deeply understand the structures and functions of a network as well as design a robust system with minimum costs. Although there are several algorithms identifying the modules in literature, none can adaptively update modules in evolving networks without recomputing the modules from scratch. In this chapter, we introduce a general approach to efficiently detect and trace the evolution of modules in an evolving network. Our solution can identify the modules of each network snapshot based on the modules of previous snapshots, thus dynamically updating these modules. Moreover, we also provide a network compact representation which significantly reduces the size of the network, thereby minimizing the running time of any existing algorithm on the modules identification.

The work of My T. Thai is supported by DTRA, Young Investigator Award, grant number HDTRA1-09-1-0061.

T.N. Dinh (✉) · I. Shin · M.T. Thai
University of Florida, Gainesville, FL 32611, USA
e-mail: tdinh@cise.ufl.edu

I. Shin
e-mail: ishin@cise.ufl.edu

M.T. Thai
e-mail: mythai@cise.ufl.edu

N.K. Thai
University of Minnesota, Minneapolis, MN 55455, USA
e-mail: thai0028@umn.edu

T. Znati
University of Pittsburgh, Pittsburgh, PA 15215, USA
e-mail: znati@cs.pitt.edu

4.1 Introduction

Network modules, also known as community structures in social networks, are usually defined as groups of nodes densely connected inside and sparser between groups. Modules in networks usually correspond to groups of nodes with the same function in networks. In the case of World Wide Web, for example, modules are groups of pages with a same topic. In biological networks, modules may be groups of proteins, genes with same functions. Studying network modules becomes very active and crucial in different scientific communities with an emerging of very large and complex networks such as phone call networks in sociology, metabolic, protein-protein-interaction, and gene-regulatory networks in biology, citation networks and the power grid networks in physics, and communication networks and World Wide Web in the technology field. Studying network modules is well-motivated as it is an essential tool to study and understand the complexity of many natural and artificial systems. It reveals the relationship between nodes in a same module as well as the relationship between modules in networks. It also helps understand the interplay between network topology and functionality. Knowing the behavior of modules with respect to the movements of nodes and links in evolving networks can directly help predict the interdependent responses of network components and help to improve the robustness of networks. Knowing the evolutionary of the structure provides the basics construction/design topology we need to achieve so that it will be less sensitive to the changes of structures.

Although the study of identifying network modules has been investigated extensively [1–14], including the dynamics and evolution of a network in the analysis of inherent modules is a new task that has not yet been well investigated. The study of such evolution requires computing modules of the network at different time instances. In previous approaches, [15–17] network modules are first identified separately at each time point and then modular structures at two consecutive time points are compared to each other in order to highlight evolution changes. However, identifying network modules in each state of the network from scratch may result in *prohibitive computational costs*, particularly in the case of highly dynamic and extremely large networks, such as the online social networks MySpace and Facebook with more than hundred millions of nodes. In addition, it may be *infeasible* in the case of *limited topological data*. Moreover, identifying network modules of each snapshot independently may lead to substantial and unexpected variation, thus introducing incorrect evolution phenomena.

To overcome the above limitations, we propose a graph-based approach using modular structure found in previous steps as a guide to incrementally identify modules in the next step. Adaptively updating modules in evolving networks not only avoids recomputing them from scratch for each time point but also produces consistent modular structures over time. For further reducing the running time and computational cost, we provide a compact network representation with much smaller size such that the modular structure of compact networks is as same as that of the original networks. The modular information is embedded in this compact representation to allow detection of modules at next time point and modules' evolving at the same time. Several experiments with real data sets show that our approach performs extremely well, especially in term of the running time. In some cases, it can be 1000 times faster than that of computing the modules on each network snapshot.

The rest of the chapter is organized as follows. Section 4.2 presents the preliminaries and the problem definition. In Sect. 4.3, we discuss the construction of compact network representation together with detailed analytical results. Section 4.4 analyzes how network structure affected when elemental changes such as adding, removing new nodes, links happen and presents an adaptively updating algorithm. Section 4.5 shows the experimental evaluation. Finally, the Conclusion and future work are discussed in Sect. 4.6.

4.2 Preliminaries and Problem Definition

4.2.1 Preliminaries

We study weighted undirected graph $G = (V, E)$ i.e., each link $(u, v) \in E$ is associated with a nonnegative number $w(u, v)$. The weights on links take greater values for node pairs that are closely related or similar in some way [18].

The weighted degree of a node v is defined as:

$$d(v) = \sum_{u} w(u, v)$$

The volume of a subset $S \subseteq V$ of nodes is defined as:

$$\text{vol}(S) = \sum_{v \in S} d(v)$$

We recall that $\text{vol}(V) = 2m$ where m is the total weights of links in G. We denote the total weight of links crossing two sets of nodes $X, Y \in V$ by

$$\Phi(X, Y) = \sum_{u \in X, v \in Y} w(u, v)$$

We note that $\text{vol}(S) = \Phi(S, V)$. For simplicity, we also write $\Phi(X)$ instead of $\Phi(X, X)$.

Recall that a *modular structure, sometimes referred as partition,* $\mathscr{C} = \{C_1, C_2, \ldots, C_k\}$ is a division of network into disjoint sets of nodes C_i. To quantify the strength of a partition, the most widely accepted quantitative measurement [19] is called *modularity* $\mathscr{Q}$ which is defined as follows:

$$\mathscr{Q}(\mathscr{C}) = \sum_{i=1}^{k} \left[\frac{\Phi(C_i)}{2m} - \left(\frac{\text{vol}(C_i)}{2m} \right)^2 \right] \tag{4.1}$$

If a node u belongs to the module C_i, we say it has the membership in C_i and define the membership function $mb(u) = i$.

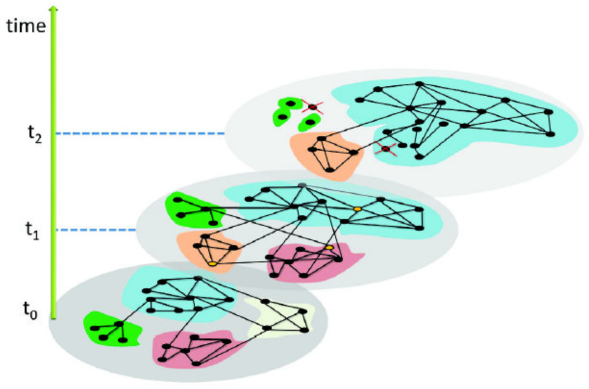


Fig. 4.1 In an evolving network, modules can split into smaller modules when connectivity inside modules weakened, or they can also merge together to form new modules

4.2.2 Problem Definition

We study *time dependent*, *weighted* and *undirected* networks. All algorithms and definitions can also be extended naturally for directed networks. In our network model, new nodes and links are frequently added into or removed from the network. See Fig. 4.1 for a pictorial representation. Assume that the network is $G^0(V^0, E^0)$ at time t_0 and it evolves into $G^i(V^i, E^i)$ at time t_i. Denote ΔV^i the set of added or removed nodes when the network evolves from G^{i-1} to G^i i.e., the symmetric difference $V^i \ominus V^{i-1}$. Similarly, we denote $\Delta E^i = E^i \ominus E^{i-1}$ the set of added or removed links when the network evolves from G^{i-1} to G^i. Given the network $G^{i-1}(V^{i-1}, E^{i-1})$ at time t_{i-1} and the changes $(\Delta V^i, \Delta E^i)$ in the network between t_{i-1} and t_i, one can deduce $G^i(V^i, E^i)$ from the formulas $V^i = V^{i-1} \ominus \Delta V^i$ and $E^i = E^{i-1} \ominus \Delta E^i$.

Problem Definition Given a network and its sequences of changes over time

$$G^0\big(V^0, E^0\big), \big(\Delta V^1, \Delta E^1\big), \ldots, \big(\Delta E^i, \Delta E^i\big), \ldots$$

Devise an online algorithm to efficiently

- Detect the modules in the network at everytime point without recomputing them from scratch, i.e., adaptively update modules overtime.
- Trace the evolution of network modules (identify the relationship between modules at different time points).

4.3 Compact Representation of a Network

Given a network $G = (V, E)$ and its modular structure $\mathscr{C} = \{C_1, C_2, \ldots, C_k\}$ where each disjoint subset of nodes C_i is called a module. Our goal is to construct a new network with equivalent structure to G but contains a significant smaller number of nodes and links. Such a network can be used in place of G to reduce the running time of identifying evolving modules and many existing modules identification algorithms in the current literature.

One way to construct such a compression of the network is to replace every module by a single node with a self-loop representing the connection inside the module [20, 21]. However, this approach might not be suitable for a general purpose as many modules identification algorithms will not work with the presence of self-loops.

We are inspired by an observation in biological networks that two proteins with identical set of neighbors belong to a same functional module. Hence, instead of introducing self-loops, we replace each module C_i by two new nodes x_i, y_i that share a same set of neighbors. The algorithm to construct the compact representation of G is presented in Algorithm 1. A link (x_i, y_i) with weight $\frac{\Phi(C_i)}{2}$ is added to represent the total weight of links inside module C_i, lines 8 and 9. The total weights of links crossing modules C_i and C_j will be distributed equally to each link $(x_i, x_j), (x_i, y_j), (y_i, x_j), (y_i, y_j)$, lines 13 to 18. This weight assignment will help preserve the modular structure. If a module contains only one single node, there will be no links inside the module. We replace that module by a single node in the compact representation to reduce the size of the compact representation as in line 6.

From the modularity optimization viewpoint, we will prove the structure equivalence of the original network and its compact representation. When we refer to the optimal partition, we mention the partition of network into modules that maximizes the modularity value.

Algorithm 1 Compact representation of a network

```
1: Input: $G(V, E)$, partition $\mathscr{C} = \{C_1, C_2, \ldots, C_k\}$.
2: Output: Compact representation $G'(V', E')$.
3: Initialize $G'(V', E') \leftarrow (\phi, \phi)$
4: for each module $C_i$ in $\mathscr{C}$ do
5:     if $|C_i| = 1$ then
6:         Create a new module $C'_i = \{x_i\}$ in the compact representation
7:     else
8:         Create a new module $C'_i = \{x_i, y_i\}$ in the compact representation
9:         $w(x_i; y_i) \leftarrow \frac{\Phi(C_i)}{2}$
10:    end if
11:    $V' \leftarrow V' \cup C'_i$
12: end for
13: for all $(u, v) \in E$ do
14:    $i \leftarrow mb(u),\ j \leftarrow mb(v)$
15:    $\forall (x, y) \in C'_i \times C'_j : w(x, y) \leftarrow \frac{\Phi(C_i, C_j)}{|C'_i||C'_j|}$
16: end for
17: return $G'(V', E')$
```

4.3.1 Structure Preservation

Lemma 4.3.1 *The contraction of $G(V, E)$ into $G'(V', E')$ preserves the modularity i.e. $\mathcal{Q}(\mathcal{C}' = \{C'_1, C'_2, \dots, C'_k\}) = \mathcal{Q}(\mathcal{C})$.*

Proof From the Definition (4.1):

$$\mathcal{Q}(\mathcal{C}') = \sum_i \left[\frac{\Phi(C'_i)}{2m'} - \left(\frac{\mathrm{vol}(C'_i)}{2m'} \right)^2 \right] \tag{4.2}$$

By the construction of G':

$$\Phi\big(C'_i\big) = \begin{cases} \Phi(\{x_i\}) = 0 = \Phi(C_i) & |C_i| < 2 \\ \Phi(x_i, y_i) = 2\frac{\Phi(C_i)}{2} = \Phi(C_i) & |C_i| \geq 2 \end{cases} \tag{4.3}$$

$$\begin{aligned} \mathrm{vol}\big(C'_i\big) &= \Phi\big(C'_i\big) + \sum_{j \neq i} \Phi\big(C'_i, C'_j\big) \\ &= \Phi(C_i) + \sum_{j \neq i} \sum_{u \in C'_i, v \in C'_j} w(u, v) \\ &= \Phi(C_i) + \sum_{j \neq i} |C'_i||C'_j| \frac{\Phi(C_i, C_j)}{|C'_i||C'_j|} \\ &= \mathrm{vol}(C_i) \end{aligned} \tag{4.4}$$

$$m' = \frac{1}{2} \sum_i \mathrm{vol}\big(C'_i\big) = \frac{1}{2} \sum_i \mathrm{vol}(C_i) = m \tag{4.5}$$

From (4.2), (4.3), (4.4), and (4.5) it follows that $\mathcal{Q}(\mathcal{C}') = \mathcal{Q}(\mathcal{C})$. □

Lemma 4.3.2 *Let $x_i, y_i \in C'_i$ obtained in construction of $G'(V', E')$ and $\mathcal{C}^* = \{C^*_1, C^*_2, \dots, C^*_k\}$ be an arbitrary partition of G' such that x_i, y_i belong to different modules in $\mathcal{C}^*$. There exists a partition of G' with modularity higher than $\mathcal{Q}(\mathcal{C}^*)$ in which x_i, y_i belong to a same module.*

Proof We show that either moving x_i to the module contains y_i or moving y_i to the module contains x_i will result in increasing of modularity (Fig. 4.2).

We first calculate the change in modularity if we move a group of nodes S from its module $C_a \supseteq S$ to a new module C_b ($C_a \cap C_b = \phi$):

$$\begin{aligned} \Delta\mathcal{Q}^S_{C_a, C_b} = \frac{1}{m} \big[\Phi(C_b, S) - \Phi(C_a \backslash S, S) \big] \\ - \frac{1}{2m^2} \mathrm{vol}(S) \big[\mathrm{vol}(C_b) - \mathrm{vol}(C_a \backslash S) \big] \end{aligned} \tag{4.6}$$

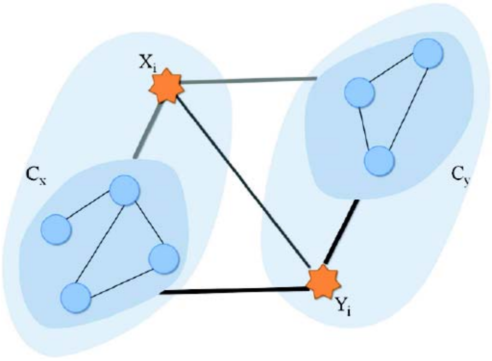


Fig. 4.2 If two nodes x_i, y_i are in different modules C_x, C_y, we can increase the overall modularity by either moving x_i to C_y or y_i to C_x

We will also use this formula in later proofs.

Assume that $x_i \in C_x$ and $y_i \in C_y$ where $C_x, C_y \in \mathscr{C}^*$.

Moving x_i from C_x to C_y: Using formula (4.6) with $C_a = C_x, C_b = C_y, S = \{x_i\}$ we have:

$$\begin{aligned}\Delta\mathscr{Q}^{\{x_i\}}_{C_x,C_y} &= 2\left[\frac{\Phi(\{x_i\},\{y_i\})}{2m} - \frac{\mathrm{vol}(\{x_i\})\mathrm{vol}(\{y_i\})}{4m^2}\right] \\ &\quad + \frac{1}{m}\left[\Phi\big(C_y\backslash\{y_i\},\{x_i\}\big) - \Phi\big(C_x\backslash\{x_i\},\{x_i\}\big)\right] \\ &\quad - \frac{1}{2m^2}\mathrm{vol}\big(\{x_i\}\big)\left[\mathrm{vol}\big(C_y\backslash\{y_i\}\big) - \mathrm{vol}\big(C_x\backslash\{x_i\}\big)\right] \qquad (4.7)\end{aligned}$$

We can similarly calculate the change in modularity moving y_i from C_y to C_x. Since x_i and y_i have same function in G' i.e., $\forall L \subseteq V\backslash\{x_i, y_i\} \rightarrow (\Phi(\{x_i\}, L) = \Phi(\{y_i\}, L)$. We have:

$$\begin{aligned}&\Delta\mathscr{Q}^{\{x_i\}}_{C_x,C_y} + \Delta\mathscr{Q}^{\{y_i\}}_{C_y,C_x} \\ &\quad = 4\left[\frac{\Phi(\{x_i\},\{y_i\})}{2m} - \frac{\mathrm{vol}(\{x_i\})\mathrm{vol}(\{y_i\})}{4m^2}\right] \\ &\quad = \frac{2\Phi(\{x_i, y_i\})}{2m} - \frac{\mathrm{vol}^2(\{x_i, y_i\})}{4m^2} \\ &\quad = \frac{2\Phi(C_i)}{2m} - \frac{\mathrm{vol}^2(C_i)}{4m^2} \\ &\quad > \frac{\Phi(C_i)}{2m} - \frac{\mathrm{vol}^2(C_i)}{4m^2} > 0 \quad \text{(this follows Lemma 4.3.3)} \qquad (4.8)\end{aligned}$$

Hence, either $\Delta\mathscr{Q}^{\{x_i\}}_{C_x,C_y} > 0$ or $\Delta\mathscr{Q}^{\{y_i\}}_{C_y,C_x} > 0$. □

The only left part is to prove that modularity of each module is nonnegative i.e., for a module C_i in a network

$$\mathscr{Q}(C_i) = \frac{\Phi(C_i)}{2m} - \left(\frac{\mathrm{vol}(C_i)}{2m}\right)^2 \geq 0 \tag{4.9}$$

The equality holds only when C_i contains only an isolated nodes or $C_i = V(G)$.

Lemma 4.3.3 *In the optimal partition of a network $G(V, E)$, there is no module with a negative modularity.*

Proof Assume that there exists a module C in the optimal partition $\mathscr{C}^* = \{C_1^*, C_2^*, \dots, C_k^*\}$ with $\mathscr{Q}(C) < 0$. Denote $\mathscr{N} = \{C' | \ C' \in \mathscr{C}^*, \Phi(C, C') > 0\}$ the set of modules adjacent to C. Merging C with any modules in $\mathscr{N}$ will not increase the modularity. Following (4.6), we have for all $C_i^* \in \mathscr{N}$:

$$\Delta\mathscr{Q}^C_{C,C'} = \frac{1}{m}\Phi\big(C', C\big) - \frac{1}{2m^2}\mathrm{vol}\big(C'\big)\mathrm{vol}(C) \leq 0$$

Take the sum over all $C' \in \mathscr{N}$, we obtain:

$$\begin{aligned}
&\sum_{C_i \in \mathscr{N}} \frac{1}{m}\Phi\big(C', C\big) - \sum_{C_i \in \mathscr{N}} \frac{1}{2m^2}\mathrm{vol}(C)\mathrm{vol}(C_i) \leq 0\\
\Leftrightarrow\quad & \frac{1}{m}\Phi\Bigg(\bigcup_{C' \in \mathscr{N}} (C', C)\Bigg) - \frac{1}{2m^2}\mathrm{vol}(C)\mathrm{vol}\Bigg(\bigcup_{C' \in \mathscr{N}} (C')\Bigg) \leq 0\\
\Rightarrow\quad & \frac{1}{m}\big(\mathrm{vol}(C) - \Phi(C, C)\big) - \frac{1}{2m^2}\mathrm{vol}(C)\mathrm{vol}(V\backslash C) \leq 0\\
\Rightarrow\quad & 2\mathscr{Q}(C) \geq 0
\end{aligned}$$

We obtain a contradiction. □

Theorem 4.3.1 *If partition $\mathscr{C}$ maximizes the modularity in $G(V, E)$, then $\mathscr{C}' = \{C_1', C_2', \dots, C_k'\}$ obtained in the construction of compact representation $G'(V', E')$ using Algorithm 1 will be the partition with maximum modularity in G'.*

Proof Let $\mathscr{C}^* = \{C_1^*, C_2^*, \dots, C_t^*\}$ be the partition with maximum modularity in G'. We will prove that $\mathscr{Q}(\mathscr{C}^*) = \mathscr{Q}(\mathscr{C}')$. As shown in Lemma 4.3.2, if $C_i = \{x_i, y_i\}$ and $x_i \in C_j^*$ then $y_i \in C_j^*$. Hence, we consider only partition of G' in which x_i, y_i are placed in a same module for all $|C_i'| = 2$. Thus, each $C_j^* \in \mathscr{C}^*$ can be decomposed into a union of modules in $\mathscr{C}'$. There exists a partition $I = \{I_1, I_2, \dots, I_t\}$ of $\{1, 2, \dots, k\}$ such that $C_j^* = \bigcup_{i \in I_j} C_i'$, $\forall j = 1 \dots t$.

Construct a partition $\mathscr{C}'' = \{C_1'', C_2'', \dots, C_t''\}$ of $G(V, E)$ corresponding to $\mathscr{C}^*$ of G' by assigning $C_j'' = \bigcup_{i \in I_j} C_i$, $\forall j = 1 \dots t$. Using similar proof showed in Lemma 4.3.1, we can prove that $\mathscr{Q}(\mathscr{C}'') = \mathscr{Q}(\mathscr{C}^*)$.

Since $\mathscr{Q}(\mathscr{C}') = \mathscr{Q}(\mathscr{C}) \geq \mathscr{Q}(\mathscr{C}'')$ (Lemma 4.3.1) and $\mathscr{Q}(\mathscr{C}'') = \mathscr{Q}(\mathscr{C}^*) \geq \mathscr{Q}(\mathscr{C}')$. We have $\mathscr{Q}(\mathscr{C}') = \mathscr{Q}(\mathscr{C}^*)$. □

4.3.2 Size of the Compact Representation

Worst Case Analysis In the worst case, the number of nodes in the compact representation can reach to the number of nodes in the network. Each modules in the network will have exactly two nodes.

Theorem 4.3.2 *The size of a compact representation constructed by the Algorithm 1 is at most the size of its original network.*

Proof The proof is based on the fact that each module is represented by two nodes in the compact representation and each module in the network contains at least two nodes.

Assume that there exists a module that contains only a single node. The modularity of that module will be negative (unless the single node in the module is isolated, but we remove isolated nodes from the network as the beginning as they have no contribution to the modularity). We obtain the contradiction to Lemma 4.3.3.

In case the network has a module with a single node, we can always merge that module to some neighbor module and obtain a new set of modules with higher modularity. □

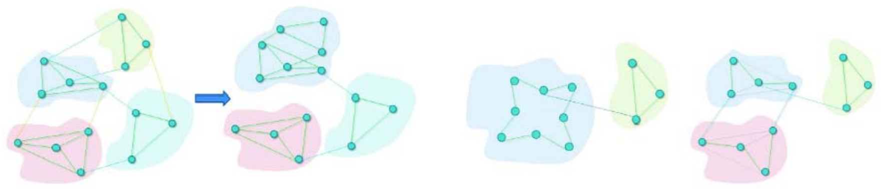
In our experiments, the actual size of the compact representation is far smaller than the theoretical bound.

Social Networks The number of nodes in the compact representation is at most the number of modules in the network. Hence, the average number of modules is an upper bound for the average size of the compact representation. In some social networks [11, 12, 22], the distribution of the module size s follows the power-law form $P(s) = c \cdot s^{-\alpha}$ for some constant $\alpha \approx 2$ at least for a wide range of s. If the network has n nodes, then the average number of modules $\overline{k}$ will be

$$\overline{k} = \frac{n}{\overline{s}} = \frac{n \times \sum_{s=2}^{n} c \cdot s^{-\alpha}}{\sum_{s=2}^{n} c \cdot s^{-\alpha} \cdot s} = \frac{n \times \sum_{s=2}^{n} s^{-\alpha}}{\sum_{s=2}^{n} s^{1-\alpha}} \tag{4.10}$$

where $\overline{s}$ is the average module size of the network.

When $\alpha = 2$ the denominator in (4.10) can be approximated by $\ln n$ and the sum $\sum_{s=2}^{n} c \cdot s^{-\alpha}$ in the numerator is bounded by a constant. Hence, the average size of the compact representation is upper-bounded by $O(\frac{n}{\log n})$.



(a) Removing inter-modules links (b) Adding intra-module links

Fig. 4.3 Small changes may affect network modular structure. (**a**) Removing inter-modules links usually results in more separated modules. However, in this case removing *yellow* inter-modules links make modules merged. (**b**) Adding intra-module links usually strengthen the module. However, in this case, adding 6 *white* links into the *light blue* module make it split into two

4.4 Partition Based on Evolution History

Assume that we have a network $G = (V, E)$ and its modular structure $\mathscr{C} = \{C_1, C_2, \ldots, C_l\}$. The connection inside a module is denser than connection between modules i.e., we should have more intra-module links than inter-modules ones. Hence, intuitively adding intra-module links or removing inter-modules links will make modules "stronger" and vice versa removing intra-module links or adding inter-modules links will make the existing module structure less clear. In fact, removing inter-modules links may cause merging of modules as shown in Fig. 4.3(a). When two modules have less "distraction" caused by other modules, they become more attractive to each other and can be combined as one. Other rare but possible events are splitting of the module when adding intra-links. Figure 4.3(b) show an example in that adding 6 more white links make the module split into two smaller ones. From the quantitative aspect, one can verify that adding an intra-module link will not always increase the modularity of the module. Assume that we have a module C_i that cannot be divided further in order to increase the modularity. Among all possible bisections of C_i, consider $\{X, Y = C_i \backslash X\}$ that X, Y are the most loosely connected e.g., X, Y are obtained through the sparsest cut in the subgraph induced by C_i. In the case of new links crossing X and Y, they will enhance the local structure of C_i. Otherwise, adding new links that both ends belong to X or Y will make the connection inside either subcomponent X or Y stronger but weaken the structure of C_i, thus leading to the splitting of X and Y. Similar observations can be seen for adding and removing vertices.

The behavior of splitting modules makes it extremely challenging to adaptively update evolving modules, especially on the compact representation where each module is represented by only two nodes. The modules splitting requires us to "uncompact" the correcting modules. We will discuss our proposed solution in the next section.

4.4.1 Algorithm

The algorithm for *Modules Identification in Evolving Networks* (MIEN) is presented in Algorithm 2. The presented algorithm is a general approach such that it allows the use of any existing algorithm $\mathscr{A}$ on modules identification. The only requirement for $\mathscr{A}$ is to work with weighted networks. This requirement is reasonable as most existing modules identification algorithms have their weighted versions.

Assume that at time t, changes of the network are given by $(\Delta V^t, \Delta E^t)$. Each link (u, v) in ΔE^t will affect the weights of links crossing some modules C'_i, C'_j or link that both ends lie in a same module C'_i in the compact representation. Updating the weight of corresponding links in the compact representation is easy as lines 8 to 15 in Algorithm 2.

However, the compact representation of a network may not reflect evolutionary phenomena in its original network. We note that merging of two modules C_i and C_j in the network G corresponds to the joining of C'_i and C'_j in the compact representation of G. However, if a module is split or some of its member are leaving to join in other modules,running module identification algorithm $\mathscr{A}$ only on the compact representation will not be able to detect those changes. Hence, when the network evolves, we will modify the compact representation based on the changes in such a way that the new compact representation allows the module identification algorithm to recognize the changing in memberships.

Note that the most sensitive nodes to changing membership are newly removed or added nodes or the ones that are incident to recently added or removed links. For each of those nodes, say node u, we 'exclude' it from its compact module C_i where $i = mb(u)$ and assign it to a new singleton module, say C_k. It is equivalent to replacing the module C'_i in the compact representation by two compact modules C'_i and C'_k. Figure 4.4 shows an example of excluding a node from its module. At the beginning, the module contains 5 nodes and 10 links of unit weight. The corresponding module in the compact representation has two nodes connecting by a link of weight 10. After the network evolves, the node marked with star has some new incident links and is excluded from its module. The module at the beginning now contains only 4 nodes and the link weights in the compact representation are redistributed as shown in the figure.

The sketch of the algorithm is as follows. At the very beginning, we obtain the modular structure of the network $G(V, E)$ using algorithm $\mathscr{A}$, lines 3 to 4. The compact representation $G'(V', E')$ of G is then constructed using Algorithm 1. From now on, all operations will be performed on G' to reduce the computational cost and time complexity.

For each time point t, we update the compact representation according to the given change $(\Delta V^t, \Delta E^t)$ as in lines 8 to 26. We handle with four separate type of changes: set of old nodes removed $(V^{t-1} \cap \Delta V^t)$, set of new nodes coming $(\Delta V^t \backslash V^{t-1})$, set of old links removed $(E^{t-1} \cap \Delta E^t)$ and set of new links coming $(\Delta E^t \backslash E^{t-1})$.

The set of nodes that are removed at time t is given by $V^{t-1} \cap \Delta V^t$. In lines 8 to 13, we remove those nodes from their modules. Whenever their modules contain

Algorithm 2 MIEN—Modules Identification in Evolving Networks

```
1: **Input:** $G^0(V^0, E^0), (\Delta V^1, \Delta E^1), \ldots, (\Delta V^s, \Delta E^s)$
2: **Output:** Partitions $\mathscr{C}^1, \mathscr{C}^2, \ldots, \mathscr{C}^s$ and 'parents' of each module.
3: Find the partition $\mathscr{C}^0$ using the selected algorithm $\mathscr{A}$.
4: $\mathscr{C} = \mathscr{C}^0 \leftarrow \mathscr{A}(G^0) = \{C_1, C_2, \ldots, C_{l_0}\}$
5: $G'(V', E') \leftarrow$ Compact representation of $(G^0, \mathscr{C}^0)$ using Algorithm 1.
6: **for** $t \leftarrow 1$ to $s$ **do**
7:    $\mathscr{C}' \leftarrow$ Partition of the compact representation $G'$ corresponding to $\mathscr{C}$ (as in Algorithm 1).
8:    **for all** $u \in V^{t-1} \cap \Delta V^t$ **do**
9:       Remove $u$ from $\mathscr{C}$: $C_{mb(u)} = C_{mb(u)} - \{u\}$
10:      **if** $|C_{mb(u)}| = 1$ **then**
11:         Remove $y_{mb(u)}$ from $C'_{mb(u)}$ and double link weights of links coming from $x_{mb(u)}$
12:      **end if**
13:   **end for**
14:   **for all** $u \in \Delta V^t \setminus V^{t-1}$ **do**
15:      Create a new module $C^{t-1}_{mb(u)} = \{u\}$ in $G$
16:      Create a new module $C'_{mb(u)} = \{x_{mb(u)}\}$ in $G'$
17:   **end for**
18:   **for all** $(u, v) \in \Delta E^t$ **do**
19:      $\Delta_{u,v} = \begin{cases} +w(u,v) & (u,v) \in \Delta E^t \setminus E^{t-1} \\ -w(u,v) & (u,v) \in E^{t-1} \setminus \Delta E^t \end{cases}$
20:      $i \leftarrow mb(u),\ j \leftarrow mb(v)$
21:      **if** $i \neq j$ **then**
22:         $\forall (u', v') \in C'_i \times C'_j : w(u', v') = w(u', v') + \frac{\Delta_{u,v}}{|C'_i||C'_j|}$
23:      **else**
24:         $w(x_i, y_i) = w(x_i, y_i) + \Delta_{u,v}$
25:      **end if**
26:   **end for**
27:   Run algorithm $\mathscr{A}$ on the updated compact representation: $\mathfrak{C} \leftarrow \mathscr{A}(G') = \{C'_1, C'_2, \ldots, C'_{l_t}\}$
28:   Refine the module structure $\mathfrak{C}$ in $G'$ following Lemmas 4.3.2, 4.3.3.
29:   Reconstruct partition $\mathscr{C} = \{C^t_1, C^t_2, \ldots, C^t_{l_t}\}$ from $\mathfrak{C}$.
30:   $\mathscr{C}^t \leftarrow \mathscr{C}$
31:   **for** $i \leftarrow 1$ to $l_t$ **do**
32:      parents$(C^t_i) \leftarrow \{C^{t-1}_j | x_j, y_j \in C'_i\}$
33:   **end for**
34:   $G'(V', E') \leftarrow$ Compact representation of $(G', \mathfrak{C})$
35: **end for**
```

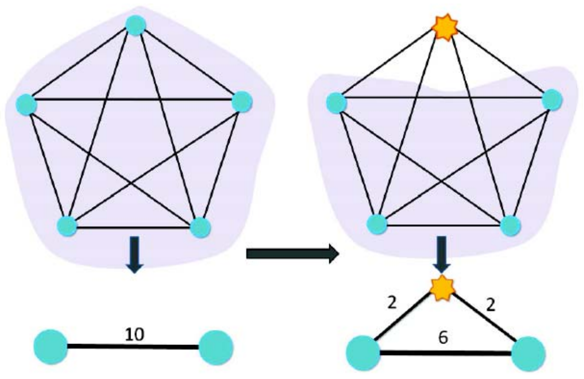


Fig. 4.4 Excluding a node from its module. Links' weights are also updated accordingly

only one node we simplify the corresponding module in the compact representation. Lines 14 to 17 handle new coming nodes at time t that are given by $\Delta V^t \setminus V^{t-1}$. For each new node, we create a new module contain only that node and a corresponding module in the compact representation. We update removing links and adding links in lines 18 to 26 in which we distinguish two cases: links crossing two modules (lines 21 to 22) and links resting completely in one module (lines 23 to 25).

After the changes in links and nodes are fully incorporated into the compact representation, we run the algorithm $\mathscr{A}$ again on the compact representation $G'(V', E')$ to obtain the module structure in G'. To enable transforming back the modular structure in G' to the modular structure in the network $G^t(V^t, E^t)$, we refine the modular structure $\mathfrak{C}$ in G' following Lemma 4.3.2 i.e., if the module C_i^{t-1} in V^{t-1} is represented by two nodes x_i, y_i in the compact representation and x_i and y_i be placed in different modules in G' we will move them to a same module (and increase the modularity). We can also merge every module with negative modularity or module with only a singleton node to one of its neighbor module to increase the modularity as in Lemma 4.3.3 and the proof of Theorem 4.3.2.

Finally, we use the refined modular structure in $G'(V', E')$ to obtain the modular structure in $G^t(V^t.E^t)$. It can be done easily by merging modules in the network. For example, if x_i and x_j are identified to be in a same module in the compact representation, then we merge corresponding modules C_i^{t-1} and C_j^{t-1} in the network together and claim that the new module is originated from C_i^{t-1} and C_j^{t-1} as in lines 31 to 33.

4.4.2 *Complexity*

Time to construct the compact representation of a network $G(V, E)$ using Algorithm 1 is $O(|V| + |E|)$. We iterate through each link exactly once.

Time complexity of Algorithm 2 depends on the selection of the algorithm $\mathscr{A}$ and the size of the compact network representation. In the compact representation, the number of nodes is no more than twice the number of modules in the original network that is relative small to the number of nodes in the original network. Updating the compact representation can introduce at most $O(\text{vol}(V_C))$ new nodes and links into the compact representation. Hence, we only need to run the algorithm $\mathscr{A}$ on a compact network that size is proportional to the amount of changes $|\Delta V^t| + |\Delta E^t|$ instead of running $\mathscr{A}$ on the complete network $G^t(V^t, E^t)$.

For very large networks such as World Wide Web, the changing part is much smaller than the scale of whole network. Hence, our online algorithm works much faster than running algorithm $\mathscr{A}$ on individual snapshots. For dynamic biological networks that the structures evolve rapidly, our algorithm can be applied to recompute the structures whenever changes happen.

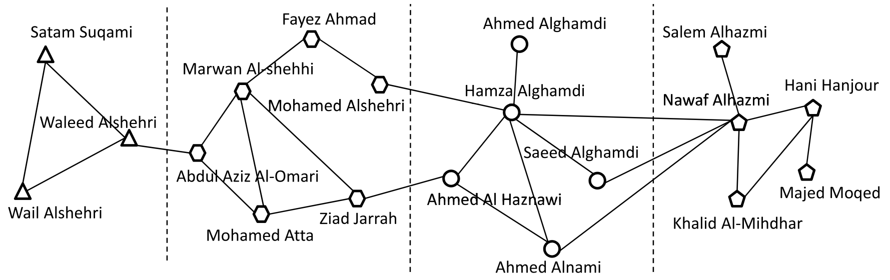
In the worst case, when all nodes are excluded from their modules i.e., $V_C = V^t$ the compact representation will be exactly the original network and the running time is equal to the running time without using the compact representation plus the overhead $O(|V^t| + |E^t|)$ for updating the compact representation. Hence, if the network changes too much since the last time point, our online algorithm will show no advantages compared to using algorithm $\mathscr{A}$ directly. In this case, what we can do is to divide the long evolving duration into smaller ones. Note that we should not divide the duration into too small intervals due to the overhead of updating the compact representation and running algorithm $\mathscr{A}$. Or we can simply reconstruct the compact representation at that time point and continue the updating process.

4.5 Experimental Evaluation

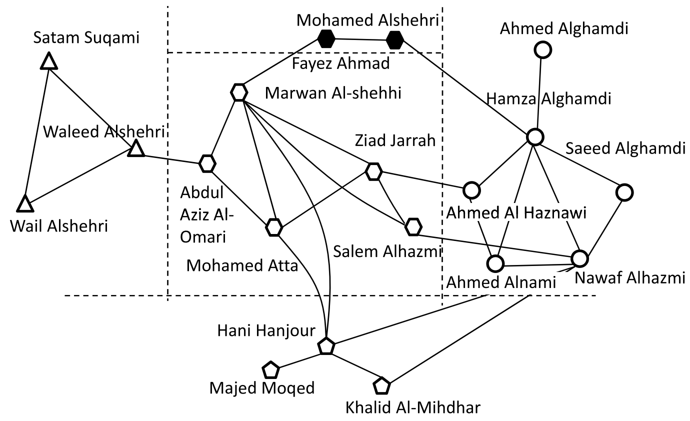
We test the performance of our approach with CNM algorithm [12] that is capable of identifying modules in networks of millions nodes. We show that our approach is faster in significant magnitude in the case of identifying modules in evolving networks.

We perform extensive experiments on different types of networks. We present here results for the Terrorist network, ENRON email network [23], and the citation network of arXiv e-print provided in the KDD Cup 2003 [24]. Time information are extracted in different ways for each network. In the terrorist network, the network topology after and before September 11 are directly given. In the email network, a link between the sender and the receiver is timed to the point the email sent. Based on timing information, we construct two snapshots of the email network at two time points; the later is one week after the former. Two snapshots have approximately 10K links. Two snapshots of the citation networks are also constructed in a same way. Each snapshot includes around 30K nodes and 300K links.

For each data set, modules in the first snapshot of the network are identified using CNM algorithm [12]. Then we run our approach on the second snapshot using the information about the modules identified by CNM. To demonstrate the effectiveness of our approach, we compare it with the results obtained by running CNM directly on the second snapshot.



(a) Before Sep 11st



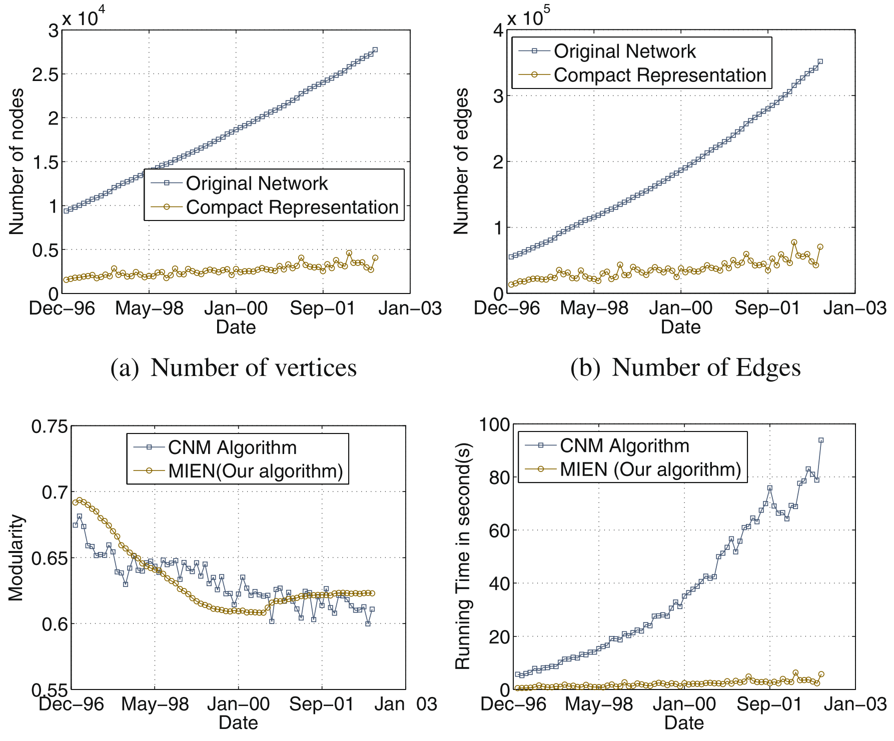
(b) After Sep 11st

Fig. 4.5 The growth of the terrorist network. Nodes with same color belong to the same module. The modular structure returned by the MIEN algorithm is only different with the result using CNM algorithm at place of one node (Nawaf Alhazmi)

We observe that the modules identified by our approach in the second snapshots show high similarity in the structure with the results of running CNM directly, meanwhile the computation cost is significantly reduced as the size of compact representation is much smaller than that of the original network. Figure 4.5 shows the modules in terrorist networks. CNM detects 4 modules in the terrorist network before September 11 and 5 modules after September 11. Our approach also returns 5 modules for the network after September 11, different from the CNM in place of only one node. Note that our partition has a slightly higher modularity, 0.408 comparing to 0.407 of CNM's. We present the modularity and running time achieved by CNM in comparison to our approach in Table 4.1. The difference in modularity is small suggesting our approach results are consistent with the results of the selected algorithm $\mathscr{A}$, CNM in our experiments. However, our approach shows considerable advantage in term of running time. For example, our approach is more than 1000 times faster in the citation network. We further perform an intensive experiment on

Table 4.1 Performance of our approach comparing to CNM

	CNM		MIEN	
	Q	Time	Q	Time
Terrorist network	0.407	<1 ms	0.408	<1 ms
ENRON email	0.568	50 ms	0.555	2 ms
ArXiv citation	0.617	105 s	0.615	5 ms



(a) Number of vertices (b) Number of Edges

(c) Modularity (d) Running Time

Fig. 4.6 Modules Identification of the Citation Network Through Time. Module structure of the network is identified everymonth between Jan. 1997 and Jan. 2003 using CNM and MIEN algorithms. The quantitative measurement modularity, running time, and size of the network are measured at every time point

the Arxiv citation network. We identify modules in the citation network every month in the duration between Jan. 1997 and Jan. 2003. The number of nodes and links in the network are shown in the Figs. 4.6(a) and 4.6(b). The size of network increase linearly and the number of links steps up slightly faster. The compact representation shows its advantages with substantial reduction in size. The modularity and running time at each time point is presented in Figs. 4.6(c) and 4.6(d). The running time of the CNM algorithm increase almost linearly in term of the size of the network that conforms to the known average $O(n \log^2 n)$ complexity of CNM. Because of running on the small size compact representation, the MIEN algorithm is dramatically faster as shown in Fig. 4.6(d). The graph for the modularity (Fig. 4.6(c)) is a rough line that shows the significant changes in memberships at each time point. The CNM produces inconsistent partition between consecutive time points. Only small changes in the network can result in large changes in assigning nodes to modules. On the other hand, the MIEN algorithm results in a smoother graph showing the stability of module detection algorithm. Moreover, predicting the trend of module structure in the network is also easier with MIEN algorithm than with the unstable CNM.

4.6 Conclusions

In this chapter, we have analyzed the evolving of modules in networks and proposed a general approach for adaptively updating modules in evolving networks. We show that using modules information of the network at a given time can help ‘predicting’ effectively the modules of network later. Our algorithm is especially designed for very large networks such as the World Wide Web in that the proportion of changes is relatively small to the scale of the network. The approach shows enormous improvement in terms of running time compared to one of the fastest module identification method, CNM, and promises tremendous applications as it can be integrated to many different modules identification algorithms. Still the approach needs to be further explored as it is limited to nonoverlapping modules. Our future work is to find a more flexible and efficient representation of networks based on their modular structures.

References

1. Macqueen, J.B.: Some methods of classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
2. Donetti, L., Munoz, M.A.: Detecting network communities: a new systematic and efficient algorithm (October 2004)
3. Newman, M.E.J.: Modularity and partition in networks. In: Proceedings of the National Academy of Sciences, pp. 8577–8582 (2005)
4. White, S., Smyth, P.: A spectral partition approach to finding communities in graph. In: SIAM Data Mining Conference (2005)
5. Wu, F.: The Potts model. Rev. Mod. Phys. **54**(1), 235 (1982)
6. Reichardt, J., Bornholdt, S.: Detecting fuzzy partition in complex networks. Phys. Rev. Lett. **93**(21) (2005)
7. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection (Mar 2006)
8. Palla, G., Derenyi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping partition of complex networks in nature and society. Nature **435**, 814–818 (2005)
9. Duch, J., Arenas, A.: Weighted network communities. New J. Phys. **9**, 180 (2007)

10. Palla, G., Farkas, I., Pollner, P., Derenyi, I., Vicsek, T.: Directed network communities. New J. Phys. **9**, 186 (2007)
11. Newman, M.E.J.: Fast algorithm for detecting community structure in networks (September 2003)
12. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks (August 2004)
13. Duch, J., Arenas, A.: Community detection in complex networks using extremal optimization. Phys. Rev. E **72** (2005)
14. Guimera, R., Sales-Pardo, M., Amaral, L.A.N.: Modularity from fluctuations in random graphs and complex networks (Aug 2004)
15. Hopcroft, J., Khan, O., Kulis, B., Selman, B.: Tracking evolving communities in large linked networks. Proc. Natl. Acad. Sci. USA **101**(1), 5249–5253 (2004)
16. Palla, G., Barabasi, A.: T. Vicsek. Quantifying social group evolution. Nature **446**(7136), 664–667 (2007)
17. Pollner, P., Palla, G., Vicsek, T.: Preferential attachment of communities: the same principle, but a higher level. Europhys. Lett. **73**, 478 (2006)
18. Newman, M.E.J.: Analysis of weighted networks. Phys. Rev. E **70**(5), 056131 (2004)
19. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E Stat. Nonlinear Soft Matter Phys. **69**(2) (2004)
20. Arenas, A., Duch, J., Fernandes, A., Gomez, S.: Size reduction of complex networks preserving modularity. New J. Phys. **9**, 176–180 (2007)
21. Blondel, V.D., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. P10008 (2008)
22. Arenas, A., Danon, L., Diaz-Guilera, A., Gleiser, P.M., Guimera, R.: Community analysis in social networks (2003)
23. Sun, J., Papadimitriou, S., Yu, P.S., Faloutsos, C.: Graphscope: Parameter-free mining of large time-evolving graphs. In: KDD (August 2007)
24. Arxiv citation datasets. KDD Cup 2003, in Conjunction with the Ninth Annual ACM SIGKDD. http://www.cs.cornell.edu/projects/kddcup/datasets.html (2003)

Chapter 5
Topology Information Control in Feedback Based Reconfiguration Processes

Alexandru Murgu, Ian Postlethwaite, Dawei Gu, and Chris Edwards

Summary In this paper, we describe an information control and coding framework devoted to reconfiguration processes based on a modified perspective on the Shannon information where the information flows are regarded as network commodities. This interpretation is suitable for independent multipoint-to-multipoint channels in group communication, such as the multiple-access or broadcast channels, and allows the flow control class of techniques to implement the information coding by invoking the multi-layered protocol stack. Iterative parametric dynamic programming is a good modeling candidate for describing the reconfiguration process as multi-objective relational optimization in a multilevel fashion. At the lower processing level, an auxiliary weighted power Lagrangian problem is solved using dynamic programming associated to the topology control. The upper processing level adjusts the value of the weighting vector in a weighted power Lagrangian formulation which is responsible for the information control monitoring. The low level solution process is repeated until the optimal solution of the nonseparable optimization problem is attained by the optimal solution of an auxiliary weighted power Lagrangian problem. The application of this information control framework is to the management traffic in self-healing networks of UAV surveillance missions.

5.1 Introduction and Motivation

One of the features of the new information organizational forms that is of great interest to the military establishment is to increase the ability to adapt to a dynamic environment through networked and/or distributed information processing technologies. Command and control of autonomous unmanned aerial vehicles (UAVs) by remote supervision over ad-hoc networks is a demonstrated technology, including the two-way communication of commands to the aircraft fleets and telemetry data to the

A. Murgu (✉) · I. Postlethwaite · D. Gu · C. Edwards
Department of Engineering, Control and Instrumentation Group, University of Leicester, Leicester LE1 7RH, UK
e-mail: murgu@btinternet.com

headquartered supervisors [1]. The important aspect is that the network is not configured in advance, but the ad-hoc networking protocols self-organize in response to the dynamic environment, aircraft motion, and changing network demands. The traffic transport allows the mobile nodes to store and carry delay-tolerant data between nodes in the network. This concept enables communication in stressed or highly dynamic networks that otherwise would not be possible and, in some cases, can improve the network performance over relaying or direct communication. Decentralized adaptive model free control strategies can mitigate the interference and uncertainty effects that cannot be predicted in advance. Multivariable extremum seeking control is applied to the problem of UAV motion control in a communication field, and such an approach can lead to improvements of communication ability over position-based policies [2]. Networked communication demands of UAVs are large compared to manned aircraft since telemetry, command and control, health and safety, and payload/mission management data must be sent from multiple UAVs to other UAVs in the vicinity. Other communication needs are related to detecting, sensing, and obstacle avoidance requirements. This may require onboard radar (active sensing), backhauling of image data (passive sensing), transponders, or cooperative sharing of information between UAVs. Amongst the networking technologies, meshing is a networking architecture where each node (e.g., a radio on a UAV or coordination center) can act as a relay to forward data. Communication between a UAV and a supervisor can take place over several hops through intermediate nodes. The shorter range simplifies the link requirements, and the bandwidth can be reused more frequently and thus more efficiently. UAV-to-UAV communication can be direct and also benefit from the mesh routing protocols that employ additional relays as needed to maintain communication. However, such meshing requires intermediate nodes to be present for such relaying to take place. Nodes may be required to move specifically in order to support communication.

Reconfiguration of the UAVs working in networked environments is primarily seen as an operational planning framework concerning the optimal functional organization and deployment of the UAVs in the battlefield during ongoing missions in the presence of subsystem failures, battle-damage or changes of asset functionality as a result of multi-standard cooperative tasks [3]. The defining attribute of the networked operational framework is concerned with the control of dynamics of the leaving/joining entities in the network and the information processing related to failure localization and isolation in the network. Related features such as quality of service (QoS), resilience, control of distributed interactions, resource allocation and location based control are the basic artifacts that characterize the operational means by which the mission success can be guaranteed. Increasing demand for the overall system safety and reliability calls for integrating the fault detection and isolation (FDI) and fault-tolerant control (FTC) methods at the very early stages of control design. Typically, FDI involves checking the consistency of interactions and measurements from the real-time system operation. With FTC, the problem is controlling to the maximum extent possible, the interactions and operation of the system in the presence of faults.

The paper is organized as follows. In Sect. 5.2, we discuss the group communication paradigm for realizing the FDI/FTC system functionality based on a modified perspective on the Shannon information where the information flows are regarded as network commodities. Section 5.3 is devoted to the formal statement of the reconfiguration process as a optimization problem. A feedback information control loop is formulated via dynamic programming as multi-objective relational optimization. In Sect. 5.4, the topology information control solution is presented and qualitative results are discussed. Finally, in Sect. 5.5, we draw conclusions of this research work and discuss some perspectives on the potential future work.

5.2 Group Communication Networking

With the growth of complexity and functional extent, the network management operations are steadily becoming autonomous, scalable, interoperable and adaptable to increasingly dynamic and distributed network demands. The emergence of distributed computing environments is a major shift in designing the Operations Support System (OSS) for current services. Packet switched networks use the spanning tree protocol (IEEE 802.1d) for routing data packets in the network. Spanning Tree Protocol (STP) is standardized in IEEE 802.1d and it is a Layer 2 protocol that can be implemented in switches and bridges. STP essentially uses a shortest-path approach in forming a tree that is overlaid on top of the mesh engineered networks. Spanning trees are used primarily to avoid the formation of cycles or loops in the network. For example, unlike IP packets, the Ethernet frames do not have a time-to-live field and the STP prevents the formation of loops in the network by blocking the redundant links. Correspondingly, the load is concentrated on a single link which leaves it at risk of failures and with no load balancing mechanism [4]. The root of the information distribution tree is chosen based on the bridge priority, and the path cost to the root is propagated throughout so that each switch can determine the state of its ports. Only the ports that are in the forwarding state can forward incoming frames. This ensures a shortest single path to the root. Whenever there is a change in the connectivity topology, switches rerun the protocol that can take 30 to 60 seconds. At any one time, only one spanning tree dictates the network [5]. These shortcomings are listed as follows: (1) Spanning trees restrict the number of ports being used (in high-capacity networks, this restriction translates to a very low utilization of the network); (2) STP has reduced resiliency and a very high self-healing time (30 to 60 seconds) after a link failure; (3) STP does not have any mechanisms to balance the traffic load across the network; (4) STP does not support directly the quality of service (QoS) mechanisms. An improvement of STP is the Rapid Spanning Tree Protocol (RSTP) specified in IEEE 802.1w [6, 7]. RSTP is designed to reduce the number of port states from five in STP to three: discarding, learning, and forwarding are the fundamental options for connectivity enhancement. Through faster aging time and rapid transition to forwarding state, RSTP is able to reduce the convergence time depending on the network topology. Additionally, the topology change notification is propagated throughout the network simultaneously, unlike STP, in which a node first notifies the root and then the root broadcasts the changes. Similar to STP, there is only one spanning tree over the whole network. RSTP still blocks the redundant links to ensure loop free paths leaving the network underutilized, vulnerable to failures, and with no load balancing facility. Multiple Spanning Tree Protocol (MSTP) is defined in IEEE 802.1 [7]. MSTP uses a common spanning tree that connects all of the regions in the topology. The regions in MSTP are multiple instances of the spanning tree. Each instance is an instance of the RSTP. An instance of RSTP governs a region, where each region has its own regional root. The regional roots are in turn connected to the common root that belongs to the common spanning tree. Since MSTP runs a pure RSTP as the underlying protocol, it inherits some drawbacks of RSTP as well. However, a failure in MSTP can be isolated into a separate region leaving the traffic flows in other regions untouched. In addition, the administrators can perform light load balancing manually by assigning certain flows to a specific spanning tree. The primary motivation of this study is to allow the information allocation flexibility of using more than one spanning tree while a flow is on its way to the destination and describe a mathematical framework for modeling the connectivity reconfiguration information to be broadcast around the faulty region of the network. In order to avoid the formation of cycles in the network, certain restrictions are imposed. The basic idea is to identify multiple spanning trees for management information distribution and to index these trees sequentially in an ordered list. Typically, the Virtual Local Area Network (VLAN) identification numbers can be used as sequences for the spanning trees. Frames of a flow start using one spanning tree and can be switched over to the next spanning trees. None of the other variants of spanning tree allow this flexibility in the information sequencing. This procedure can be repeated until the cooperation information reaches the spanning tree which has the highest identifier in the sequence, as seen in Fig. 5.1. At no point in time is an information frame allowed to change from a spanning tree with a higher identifier to a spanning tree with a lower identifier. A flow is switched, or crossed-over, from one spanning tree to another spanning tree whenever one of the following events occurs: (1) link failure; (2) load imbalance.

The main issue with the Spanning Tree Protocol for topology information distribution used in a Local Area Network (LAN) is the potential conflict with the information SPT. As shown in Fig. 5.2, suppose that the information flow dashed circuit has two physical connections to provide redundancy in the event of a connection or network failure. This creates the possibility of two physical loops due to the two physical connections. The IEEE 802.1d Spanning Tree Protocols solution to heal the problem of loops is to allow only one path by blocking and separating the redundant paths [8]. Consequently, we are concerned with the modeling of a distributed topology information control management system and the corresponding control

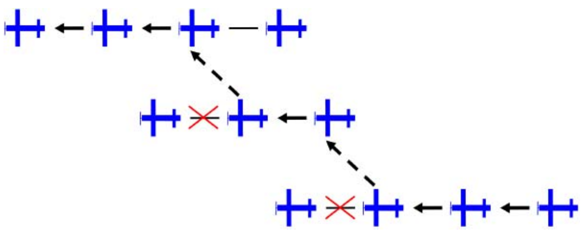


Fig. 5.1 UAV formation involving three cooperating layers

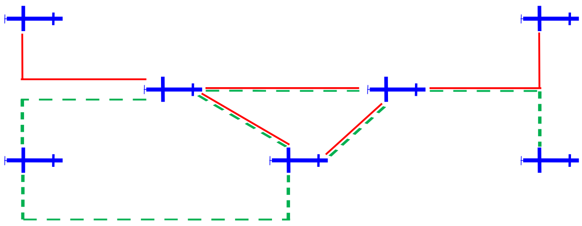


Fig. 5.2 Physical loops of information management

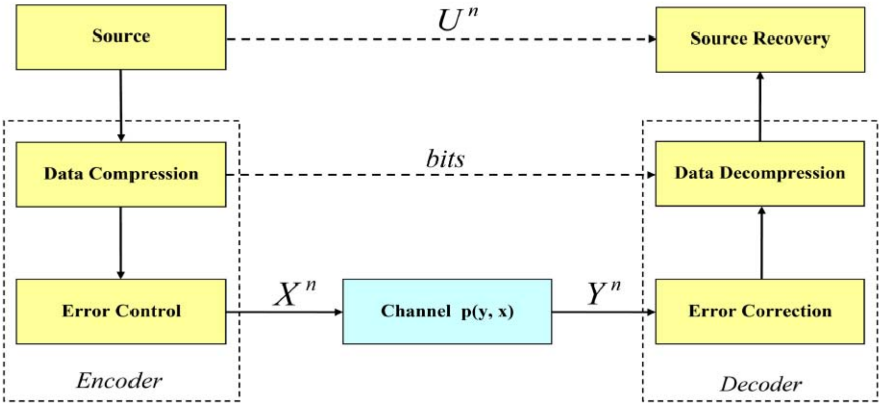


Fig. 5.3 Shannon model of point-to-point communication

processes among the cooperative network elements regarded as agents broadcasting periodical signals to enable the network connectivity information propagation.

Let us consider a group communication network represented by a directed graph $G = (V, E)$, where the set of nodes is $V = \{v_0, v_1, \ldots, v_M\}$ and $E \subseteq V \times V$ is the set of edges. For the edge $(v_i, v_j) \in E$, node i can send messages to node j over a discrete time memoryless channel represented as a triple $(\tilde{X}_{ij}, p_{ij}(y|x), \tilde{Y}_{ij})$ having the capacity $C_{ij} = \max_{p_{ij}(x)} I(X_{ij}; Y_{ij})$. In the channel notation, $\tilde{X}_{ij}$ denotes the transmitted control symbol, $\tilde{Y}_{ij}$ represents the received control symbol, and $p_{ij}(y|x)$ is the transmission error conditional probability. Also, $I(X_{ij}; Y_{ij})$ is the information function of the transmission channel (Fig. 5.3) [9]. At the node $v_i \in V$, a random variable U_i, $i = 0, 1, \ldots, M$, is observed and drawn from a known joint probability distribution $p(U_0, U_1, \ldots, U_M)$. Node v_0 represents the decoder and its goal is to process the received information such that $[U_1, \ldots, U_M]$ can be reliably reproduced at the node v_0. The time is discrete and every N time steps, the node v_i collects a block U_i^N of control symbols. The set of all blocks $[U_0^N(k), U_1^N(k), \ldots, U_M^N(k)]$ collected at time kN, $k \geq 1$, is called the control snapshot. Then, node v_i sends a control signal X_{ij}^N to node v_j related to a connectivity request. This control signal depends on a window of K previous blocks of control sequences U_i^N observed at node v_i and of T previously received blocks of channel outputs, corresponding to noisy versions of the control signals sent by all nodes to the node v_i in the previous T signaling steps (corresponding to NT time steps).

For a block of snapshots observed at time kN, at time $(k + W)N$, an attempt is made to decode at v_0. During the time a block of snapshots spends within the network, arbitrarily complex coding operations are allowed, that is, nodes can exchange information, redistribute their load and perform joint source-channel coding operations [10]. The only constraint imposed is that all the control information eventually be delivered to the destination in finite time. The decoder tries to produce an estimate of the block of snapshots $[U_0^N(k), U_1^N(k), \ldots, U_M^N(k)]$ based on the local observations $U_0^N(k)$ and the previous W blocks of N channel outputs generated by control signals sent to v_0 by the other nodes. The topology information for this network consists of:

- Parameters N, K, T and W
- Encoding functions at each node

$$g_{ij} : \bigotimes_{l=1}^{K} \tilde{U}_i^N \bigotimes_{t=1}^{T} \bigotimes_{m=0}^{M} \tilde{Y}_{mi}^N \rightarrow \tilde{X}_{ij}^N, \quad 0 \le i, j \le M \tag{5.1}$$

- Decoding function at node v_0:

$$h : \tilde{U}_0^N \times \bigotimes_{w=1}^{W} \bigotimes_{m=1}^{M} \tilde{Y}_{m0}^N \rightarrow \bigotimes_{m=1}^{M} \tilde{U}_m^N \tag{5.2}$$

- Block error probability:

$$P_e^{(N)} = \Pr\big\{\big[U_1^N, \ldots, U_M^N\big] \neq \big[\hat{U}_1^N, \ldots, \hat{U}_M^N\big]\big\} \tag{5.3}$$

The blocks of snapshots $[U_1^N, \ldots, U_M^N]$ can be reliably communicated to v_0 if there exists a sequence of control signals as presented above, which determines that $P_e^{(N)} \rightarrow 0$ as $N \rightarrow \infty$, for some finite values K, T and W independent of N [11].

Proposition 1 *Let S denote a nonempty subset of node indices that does not contain node* 0, $S \subseteq \{0, 1, \ldots, M\}$, $0 \in S^c$. *Then, it is possible to communicate* $[U_1, \ldots, U_M]$ *reliably to v_0 if and only if, for all S as above,*

$$H(U_S|U_{S^c}) < \sum_{i \in S, j \in S^c} C_{ij} \tag{5.4}$$

where $H(U_S|U_{S^c})$ is the set cross-entropy function [10].

Proof We regard the network as a pipeline, in which "packets" (i.e., blocks of N control symbols injected by each source) take NW units of time to flow and each network element gets to inject L packets total. The interest is in the behavior of this pipeline in the regime of large L. For any fixed L, the probability of at least one of the L blocks being decoded in error is

$$P_e^{(LN)} = 1 - \big(1 - P_e^{(N)}\big)^L \tag{5.5}$$

From the existence of a control signal with low block probability of error, we can infer the existence of control signal s for which the probability of error for the entire pipeline is low as well, by considering a large enough block length N. If there is a suitable code as defined in the problem statement, then we must have

$$\begin{aligned} &H\big(U_1^{LN}, U_2^{LN}, \ldots, U_M^{LN} | \hat{U}_1^{LN}, \hat{U}_2^{LN}, \ldots, \hat{U}_M^{LN}\big) \\ &\quad \le P_e^{(LN)} \log\big(\big|\tilde{U}_1^{LN} \times \tilde{U}_2^{LN} \times \cdots \times \tilde{U}_M^{LN}\big|\big) + h\big(P_e^{(LN)}\big) \end{aligned} \tag{5.6}$$

where $h(\cdot)$ is the binary entropy function [10], and $\hat{U}_i^{LN} = [\hat{U}_i^N(1), \hat{U}_i^N(2), \ldots, \hat{U}_i^N(L)]$ denotes L blocks of N snapshots reconstructed at v_0. We also define

$$\delta\big(P_e^{(LN)}\big) = \frac{1}{LN}\Big[P_e^{(LN)} \log\big(\big|U_1^{LN} \times U_2^{LN} \times \cdots \times U_M^{LN}\big|\big) + h\big(P_e^{(LN)}\big)\Big] \tag{5.7}$$

It follows that

$$H\big(U_1^{LN}, U_2^{LN}, \ldots, U_M^{LN} | U_0^{LN}, Y_{10}^{BN}, Y_{20}^{BN}, \ldots, Y_{M0}^{BN}\big) \le LN\delta\big(P_e^{(LN)}\big) \tag{5.8}$$

where $Y_{ij}^{BN} = [Y_{ij}^N(1), Y_{ij}^N(2), \ldots, Y_{ij}^N(B)]$ are $B = W + (L-1)$ blocks of N channel outputs observed by node v_j while communicating with node v_i. From the chain rule for entropy and the fact that the conditioning does not increase the entropy, for any $S \subseteq \{0, \ldots, M\}$, $0 \notin S$, it follows that

$$\begin{aligned} H\big(U_S^{LN} | U_{S^c}^{LN}, Y_{S\to S^c}^{BN}, Y_{S^c\to S^c}^{BN}\big) &\le H\big(U_S^{LN} | U_{S^c}^{LN}, Y_{S\to 0}^{BN}, Y_{S^c\setminus\{0\}\to 0}^{BN}\big) \\ &\le LN\delta\big(P_e^{(LN)}\big) \end{aligned} \tag{5.9}$$

$$\begin{aligned} H\big(U_S^{LN}\big) &\le I\big(U_S^{LN}; U_{S^c}^{LN}\big) + \sum_{i\in S, j\in S^c} BNC_{ij} \\ &\quad + LN\delta\big(P_e^{(LN)}\big) \end{aligned} \tag{5.10}$$

Using the fact that the sources are independent, then (5.10) can be rewritten as

$$H(U_S | U_{S^c}) \le \frac{B}{L} \sum_{i\in S, j\in S^c} C_{ij} + \delta\big(P_e^{(LN)}\big) \le \frac{W+L-1}{L} \sum_{i\in S, j\in S^c} C_{ij} + \delta\big(P_e^{(LN)}\big) \tag{5.11}$$

We observe that this inequality holds for all finite values of L, so it must also be the case that

$$H(U_S | U_{S^c}) < \inf_L \frac{W+L-1}{L} \sum_{i\in S, j\in S^c} C_{ij} + \delta\big(P_e^{(LN)}\big) = \sum_{i\in S, j\in S^c} C_{ij} + \delta\big(P_e^{(LN)}\big) \tag{5.12}$$

Since $\delta(P_e^{(LN)}) \to 0$ as $P_e^{(N)} \to 0$, then (5.4) holds. $\square$

Adaptive decentralized mobility control algorithms can mitigate the noise and networking environment uncertainties. Using these algorithms, relay nodes in a linear chain can be moved to locations that improve the overall throughput capacity of the network. This concept can also be extended to more general network configurations. Likewise, the concept of data transport allows mobile nodes to store and carry delay tolerant data between nodes in the network. This concept mitigates the decreased connectivity in stressed or fractured networks, enabling communication that otherwise would not be possible. Quality of service demands can be mapped into a phase space that indicates what mobility methods are needed to achieve a given demand. Distributed control model is well suited for partial dynamic reconfiguration, since it enforces a systematic way to deal with state, execution of operations and information encapsulation. Reconfiguration can be provided as yet another service of the middleware. The main objective is to provide an extra degree of transparency (reconfiguration transparency) which has two implications: (1) reconfigurable objects will offer exactly the same interface as their static counterparts, thus will make such capability transparent to the network operations; (2) reconfiguration management (creation, destruction and control object persistence) will be performed automatically (implicitly), although that will not prevent from having a higher degree of control (explicit management). The important issue of dynamic reconfiguration is state consistency. It may be of interest to keep state information of network configurations, so it can be recovered at a later instantiation of the same environment conditions, or even to be migrated to a software task.

5.3 Reconfiguration Process Optimization

5.3.1 Topology Information Model

Let $\varphi(v_i, v_j)$ be a feasible flow in such network, with M sources $v_1, \ldots, v_M$, supply R_i at source v_i, and a single coordination node v_0. If there is a feasible flow φ, then this uniquely determines at each node v_i the number of bits that need to be sent to each of its neighbors. The topology information model is defined as follows:

- Consider the directed acyclic graph G' of G induced by φ, by taking $V(G') = V(G)$, and $E(G') = \{(v_i, v_j) \in E : \varphi(v_i, v_j) > 0\}$. Let us define a permutation $\pi : \{0, 1, \ldots, M\} \to \{0, 1, \ldots, M\}$, such that $[v_{\pi(0)}, v_{\pi(1)}, \ldots, v_{\pi(M)}]$ is a topological sort of the nodes in G.
- For a block of snapshots $\mathbf{U}(k) = [U_0^N(k), U_1^N(k), \ldots, U_M^N(k)]$ captured at time kN, at time $(k+l)N, l = 0, 1, \ldots, M$, the node $v_{\pi(l)}$ receives all bits with portions of the encodings of $\mathbf{U}(k)$ generated by the nodes upstream in the topological order. Together with its own encoding of $U_{\pi(l)}^N(k)$, all the bits for $\mathbf{U}(k)$ up to and including node $v_{\pi(l)}$ will be available there, and thus can be routed to nodes downstream in the topological order.
- Consider all edges of the form $(v_{\pi(k)}, v')$ for which $\varphi(v_{\pi(k)}, v') > 0$.

  1. Collect $m = \sum_{v'} \varphi(v', v_{\pi(k)})$ information sent by the upstream nodes v'.
  2. Consider now the set of all downstream nodes v'', for which $\varphi(v_{\pi(k)}, v'') > 0$.

3. Due to flow conservation constraints, we have

$$\sum_{v''} \varphi\big(v_{\pi(k)}, v''\big) = m + R_{\pi(k)} \tag{5.13}$$

   where $R_{\pi(k)}$ is the rate allocated to node $v_{\pi(k)}$.
4. For each v'' as above, define $g^{(k)}_{\pi(k)v''}$ to be a message such that $|g^{(k)}_{\pi(k)v''}| = \varphi(v_{\pi(k)}, v')$. Partition the $m + R_{\pi(k)}$ according to the values of φ and send them downstream.

- To decode at time $(k+M)N$, node v_0 reassembles the set of bin indices out of the noisy control signals received in the last M time steps (one from each neighbor) and then performs typical set decoding to recover the delivered block of snapshots $[U_1^N(k), \ldots, U_M^N(k)]$.
- Concerning the network flows, if the flow φ is feasible in a network G, then for all $S \subseteq \{0, 1, \ldots, M\}$, $0 \in S^c$, we have

$$\sum_{i\in S} R_i = \sum_{i\in S, j\in V} \varphi(v_i, v_j) = \sum_{i\in S, j\in S^c} \varphi(v_i, v_j) \le \sum_{i\in S, j\in S^c} C_{ij} \tag{5.14}$$

  where facts such as flow conservation property of a feasible flow (the flow injected by the sources has to go somewhere in the network, and in particular all of it has to go across a network cut with the destination on the other side) and the fact that in any flow network, the capacity of any cut is an upper bound to the value of any flow, are used. If for all partitions S as above, we have

$$H(U_S|U_{S^c}) < \sum_{i\in S, j\in S^c} C_{ij} \tag{5.15}$$

  then

$$P_e^{(N)} \to 0, \quad \text{as } N \to \infty \tag{5.16}$$

Several topologies can be considered to realize the control information propagation, but the multistage interconnection topologies (Fig. 5.4) offer several modeling and implementation advantages [12]. In such topologies, the natural question is the information control flow optimization, that is, given a nonempty set $\tilde{R}$ for the feasible rates, choose among the multiple assignments of flow variables and find if there is an optimal flow and the corresponding optimized topology. State persistence refers to the state transitions to and from any kind of topological arrangement, so it can be later restored or even migrated. This concept is better suited to the topology information model, since state is explicitly defined.

Define the cost function

$$J(\varphi) = \sum_{(v_i, v_j)\in E} c(v_i, v_j) \cdot \varphi(v_i, v_j) \tag{5.17}$$

where $c(v_i, v_j)$ is a constant which when multiplied by the number of blocks $\varphi(v_i, v_j)$ that a flow φ assigns to an edge (v_i, v_j) determines the cost of sending

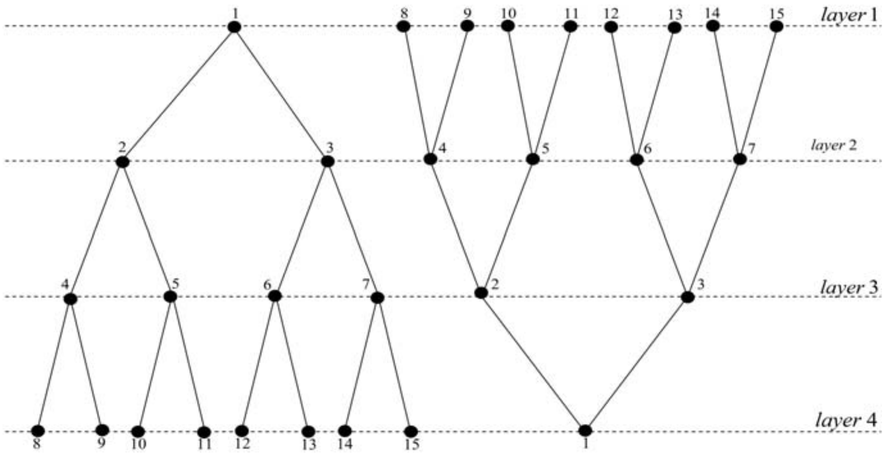


Fig. 5.4 Multi-layer flow network

the information over the channel. The resulting optimization problem can be formulated as follows

$$\min \sum_{(v_i,v_j)\in E} c(v_i,v_j)\cdot\varphi(v_i,v_j) \tag{5.18}$$

subject to:

- Flow constraints (capacity/flow conservation):

$$\varphi(v_i,v_j)\le C_{ij}, \quad 0\le i,j\le M \tag{5.19}$$

$$\varphi(v_i,v_j)=-\varphi(v_j,v_i), \quad 0\le i,j\le M \tag{5.20}$$

$$\sum_{v\in V}\varphi(v_i,v)=0, \quad 1\le i\le M \tag{5.21}$$

- Rate admissibility constraints:

$$H(U_S|U_{S^c})<\sum_{i\in S}\varphi(s,v_i)\le\sum_{i\in S,\,j\in S^c}C_{ij} \tag{5.22}$$

$$S\subseteq\{0,1,\dots,M\}, \quad 0\notin S \tag{5.23}$$

$$\varphi(s,v_i)=R_i, \quad 1\le i\le M \tag{5.24}$$

Dynamically controlled network elements do not differ from static ones from the functional point of view. Both expose exactly the same functional interface to the network signalling system. The main difference lies in the way the element is created and destroyed, as well as how its networking persistence is managed. Considering these two aspects, the first one only affects to reconfigurable adapters, while the second has also some implications on the element itself, since it must be able to import and export its internal state. The signalling core of a dynamically reconfigurable network element includes extra logic for controlling the execution state of the element. A previous step for the eviction of the element from the reconfigurable area will be to issue a stop request. This will disable the reception of incoming method invocations, and wait for the completion of the pending ones, and thus guarantee state coherence. On the other side, after the instantiation of a new object (creation) an explicit start request will activate the object for incoming clients requests.

5.3.2 Information Control Problem

Due to sequential character of the interacting agents on the STP, the dynamic programming is a natural candidate for the modeling framework [13]. While the dynamic programming can be applied to a variety of optimization problems, linear or nonlinear, deterministic and stochastic, it has been applicable only to problems that satisfy the conditions of separability and monotonicity. However, in the present design case of sequential decision making the problem is nonseparable reflecting the constrained optimization under connectivity reliability requirements. Under these circumstances, it is needed a self-healing scheme that has adaptation capabilities to the dynamic environment changes and is able to cope with the uncertainties of the managed elements in the Operations Support System. This is the meaning of the local information allocator concept applied to the faulty service configuration processes.

There are mathematical programming techniques dealing with nonseparable dynamic programming problems whose performance indices are of the form

$$\sum_l F_l(\varphi_l) + \Phi\left[\sum_l g_l(\varphi_l)\right] \tag{5.25}$$

where $\Phi(\cdot)$ is either quasi-concave or quasi-convex. Other proposals include a generalized dynamic programming for a class of combinatorial optimization problems where the monotonicity is not satisfied using information about the local preference relations at each state of the optimization stage. For the purpose of modeling the information control problem, let us consider the following class of nonseparable optimization problems

$$\min J = \Phi\big[J_1(\varphi_1), \ldots, J_k(\varphi_k)\big] \tag{5.26}$$

$$\text{subject to} \quad \sum_{i=1}^{k} g_{ji}(\varphi_i) \le b_j, \quad j = 1, 2, \ldots, m \tag{5.27}$$

$$x_i \in X_i, \quad i = 1, 2, \ldots, k \tag{5.28}$$

where Φ, J_i $(i = 1, 2, \ldots, k)$, g_{ji} $(j = 1, 2, \ldots, m; i = 1, 2, \ldots, k)$ are second order differentiable and X_i $(i = 1, 2, \ldots, k)$ is a subset of n_i dimensional real space. It is assumed that the objective function J is a nondecreasing function of each J_i,

$$\frac{\partial J}{\partial J_i} \geq 0, \quad i = 1, 2, \ldots, k \tag{5.29}$$

Furthermore, each J_i is assumed to have a nonnegative minimum under the constraints given in (5.27) and (5.28). Each J_i can be viewed as an individual performance measure associated with the decision variable x_i, while J serves as the overall performance measure. Condition (5.29) implies that an improvement in an individual performance measure always leads to the improvement of the overall performance measure. Such assumption is motivated by the need that the control and signaling processes among the cooperative agents broadcasting periodical connectivity signals will help to obtain a better network connectivity performance. Problem (5.26)–(5.28) is nonseparable in the sense of dynamic programming, that is, if there do not exist the functions ϕ_i, $i = 1, 2, \ldots, k$, such that the overall objective function J can be decomposed as follows:

$$\varPhi\big[J_1(\varphi_1), \ldots, J_k(\varphi_k)\big] = \varPhi_1\big[\varphi_1, \varPhi_2\big[\varphi_2, \varPhi_3\big[\varphi_3, \ldots, \varPhi_k(\varphi_k)\big]\big]\big] \tag{5.30}$$

The optimality condition is derived under which the optimal solution of the originally nonseparable optimization problem is reached by the optimal solution of an auxiliary weighted pth power Lagrangian formulation. The weighted pth power Lagrangian form is of a separable structure and can be solved at the lower level by dynamic programming for each given weighting vector. The convergence of this multilevel solution scheme is discussed. The basis of performing the system reconfiguration helping the prescribed performance of service classes is finding a control signal path for a new incoming operational asset joining the network while being able to guarantee the quality parameters such as bandwidth and delay of the already existing operations in the networking environment. When more than one path satisfying the bandwidth demand exists, the selection of the path aims to minimize the blocking probability of future reconfiguration requests.

The problem formulated in (5.26)–(5.28) is separable of order k if the assumption (5.25) holds. We propose a separation strategy which enables the development of a solution methodology of iterative parametric dynamic programming for the class of nonseparable problems in (5.26)–(5.28). We formulate the following associated multi-objective optimization problem as follows

$$\min\big\{J_1(\varphi_1), \ldots, J_k(\varphi_k)\big\} \tag{5.31}$$

subject to constraints (5.27)–(5.28). In general, the solution to a multi-objective optimization problem is not unique [14]. Solving such problem (5.31) yields a set of the Pareto frontier. Conceptually, a noninferior solution is one that is not dominated by other feasible solutions. The common way in dealing with highly complex and dynamic systems is based on the hierarchical decomposition of the reconfiguration problem into a sequence of lower level problems that can be solved more easily, based on the "divide and conquer" principle. Such design results in the introduction of a hierarchy of control and decision making layers which are very attractive form a conceptual point of view.

The reconfiguration controller (RC) is responsible for run-time management of the creation and destruction processes, including the networking configuration state persistence management. The RC holds an internal table where already known configurations (those that have been used at any time even if they are not currently instantiated) are registered. For each entry the RC controls: (1) region within the reconfiguration structure where the network element was invoked; (2) a pointer to the memory block where the configuration state was stored. That information can be inserted, removed, updated and looked up through a set of administrative methods.

5.4 Topology Information Control

5.4.1 Lagrangian Solution

Definition 1 A solution $\hat{\varphi} = [\hat{\varphi}_1, \ldots, \hat{\varphi}_k]$ to problem (5.31) is said to be noninferior if there is no other feasible solution $\varphi = [\varphi_1, \ldots, \varphi_k]$ such that $J_i(\hat{\varphi}_i) \geq J_i(\varphi_i)$, $i = 1, 2, \ldots, k$, with at least one strict inequality. We assume in this study that each noninferior solution of (5.31) is attainable.

Proposition 2 *At least one optimal solution of* (5.26)–(5.28) *is attained by a noninferior solution of the multi-objective optimization problem given in* (5.31).

Proof If an optimal solution $\hat{\varphi}$ to (5.26)–(5.28) is inferior in (5.31), then there exists a feasible noninferior solution $\tilde{\varphi}$ such that $J_i(\hat{\varphi}_i) \geq J_i(\tilde{\varphi}_i)$, $i = 1, 2, \ldots, k$, with at least one strict inequality. From (5.29), we know that J is a nondecreasing function of the J_i's, thus

$$\phi\big[J_1(\hat{\varphi}_1), J_2(\hat{\varphi}_2), \ldots, J_k(\hat{\varphi}_k)\big] \geq \phi\big[J_1(\tilde{\varphi}_1), J_2(\tilde{\varphi}_2), \ldots, J_k(\tilde{\varphi}_k)\big] \tag{5.32}$$

If the strict inequality holds, this is a contradiction to the assumption that $\hat{\varphi}$ is an optimal solution to (5.26)–(5.28). If the equality holds, we conclude that the optimal solution of (5.26)–(5.28) can be also reached by $\tilde{\varphi}$ that is a noninferior solution of (5.31).

Proposition 2 enables the search of the optimal solution of problem (5.26)–(5.28) to be confined to the set of noninferior solutions of problem (5.31). If the solution to problem (5.26)–(5.28) is unique, this optimal solution must be a noninferior solution. Note that all J_i, $i = 1, 2, \ldots, k$, are nonnegative. There always exists an integer p greater than or equal to one such that every noninferior solution of (5.31) can be generated by the following weighted pth power Lagrangian form

$$\min\left\{ J_1(\varphi_1)^p + \sum_{i=2}^{k} \lambda_i J_i(\varphi_i)^p \right\} \tag{5.33}$$

subject to constraints (5.27)–(5.28), where all weighting coefficients λ_i, $i = 2, \dots, k$, are nonnegative. Denote the weighting vector by $\lambda = [1, \lambda_2, \dots, \lambda_k]$. In the above weighted pth power Lagrangian formulation, without loss of generality, the pth power of J_1 is chosen as the primal objective with weighting coefficient equal to 1. The pth power of any J_i can be chosen as the primal objective. Similar to the definition in the weighted p-norm method, a noninferior solution of (5.31) is said to be of degree d if it can be found using any weighted pth power Lagrangian formulation with $p \geq d$. The degree of the set of noninferior solutions is further defined as the supremum of the degree of all the noninferior solutions. If $p = 1$, the formulation (5.33) is of a weighting like form. If $p = \infty$, the formulation (5.33) becomes the weighted minimax formulation. We consider only the cases where the degree of the set of noninferior solutions is finite. This weighted pth power Lagrangian formulation is very general in generating noninferior solutions. If problem (5.31) is convex, all noninferior solutions are of degree 1. If the noninferior frontier is nonconvex to some degree, a higher power Lagrangian formulation is needed. In most cases, the degree of the set of noninferior solutions of a given multiobjective problem is difficult to determine. A large enough p needs to be chosen to guarantee the generation of any specific noninferior solution. Let us further examine the existence of supporting hyperplanes on the noninferior frontier in both objective spaces $\{J_1, \dots, J_k\}$ and $\{J_1^p, \dots, J_k^p\}$. For example, consider that φ is two-dimensional, $J_i(\varphi_i) = \varphi_i$, $i = 1, 2$, and the constraint $\varphi_1^2 + \varphi_2^2 \geq 1$. The feasible region in the space $\{J_1, J_2\}$ space for this example problem is $J_1^2 + J_2^2 \geq 1$. It is easy to see that there are no supporting planes on the noninferior frontier, that is, $J_1^2 + J_2^2 = 1$. This shows why the weighting method fails to generate the set of noninferior solutions in such nonconvex situations. If we recast the feasible region in the $\{J_1^4, J_2^4\}$ space, the feasible region becomes $\sqrt{J_1^4} + \sqrt{J_2^4} \geq 1$ and it is convex, where the supporting planes exist at every point on the noninferior frontier. This shows that by selecting p large enough, the supporting hyperplane will exist everywhere on the noninferior frontier. If the supporting hyperplanes exist everywhere on the noninferior frontier in $\{J_1^p, \dots, J_k^p\}$ space, the pth power Lagrangian form can be applied successfully in identifying any noninferior point that reaches the optimum point of problem (5.26)–(5.28). The existence of the supporting hyperplanes guarantees the convergence of the solution scheme proposed in this paper. The problem (5.33) is of a separable structure and is solved using the dynamic programming. If the optimization problem (5.33) is solved for various values of the weigth vector λ, the set of noninferior solutions can be generated and theoretically expressed in the objective space as a parametric form

$$J_i = J_i(\lambda), \quad i = 1, 2, \dots, k \tag{5.34}$$

It is assumed that each $J_i(\lambda)$ is differentiable with respect to λ. With $J_i(\lambda)$, $i = 1, 2, \dots, k$, substituted into (5.26)–(5.28), the overall objective function J becomes a function of λ. From Proposition 2, we know that the solution of problem (5.26)–(5.28) can be attained by a noninferior solution of problem (5.31). The specific solution that reaches the optimum point of (5.26)–(5.28) must satisfy the Kuhn–Tucker conditions

$$\lambda_j \left[\sum_{i=1}^{k} (\partial J / \partial J_i)(\partial J_i / \partial \lambda_j) \right] = 0, \quad j = 2, \ldots, k \tag{5.35}$$

The optimal conditions given in (5.35) are hard to check in the solution process. It is unnecessary to generate the whole set of noninferior solutions in order to search for the optimal solution of the nonseparable problem given in (5.26)–(5.28). Since $\partial J_i / \partial \lambda_j$ is unavailable in the iterative process, we will find an alternative optimal condition which uses only the current point-wise values of J_i's and λ_i's in the iterative process. □

Proposition 3 *If $J_i(\hat{\lambda})$, $i = 1, \ldots, k$, is a noninferior solution generated by a weighted pth power Lagrangian formulation with all values of $\hat{\lambda}_j$, $j = 2, \ldots, k$, strictly positive, the following holds*

$$J_1^{p-1}(\hat{\lambda}) \frac{\partial J_1(\hat{\lambda})}{\partial \lambda_j} + \sum_{i=2}^{k} \hat{\lambda}_i J_i^{p-1}(\hat{\lambda}) \frac{\partial J_i(\hat{\lambda})}{\partial \lambda_j} = 0, \quad j = 2, \ldots, k \tag{5.36}$$

Proof Since $J_1^p(\hat{\lambda}) + \sum_{i=2}^{k} \hat{\lambda}_i J_i^p(\hat{\lambda})$ is the minimum point of the weighted pth power Lagrangian problem of minimizing $J_1^p + \sum_{i=2}^{k} \hat{\lambda}_i J_i^p$, any admissible variation dJ about the point $[J_1(\hat{\lambda}), \ldots, J_k(\hat{\lambda})]$ must satisfy

$$p J_1^{p-1}(\hat{\lambda})\, dJ_1(\hat{\lambda}) + \sum_{i=2}^{k} \hat{\lambda}_i p J_i^{p-1}(\hat{\lambda})\, dJ_i(\hat{\lambda}) \geq 0 \tag{5.37}$$

where

$$dJ_i(\hat{\lambda}) = \sum_{j=2}^{k} \frac{\partial J_i(\hat{\lambda})}{\partial \lambda_j} \Delta\lambda_j + \frac{1}{2} \sum_{j=2}^{k} \sum_{l=2}^{k} \frac{\partial^2 J_i(\hat{\lambda})}{\partial \lambda_j \partial \lambda_l} \Delta\lambda_j \Delta\lambda_l + \cdots \tag{5.38}$$

Since λ_j, $j = 2, \ldots, k$, are assumed to be strictly positive and $\Delta\lambda_j$, $j = 2, \ldots, k$, can thus take any sign, (5.16) must be held to guarantee that the inequality (5.17) is satisfied. □

We conclude from Proposition 3 that in the objective space $\{J_1^p, \ldots, J_k^p\}$, the vector $[1, \lambda_2, \ldots, \lambda_k]$ is orthogonal to the tangent hyperplane S at $\{J_1^p(\lambda), \ldots, J_k^p(\lambda)\}$ which is spanned by

$$\begin{bmatrix} J_1^{p-1} \partial J_1 / \partial \lambda_2 \\ \vdots \\ J_k^{p-1} \partial J_k / \partial \lambda_2 \end{bmatrix}, \begin{bmatrix} J_1^{p-1} \partial J_1 / \partial \lambda_3 \\ \vdots \\ J_k^{p-1} \partial J_k / \partial \lambda_3 \end{bmatrix}, \ldots, \begin{bmatrix} J_1^{p-1} \partial J_1 / \partial \lambda_k \\ \vdots \\ J_k^{p-1} \partial J_k / \partial \lambda_k \end{bmatrix} \tag{5.39}$$

The nondegenerate cases where the optimal solution (5.26)–(5.28) is attained by a solution of (5.33) with all corresponding weighting coefficients strictly positive, will be considered in the following development. In degenerate cases where the optimal solution of (5.26)–(5.28) is attained by a solution of (5.33) with one or more weighting coefficients equal to 0, two or more objective functions J_i, $i = 1, 2, \ldots, k$ in problem (5.31) do not conflict with each other in the neighborhood of the optimal solution of problem (5.26)–(5.28).

5.4.2 Distributed Implementation

Motivated by sustainable and maintainable information retrieval functionality which collects all the QoS related data (e.g., usage utility, capacity, bandwidth, CPU and memory consumption, current numbers of application running), we adopt a hierarchical server-client architecture as shown in Fig. 5.5. When more than one path satisfying the bandwidth demand exists, the information management system helps the selection of the path minimizing the blocking probability of future reconfiguration requests [15]. The connectivity request is processed independently at each network node, based only on the node identifier which is carried in the connectivity request signal. When network resources are not available on such shortest paths, the quality of service degrades. Resource Reservation Protocols allow the signaling and instantiation of the channel trails in a path oriented networking environments [16].

Dynamic reconfiguration can be explicitly or implicitly triggered. In the former, the reconfiguration process is initiated through an invocation to the allocate method, while implicit reconfiguration starts when the reconfiguration controller detects that

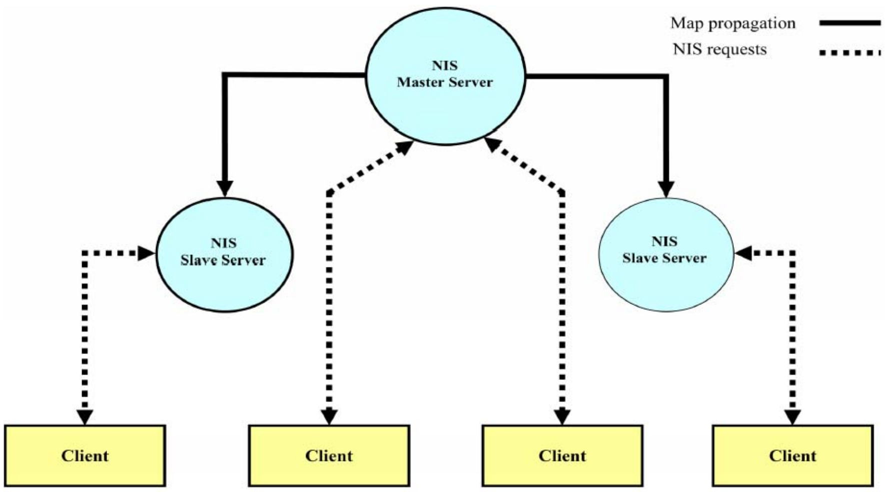


Fig. 5.5 Reconfiguration Controller information patterns

a required network element is not invoked. The invocation sequence for explicit reconfiguration is the following: (1) RC looks up for the requested network element identity in the known object table; (2) Resource Reservation Protocol generates a new configuration request; (3) if the request has a persistent trail, the RC transfers the serialized state from the lookup table to the element; (4) RC activates the network element (start invocation); (5) RC updates the network element in the location service.

Proposition 4 *If the optimal solution of problem* (5.26)–(5.28) *is attained by a noninferior solution with all corresponding weighting coefficients strictly positive*, *the following optimal condition must be satisfied*:

$$\frac{\partial J/J_1}{J_1^{p-1}} = \frac{\partial J/J_2}{\lambda_2 J_2^{p-1}} = \cdots = \frac{\partial J/J_k}{\lambda_k J_k^{p-1}} \tag{5.40}$$

Proof Since the tangent space S at the noninferior solution that attains the optimal point of problem (5.26)–(5.28) is a $(k-1)$-dimensional hyperplane, we know from Proposition 3 that the vector $[1, \lambda_2, \dots, \lambda_k]$ constitutes the basis of a one-dimensional space S^* which is the orthogonal complementary space of S. Equation (5.35) can be rewritten as

$$\lambda_j \left[\sum_{i=1}^{k} \left(\frac{\partial J}{\partial J_i^p} \frac{\partial J_i^p}{\partial J_i} \right) \left(\frac{\partial J_i}{\partial J_i^p} \frac{\partial J_i^p}{\partial \lambda_j} \right) \right] = \lambda_j \left[\sum_{i=1}^{k} \left(\frac{\partial J}{\partial J_i^p} \frac{\partial J_i^p}{\partial \lambda_j} \right) \right] = 0, \quad j = 2, \dots, k \tag{5.41}$$

Note that the vectors $[\partial J_1^p/\partial \lambda_j, \dots, \partial J_k^p/\partial \lambda_j]^T$, $j = 2, \dots, k$, constitute the $(k-1)$-dimensional basis for tangent hyperplane S, we conclude from (5.41) that the optimal solution of problem (5.26)–(5.28) is the vector

$$\left[\partial J/\partial J_1^p, \dots, \partial J/\partial J_k^p\right]^T = \frac{1}{p} \left[\frac{\partial J/\partial J_1}{J_1^{p-1}}, \dots, \frac{\partial J/\partial J_k}{J_k^{p-1}} \right]^T \tag{5.42}$$

belongs to the one-dimensional orthogonal complementary space of the tangent hyperplane S when all weighting coefficients are strictly positive. Equation (5.30) holds since the vectors $[1, \lambda_1, \dots, \lambda_k]$ and $[\partial J/\partial J_1/J_1^{p-1}, \dots, \partial J/\partial J_k/J_k^{p-1}]$ are proportional. □

Proposition 4 provides a necessary condition for implementing the optimal value of weighting vector λ. The weighting vector λ is viewed as the signaling information that occurs among the cooperating agents involved in controlling the network connectivity. Let $J_i(\lambda^t)$, $i = 1, \dots, k$, be the optimal solution generated by the weighting pth power Lagrangian formulation with weighting vector λ^t at iteration t. The gradient of the objective function at point $[J_1^p(\lambda^t), \dots, J_k^p(\lambda^t)]$ in the objective space is given by

$$\nabla J\left(\lambda^t\right) = \left[\partial J/\partial J_1^p, \dots, \partial J/\partial J_k^p\right]_{|\lambda^t}^T = \left[\frac{\partial J/\partial J_1}{p J_1^{p-1}}, \dots, \frac{\partial J/\partial J_k}{p J_k^{p-1}} \right]_{|\lambda^t}^T \tag{5.43}$$

It is known from Proposition 3 that the vector

$$\lambda^t = \left[1, \lambda_2^t, \ldots, \lambda_k^t\right]^T \tag{5.44}$$

is a normal vector on the noninferior frontier at point $[J_1^p(\lambda^t), \ldots, J_k^p(\lambda^t)]$ in the objective space. The projection of the negative gradient $-\nabla J$ on the tangent hyperplane at the current noninferior point can be calculated by

$$\Delta J(\lambda^t) = \left[\Delta J_1^p(\lambda^t), \ldots, \Delta J_k^p(\lambda^t)\right]^T = -\nabla J(\lambda^t) + \Delta J^t(\lambda^t)^T \lambda^t \frac{\lambda^t}{\|\lambda^t\|^2} \tag{5.45}$$

where $\|\lambda^t\|$ is the norm of vector λ^t,

$$\|\lambda^t\|^2 = 1 + \sum_{i=2}^{k} (\lambda_i^t)^2 \tag{5.46}$$

From the Cauchy–Schwarz inequality, we have

$$\nabla J(\lambda^t)^T \nabla J(\lambda^t) = -\|\nabla J(\lambda^t)\|^2 + \|\nabla J(\lambda^t)\lambda^t\|/\|\lambda^t\|^2 \le 0 \tag{5.47}$$

We further need a way to adjust the value of the weighting vector λ in order to achieve the improvement in the objective space $\{J_1^p, \ldots, J_k^p\}$. The tangent hyperplane at point $[J_1^p(\lambda^t), \ldots, J_k^p(\lambda^t)]$ is of dimension $(k-1)$, so only $(k-1)$ of the k terms $\Delta J_i^p(\lambda^t)$ in (5.45) are independent. The following ε-constraint problem can be formulated to realize the descent direction

$$\min J_1^p \tag{5.48}$$

$$\text{subject to} \quad J_j^p \le J_j^p(\lambda^t) + \alpha \Delta J_j^p(\lambda^t), \quad j = 2, \ldots, k \tag{5.49}$$

where α is a step size parameter which can be adjusted on line to guarantee a decrement of the overall objective function J. The dual function of (5.48)–(5.49) is

$$H(\lambda) = \min J_1^p + \sum_{j=2}^{k} \lambda_j \left[J_j^p - J_j^p(\lambda^t) - \alpha \Delta J_j^p(\lambda^t)\right] \tag{5.50}$$

subject to (5.47) and (5.48). The descent direction $[\Delta J_1^p(\lambda^t), \ldots, \Delta J_k^p(\lambda^t)]$ at iteration $t+1$, needs the new values of λ_i, $i = 1, \ldots, k$, which can be obtained through maximizing the dual function with respect to λ at current noninferior point $[J_1^p(\lambda^t), \ldots, J_k^p(\lambda^t)]$. The derivative of the function $H(\lambda)$ with respect to λ_j at the current point $[J_1^p(\lambda^t), \ldots, J_k^p(\lambda^t)]$ is

$$\frac{\partial H(\lambda)}{\partial \lambda_j} = -\alpha \Delta J_j^p(\lambda^t), \quad j = 2, \ldots, k \tag{5.51}$$

The value of λ is the gradient control process algorithm

$$\lambda_j^{t+1} = \lambda_j^t + \frac{\partial H(\lambda)}{\partial \lambda_j} = \lambda_j^t - \alpha \Delta J_j^p(\lambda^t), \quad j = 2, \ldots, k \tag{5.52}$$

The dual function $H(\lambda^{t+1})$ in (5.50) is equivalent to the following weighted pth power Lagrangian problem for a selected λ^{t+1}

$$\min J_1^p + \sum_{j=2}^{k} \lambda_j^{t+1} J_j^p \tag{5.53}$$

This sets the weighted pth power Lagrangian formulation with an updated value of λ at iteration $t + 1$ for the lower level.

The main purpose of the system-level reconfiguration is to provide transparency (for location, access, transport mechanism). Location transparency allows clients to invoke service requests without any prior knowledge of the real location of the service. This is of special interest in dynamic reconfiguration environments, since the reference (address assigned in the network information map) of a reconfigurable component can be deferred to the instantiation inside a certain reconfigurable area, at runtime. The location service, then is in charge of providing a valid endpoint (the address where the instantiated component is located) when a client requests the location of concrete service [17].

Two entities are involved in the location process. The proxy stores a reference for the locator object, instead of a static endpoint to the server hardcoded at design time. On the other side, the location service (or directory service), provides the valid endpoint from the requested networked element identity. When an invocation is received from the client network element, the proxy will first request the current location of the remote element (using the object identifier). The location service contains a location table where object identities are linked with valid endpoints. Once the location is obtained, the networked element will perform a second request: the real invocation. However, the indirection doesn't necessarily imply two requests per invocation. That can be easily avoided simply caching the obtained location. The interface for the locator object has two kind of methods. The locate method provides the location functionality previously described [18].

Another problem in the dynamically reconfigurable environments is that the need for information processing may be difficult to predict at design time. Bitstreams for new networked element types may be deployed at any time, and extra unpredictable space is also needed for state storage of object instances with persistence capabilities. The solution is to define a dynamic memory allocation service that will transparently provide basic primitives for information management. The service is provided by an specialized object called the Allocator. The Allocator has two main characteristics. On one side it centralizes information management for the whole system. On the other side it offers a well-known interface, completely independent from the implementation technology or information hierarchy. The service interface is based on two methods. The allocate method requests the allocation of a certain memory block size, while the release method frees the information block. The return value for the allocate method is a proxy to network information map.

5.4.3 Summary of Computational Results

In this section, we briefly discuss sample results for the topology information control framework applied to self-healing UAV formations dedicated to surveillance missions. Two double tree networking topologies are represented in Fig. 5.6 as start configurations on which the information control and coding schemes previously described are applied. Figure 5.7 shows the corresponding converged topologies after the reconfiguration process was terminated. In the group communication networking, the decrease of the constraints number and the use of the intrinsic parallelism

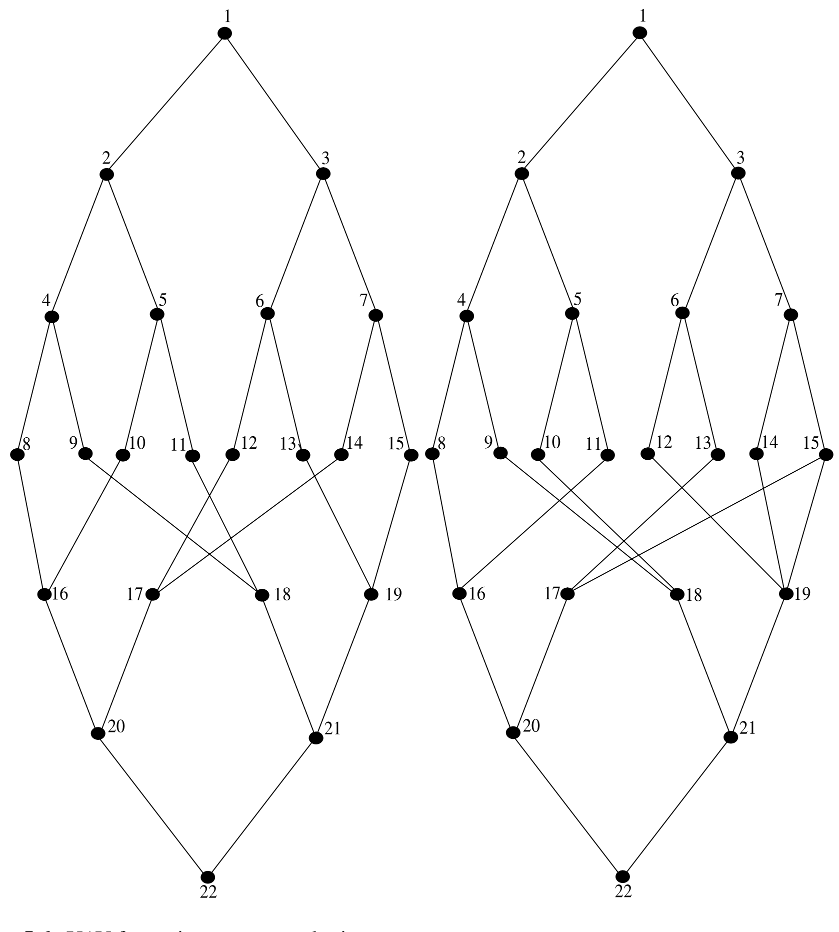


Fig. 5.6 UAV formations: start topologies

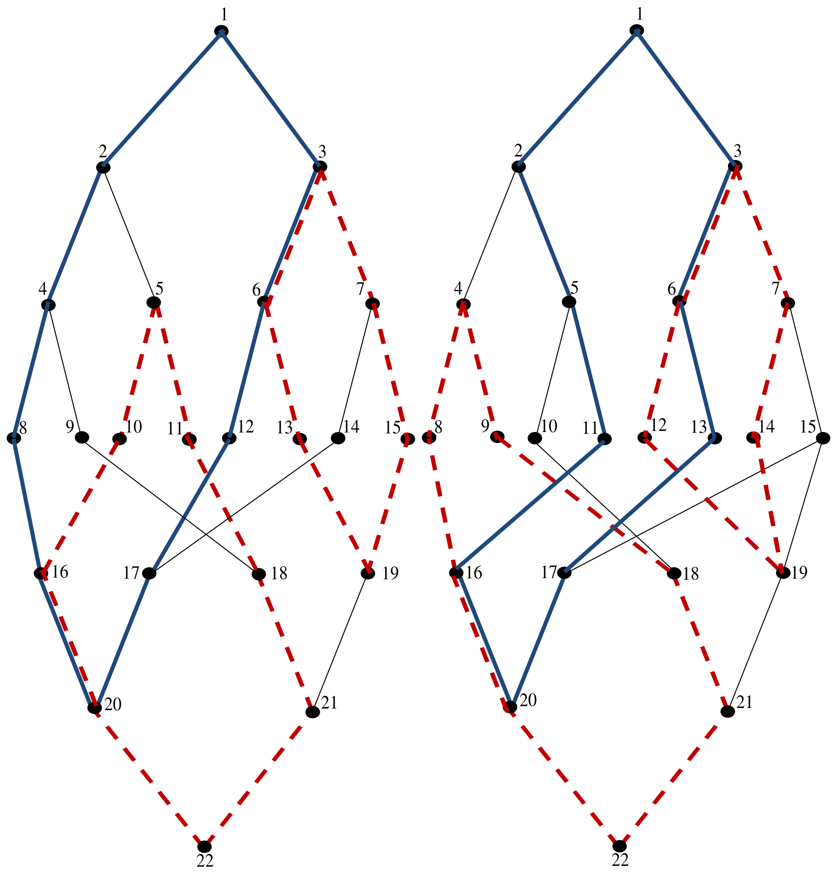


Fig. 5.7 UAV formations: converged topologies

of the signalling channels, the data control processing is based on coding of coarse-grain control symbols. Each networking element has a local register file, filled from external caches of control data, global for the whole network. The processing is applied on larger data sets and the control is distributed. Reconfigurable network architectures face the problem of keeping under control the communication delays in reconfigurable devices designed in advance technologies. Together with algorithmic and technological evolutions, reconfigurable architectures have to face another challenge related to their communication energy efficiency. Fixed architectures can not address this kind of energy optimizations when various services or accuracy constraints are concerned. Reconfigurable architectures can be specifically configured depending on the particular piece of code fragment to be implemented which leads to interesting group communication behaviors.

The dependency of information control savings on the kth modeling order is shown in Fig. 5.8. Design flow consists first in selecting the applicative domain, and defining the kind of computations that are typically found into the loop kernels and

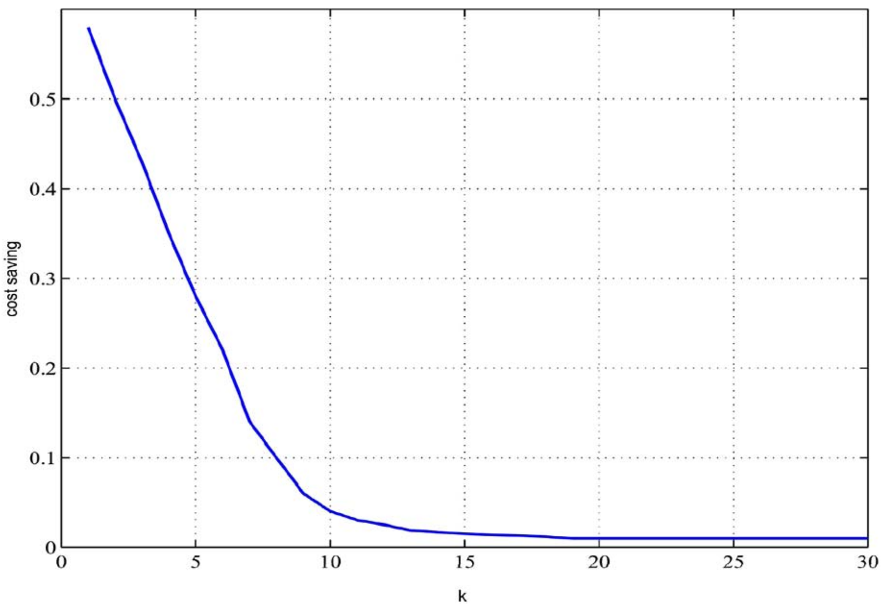


Fig. 5.8 Information control savings versus order *k*

frequently executed code fragments. Creating a bench representative of the targeted application domain can greatly facilitate the design space exploration by further refinements of the architecture. Selecting operations at a too low level, such as sum of absolute differences in image and video processing, or selecting an insufficient number of functions for the bench will lead to optimize only the peak performance of the architecture, but this will prove to be not representative enough of a real applicative behavior, in particular because control statements have a great impact on the overall performance.

The fundamental objective in topology information control is to maximize the throughput carried in a networked environment. The closed loop performance was studied computationally on sample networks that are subject to various traffic matrices. The goal was to assess the performance levels of reconfiguration processes in dynamic networking environments.

5.5 Concluding Remarks

In this paper, we have introduced a mathematical framework for information control and coding occurring in reconfiguration processes in systems of systems. As a feedback control mechanism, the iterative parametric dynamic programming is a good modeling candidate in describing the reconfiguration computations, where a multi-objective relational optimization problem is considered in enabling a separation strategy and finding the optimal solution in a multilevel fashion. At the lower processing level, an auxiliary weighted power Lagrangian problem is solved using dynamic programming associated to the topology control. The iterative parametric dynamic programming for a class of nonseparable optimization problems approach extends the reach of dynamic programming and provides an efficient solution scheme through separation, convexification, decomposition, and coordination. The insights offered by such modeling are useful in deriving resource management protocols allowing the path selection and QoS parameters monitoring to be taken into account by the system reconfiguration mechanism. Distributed topology information control among the cooperative network elements regarded as agents broadcasting periodical signals to enable the network connectivity information propagation.

Future work will continue to characterize wireless communication networks and the information control techniques for guaranteeing the mission performance and safety of the networked UAVs. In this consideration, communications will be key and will require further advances based on the results described in this paper. Multivariable extremum seeking control can be applied to the problem of UAV motion control in a communication environment, and such approach should lead to improvements in the communication ability over position-based policies.

References

1. Elston, J., Frew, E.W., Argrow, B.: Networked UAV communication, command, and control. In: Proc. AIAA Guidance, Navig., Contr. Conf. (2006)
2. Beard, R., McLain, T., Nelson, D., Kingston, D., Johanson, D.: Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs. In: Proc. IEEE, vol. 94, no. 7, pp. 1306–1324 (2006)
3. Rezano, J., Mozos, D., Catthoor, F., Verkest, D.: A reconfiguration manager for dynamically reconfigurable hardware. In: IEEE Design and Test of Computers, pp. 452–460 (2005)
4. Ross, K.W.: Multiservice Loss Models for Broadband Telecommunication Networks. Springer, London (1995)
5. Even, S., Tarjan, E.: Network flow and testing graph connectivity. SIAM J. Comput. **4**, 507–518 (1975)
6. IEEE standard for local and metropolitan area networks: virtual bridged local area networks. IEEE Std 802.1Q-1998
7. IEEE standard for local and metropolitan area networks—common specifications. Part 3: Media access control (MAC) bridges—amendment 2: Rapid reconfiguration amendment to IEEE Std 802.1D, 1998 Edition. IEEE Std 802.1w-2001
8. Ahlswede, R., Cai, N., Li, S.-Y., Yeung, R.W.: Network information flow. IEEE Trans. Inf. Theory **46**(4), 1204–1216 (2000)
9. Cover, T.M., Thomas, J.: Elements of Information Theory. Wiley, New York (1991)
10. Berger, T.: The Information Theory Approach to Communications. Springer, Berlin (1978)
11. Even, G., Even, S.: Embedding interconnection networks in grid via the layered cross product. Networks **36**(2), 91–95 (2000)
12. Duato, J., Yalamanchili, S., Ni, L.: Interconnection Networks: An Engineering Approach. IEEE Comput. Soc., Silver Spring (1997)
13. Bertsekas, D.: Nonlinear Programming. Athena Scientific, Belmont (1995)
14. Fletcher, R.: Practical Method of Optimization. Wiley, New York (1987)
15. Wolinski, C., Kuchcinski, K.: Automatic selection of application-specific reconfigurable processor extensions. In: DATE 2008: Proc. of the Conf. on Design, Automation and Test in Europe, pp. 1214–1219 (2008)

16. Henriksen, S.J.: Estimation of future communications bandwidth requirements for unmanned aircraft systems operating in the national airspace system. In: Proc. AIAA InfoTech@Aerospace, vol. 3, pp. 2746–2754 (2007)
17. Dixon, C., Frew, E.W.: Decentralized extremum-seeking control of nonholonomic vehicles to form a communication chain. In: M.J. Hirsch, P.M. Pardalos, R. Murphey, D. Grundel (eds.), Advances in Cooperative Control and Optimization. Lecture Notes in Control and Information Sciences, vol. 369, pp. 311–322 (2007)
18. Vasseur, J.-P., Pickavet, M., Demeester, P.: Network Recovery, Protection and Restoration of Optical, SONET-SDH, IP, and MPLS. Morgan Kaufman, Amsterdam (2004)

Chapter 6
Effect of Network Geometry and Interference on Consensus in Wireless Networks

Sundaram Vanka, Vijay Gupta, and Martin Haenggi

Summary We study the convergence of the average consensus algorithm in wireless networks in the presence of interference. It is well known that convergence of the consensus algorithm improves with network connectivity. However, from a networking standpoint, highly connected wireless networks may have lower throughput because of increased interference. This raises an interesting question: what is the effect of increased network connectivity on the convergence of the consensus algorithm, given that this connectivity comes at the cost of lower network throughput? We address this issue for two types of networks: regular lattices with periodic boundary conditions, and a hierarchical network where a backbone of nodes arranged as a regular lattice supports a collection of randomly placed nodes. We characterize the properties of an optimal Time Division Multiple Access (TDMA) protocol that maximizes the speed of convergence on these networks, and provide analytical upper and lower bounds for the achievable convergence rate. Our results show that in an interference-limited scenario the fastest converging interconnection topology for the consensus algorithm crucially depends on the geometry of node placement. In particular, we prove that asymptotically in the number of nodes, forming long-range interconnections improves the convergence rate in one-dimensional tori, while it has the opposite effect in higher dimensions.

6.1 Introduction

Consensus has become an area of increasing research focus in recent years (e.g., see [1–8] and the references therein). Given n nodes each with a scalar value and a possibly time-varying interconnection graph defined on these nodes, a consensus

S. Vanka (✉) · V. Gupta · M. Haenggi
Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA
e-mail: svanka@nd.edu

V. Gupta
e-mail: vgupta2@nd.edu

M. Haenggi
e-mail: mhaenggi@nd.edu

algorithm specifies the updating rule that every node should follow. The updated value of each node at every time step is a function of the value held by itself and its neighbors at the previous time step. The conditions on graph connectivity that permit convergence to a common value have been fairly well-characterized. The focus has now shifted to analyzing the convergence properties in the face of communication constraints such as quantization [9], packet erasures [2, 10], additive channel noise [11, 12], and delays [13].

Such works typically assume that the communication channels between each pair of nodes are uncoupled. However, consensus algorithms are often employed over wireless networks, where channels are inherently coupled due to their broadcast nature and the presence of interference. Long-range interconnections lead to a smaller graph diameter, but also to decreased spatial re-use. The effect of long-range interconnections on the rate of convergence of the consensus algorithm is, thus, not clear. Moreover, in a wireless network, a communication graph cannot be considered to be given a priori, since any two nodes can communicate by spending enough energy. The communication topology in wireless networks thus depends on the network protocols and is, in fact, a design parameter.

In this work, we consider the rate of convergence of the average consensus algorithm while explicitly accounting for interference. We analyze the performance of scheduling algorithms that are optimal with respect to the rate of convergence. We also provide an analytical understanding of the impact of transmission power, and thus communication topology, on the rate of convergence.

This article is organized as follows. We begin in Sect. 6.2 by formulating the problem and introducing our notation. We concentrate on two geographical placements of the nodes: (i) nodes that are physically placed on a grid with periodic boundary conditions (considered in Sect. 6.3), and (ii) hierarchical networks with randomly placed sensor nodes and a regular communication backbone (considered in Sect. 6.4). Some avenues for future work are presented in Sect. 6.5.

6.2 Problem Formulation

Average Consensus Algorithm Consider n nodes that aim to reach consensus with the final value being the average of their initial scalar values. Denote the value held by the ith node at time k as $x_i(k)$. Also denote by $x(k)$ the n-dimensional vector obtained by stacking the values of all the nodes in a column vector. Let the nodes be connected according to a given interconnection topology at every iteration step. The topology can be described by a *consensus graph*, with an edge present between two nodes if and only if they can exchange information. Denote the neighbor set of node i at time k by $\mathcal{N}_i(k)$. An iteration consists of every node i exchanging its state variable $x_i(k)$ with all nodes in $\mathcal{N}_i(k)$. In the standard description of the consensus algorithm, this exchange of information is assumed to happen in a single packet transmission interval (also referred to as a *time slot* and normalized to 1). Then, the value of each node i evolves as

$$x_i(k+1) = x_i(k) - h \sum_{j \in \mathcal{N}_i(k)} \big(x_i(k) - x_j(k)\big) \tag{6.1}$$

where h is a scalar constant designed to ensure convergence of the algorithm.

Denote the interconnection graph at time k by $\mathcal{G}(k)$. The system thus evolves according to the discrete-time equation

$$x(k+1) = \big(I - hL(k)\big)x(k), \quad x(0) = x_0 \tag{6.2}$$

where $L(k)$ denotes the Laplacian matrix of the graph $\mathcal{G}(k)$. It can be shown (see, e.g., [5]) that under proper connectivity assumptions, if the parameter h is small enough, consensus is achieved with each node assuming the average value $x_{av} = \frac{1}{n}\sum_i x_i(0)$. Throughout our presentation, we will assume that h is fixed and has a value $h < \frac{1}{2d_{\max}}$ where $d_{\max}$ is the maximum degree corresponding to any node in the consensus graph over all time, i.e., $d_{\max} = \max_{i,k} |\mathcal{N}_i(k)|$. To ensure that the nodes converge to the average of x_0, it is also essential that the graph at every time step be balanced, i.e., at every node the in-degree equals the out-degree. The protocols we consider below will ensure that the graph is undirected, which trivially satisfies this condition.

The rate of convergence of the value of the nodes is a function of the graph topology. In the case of a static graph topology (i.e., $\mathcal{G}(k) = \mathcal{G}$ for all time k), it can be shown (see, e.g., [5, 14, 15]) that the convergence of the consensus protocol is geometric, with the rate being governed by the second largest eigenvalue modulus (SLEM) of the matrix $I - hL$. In general, a consensus algorithm on a graph with smaller SLEM converges more quickly.

However, in practice, a number of transmissions, each occupying a single time slot, may be necessary for this information exchange among the nodes to occur. This scenario is common in wireless networks, where concurrent transmissions in the same frequency band can interfere at a node, and hence, may not be decodable. Therefore, if each node can receive data from at most one neighbor at any given time, the exchange of information necessary for iteration k will require at least $1 + \max_i |\mathcal{N}_i(k)|$ time slots. This idea is developed further in this article.

Communication Protocols We consider a situation in which the physical locations of the nodes are given in a d-dimensional space. The communication model is along the lines of the disk connectivity graph model considered, e.g., in [16]. Every node then decides on the power with which it transmits. This power determines the communication radius of the node according to the relation

$$P = P_0 r_{\mathrm{c}}^{\alpha}$$

where P_0 is a normalization constant, α is the path-loss exponent (typically $2 \le \alpha \le 5$), P is the transmission power and r_{c} is the communication radius. All nodes at a distance smaller than r_{c} from the transmitter can receive the transmitted message.

We assume a Manhattan connectivity model, in which communication is possible only along axial directions. Such a model is suitable for a situation when only line of sight (LOS) communication is possible. In 2-dimensions, this is a good model for urban environments where the presence of buildings inhibits most nonline of sight links. In an 1-dimensional node arrangement, this model is identical to other connectivity models such as those based on communication disks centered at the nodes.

Similar to the communication radius, we can also define an interference radius r_{i}. A node at position x can receive a message successfully from a node at position y only if $\|y - x\| < r_{\mathrm{c}}$, and there is no node at position z that is simultaneously transmitting, such that $\|z - x\| < r_{\mathrm{i}}$ (interference constraint). The above equations should also be interpreted in the framework of the Manhattan connectivity model, i.e., the distances should be measured only along the axial directions. In this article, for simplicity, we assume $r_{\mathrm{c}} = r_{\mathrm{i}}$. The results can be generalized to other cases.

Given the above condition for successful transmission, we require a medium access control (MAC) protocol for the nodes. We focus on Time Division Multiple Access (TDMA)-based MAC protocols in this article rather than random access protocols. Such protocols assure successful communication by scheduling transmissions in time such that messages do not interfere. They demonstrate better throughput than collision-based MAC protocols, at the expense of greater synchronization and coordination requirements among the nodes [17, 18].

Problem Formulation The operation of the average consensus protocol can be divided into two phases that are repeated at every update of the node values. In the first phase, the nodes exchange their values through possibly multiple transmissions. We consider each transmission to consume one time slot. The effective communication graph at each update is composed of edges (i, j) such that node j has received the value of node i during the previous communication phase. In the second phase, the nodes update their values according to (6.1). As in the standard model, this step is assumed to be instantaneous. Therefore, due to multiple transmissions to set up the consensus graph, in our model, the state update does not occur at every time slot. In fact, assuming that each communication phase is completed in T time slots, the kth update can be expressed as

$$x(kT + T) = (I - hL)x(kT) \tag{6.3}$$

Therefore, the effect of finite communication time, possibly due to interference, is to slow down the convergence rate.

We are interested in the following problem: *Given a set of nodes at known locations, what is the effect of increasing the transmit power on the convergence rate of the consensus algorithm when the channel-access mechanism accounts for interference?* In this context, we characterize the convergence of the consensus algorithm for the optimal MAC protocol that minimizes the number of time slots needed for communication in order to form a desired consensus graph $\mathcal{G}$ (thus, maximizing the update rate). We analyze this problem for two physical distributions of the nodes on a torus:

1. A regular grid of sensor nodes.
2. A regular grid of nodes that form a backbone communication network for sensor nodes that are distributed as a binomial point process.

The transmit power at each node determines its neighbors. The periodic boundary condition is chosen for analytical tractability; the analysis becomes accurate also for the case without periodic boundaries as the number of nodes becomes large.

For analytical tractability, we make the following assumptions:

- We assume equal transmission power for all nodes.
- We limit the transmission policy to be time-invariant. Thus, we assume that the same effective communication graph is generated for every iteration of the node values.
- At the time of an update of the values of the nodes, we require that the effective communication graph be undirected, i.e., for any two nodes i, j in the network, $j \in \mathcal{N}_i \Leftrightarrow i \in \mathcal{N}_j$. Note that this is slightly stronger than the necessary and sufficient condition for convergence of the average consensus algorithm that the graph be balanced [5].
- We assume no explicit routing of values through nodes since the consensus algorithm itself incorporates implicit routing and in-network computation.
- We assume half-duplex operation, and that packets that suffer collisions cannot be decoded.

Under these assumptions, the following are the main results of the article:

1. We characterize the rate of convergence for the optimal MAC scheduling protocol for the average consensus algorithm for tori in n dimensions.
2. We show that the network geometry plays a key role in identifying the optimum power allocation that maximizes the speed of convergence. In particular, while the convergence rate increases with the transmission power in 1-dimensional tori, the opposite is true in higher dimensions.
3. In hierarchical networks, we show that a positive fraction of nodes can always achieve consensus for certain scalings of backbone node density.

In the next section, we begin by studying the convergence properties of MAC protocols that maximize the speed of convergence for a given consensus graph $\mathcal{G}$. We begin by considering nodes placed on a regular grid with periodic boundary conditions.

6.3 Analysis of a Ring and a 2D Torus

6.3.1 The 1-D Case: Nodes on a Ring

Consider n nodes numbered $\{0, 1, \ldots, n-1\}$ placed uniformly on a circle of radius r centered at the origin, as shown in Fig. 6.1. Suppose that the transmission power is such that every node can transmit information to m of its nearest neighbors on

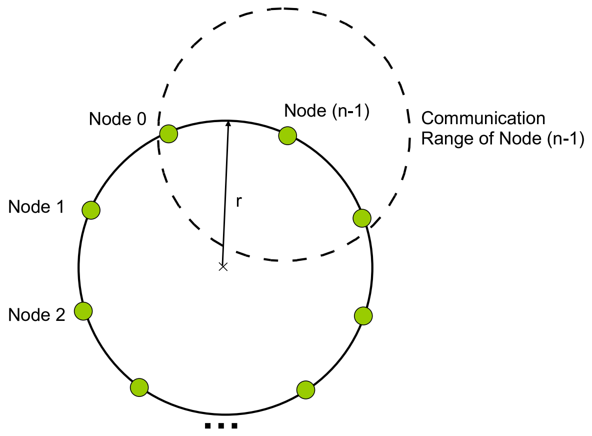


Fig. 6.1 Schematic of nodes placed along a ring

either side. As an example, in Fig. 6.1, $m = 1$. Define P_m, $m \leq \lfloor \frac{n}{2} \rfloor$, as the transmit power that provides a communication radius $r_c = 2r\sin(\frac{m\pi}{n})$. Hence,

$$P_m \propto \left(2r\sin\left(\frac{m\pi}{n}\right)\right)^{\alpha} \tag{6.4}$$

where $\alpha \geq 2$ is the path-loss exponent. As stated above, for simplicity, we will assume that the interference radius $r_i = r_c$.

We note here that an alternative interpretation of this geometry of node placement is to consider the n nodes placed on a regular one-dimensional torus [0, 1] (hereafter, called a "1-torus" or $\mathcal{T}_1(n)$). This interpretation is useful when connecting these results with those for higher dimensional tori, which are discussed later. In this case, if the first node is placed at the origin, the position of the kth node is given by $\frac{k}{n}$, $0 \leq k \leq n-1$, with a periodic boundary condition. In this geometry, the expression for P_m would be $P_m \propto (\frac{m}{n})^{\alpha}$.

If the wireless channel could support simultaneous transmissions by every node, the system would evolve according to (6.2), with $I - hL$ being an $n \times n$ circulant matrix with the first row given by

$$[1 - 2mh\ \mathbf{1}_m^*\ 0\ 0\ \cdots\ 0\ \mathbf{1}_m^*]$$

where

$$\mathbf{1}_m^* = [1\ 1\ \cdots\ 1]_{1\times m}$$

For future reference, denote by $\mathcal{G}_{1,m}$, $L_{1,m}$ and $F_{1,m}$ the consensus graph, the Laplacian and the update matrix, respectively, for such a situation. Given the nodes placed on a ring, $\mathcal{G}_{1,m}$ is the consensus graph with the highest connectivity that can be formed for a given P_m, and therefore will have the fastest convergence. The MAC protocol that we propose guarantees that the system evolves according to this matrix. However, the communication phase occurs over multiple steps.

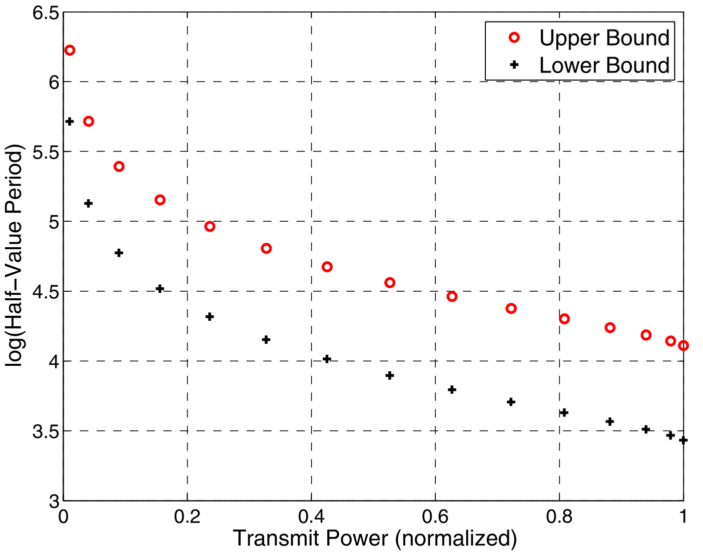


Fig. 6.2 Variation of the convergence rate with the transmission power for a ring of $n = 31$ nodes

We begin by bounding the number of time slots required to form $\mathcal{G}_{1,m}$. Denote the smallest number of time slots used to form $\mathcal{G}_{1,m}$ by $T_1^*(m)$, or, more compactly, as T_1^*. Also denote by $\mathcal{P}^*(m)$ the optimal TDMA protocol that forms the graph $\mathcal{G}_{1,m}$ in T_1^* number of steps. Observe that for all $m_1 \leq m_2$, $\mathcal{G}_{1,m_1} \subseteq \mathcal{G}_{1,m_2}$. Then $\mathcal{G}_{1,m}$ can always be formed in at most $T_1^*(m_1) \leq T_1^*(m_2)$ slots. This implies that $m_1 \leq m_2 \implies T_1^*(m_1) \leq T_1^*(m_2)$. We say a *link* is formed from node v to node u whenever the message from v is successfully decoded at u. Since $\mathcal{G}_{1,m}$ is undirected and balanced, an edge $e \in \mathcal{E}_{1,m}$ connecting v and u is formed if and only if both v and u form links with each other. The following result bounds the length of the shortest TDMA schedule that forms the consensus graph $\mathcal{G}_{1,m}$ (and hence the smallest time T in the update equation (6.3)) and was proven in [19].

Lemma 1 (From [19]) *Consider the set-up described above, where the consensus graph $\mathcal{G}_{1,m}$ is to be formed in the smallest number of time slots. The optimal TDMA protocol forms $\mathcal{G}_{1,m}$ in the smallest possible number of time slots T_1^* where*

$$2m + 1 \leq T_1^*(m) \leq 4m + 1$$

Using this result in conjunction with the spectral properties of $\mathcal{G}_{1,m}$ yields the following characterization of the fastest convergence rate that is possible for a given $\mathcal{G}_{1,m}$.

Theorem 1 (From [19]) *Consider the problem set-up described above. If the optimal TDMA protocol is used to construct $\mathcal{G}_{1,m}$ for each iteration, the error vector $\epsilon(k) = x(k) - \mathbf{1}_n x_{av}$ converges geometrically to zero with the rate of decay β*

bounded as

$$\rho_1^{\frac{1}{2m+1}} \le \beta \le \rho_1^{\frac{1}{4m+1}}$$

where

$$\begin{aligned} \rho_1 &= 1 - h(2m+1) + hS_1^{(m,n)} \\ S_p^{(m,n)} &= \frac{\sin(\frac{(2m+1)\pi p}{n})}{\sin(\frac{\pi p}{n})}, \quad p = 0, 1, \dots n-1 \end{aligned} \tag{6.5}$$

Remarks

1. For any given transmission power P_m, we see that the MAC constraints reduce the rate by a factor of T where $2m+1 \le T \le 4m+1$.
2. The speed of convergence is an increasing function in m, and hence in P_m. An illustration of this fact is provided in Fig. 6.2. For the purpose of the plot, we show the time taken for the error norm to become half, termed the "half-value period", as a function of transmission power for 31 nodes arranged regularly on a ring of radius 1 unit. We have assumed $\alpha = 2$ and the constant of proportionality in (6.4) to be unity. For each P_m, we chose $h \propto \frac{1}{2m+1}$. The results are somewhat counter-intuitive since the rate reduction due to a larger number of steps in the communication phase is always compensated by the increase in rate due to higher connectivity. That forming long range communication links would lead to faster convergence even in networks with interference was not evident a priori.
3. The effect of increasing the transmission power are the most prominent at small P_m. This can again be seen from Fig. 6.2. If $\theta = p\pi/n$ and $p \ll n$,

$$\sin\theta \approx \theta - \theta^3/3 \tag{6.6}$$

We use (6.6) to express the spectral gap $\mathrm{SG} \triangleq 1 - \rho_1^{\frac{1}{T}}$ when $m \ll n$ as

$$\begin{aligned} \mathrm{SG} &= 1 - \left(1 - h(2m+1) + h\frac{\sin(\frac{(2m+1)\pi}{n})}{\sin(\frac{\pi}{n})}\right)^{\frac{1}{T}} \\ &\approx \eta m h(m+1)(2m+1)T^{-1}n^{-2} \end{aligned}$$

where $\eta \triangleq \frac{4\pi^2}{3}$. Since $h \propto \frac{1}{2m+1}$ and $T = \Theta(m)$ for the optimal schedule, the spectral gap scales as $\frac{m}{n^2}$.

6.3.2 Nodes on a Two-Dimensional Torus

We now generalize our results to higher dimensional tori. Consider a set $\mathcal{T}_d(n)$ of $n = l^d$ regularly spaced points on a d-dimensional torus located at $[0, 1]^d$. An example when $d = 2$ is shown in Fig. 6.3. Choose a node as the origin, and label each

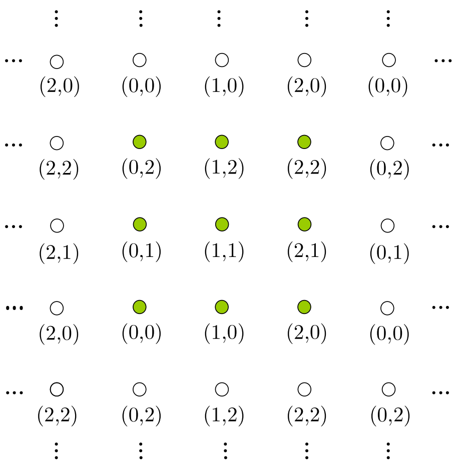


Fig. 6.3 The toroidal lattice $\mathcal{T}_2(9)$. The *shaded* nodes indicate those physically placed in $[0, 1]^2$. Node $(0, 0)$ represents the node at the origin, with each of the *shaded* nodes (i, j) being placed at $(i/3, j/3)$. Nodes that are left unfilled are the image nodes that arise due to the periodic boundary condition

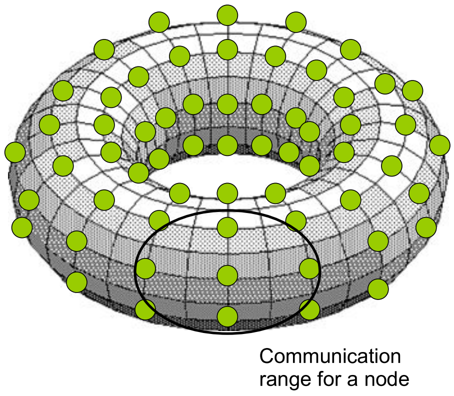


Fig. 6.4 Schematic of nodes placed along a 2-dimensional torus. The periodic square grid being considered can be considered a limiting case of a large torus, so that the effect of its curvature on small distances is not important

node using its displacements along each of the d axes (referred to as the *d axial directions* of the torus in this article). For example, in Fig. 6.3, $n = 9$ and the node $(1, 1)$ is located at $r_{11} \equiv r_{(1,1)} \triangleq (\frac{1}{3}, \frac{1}{3})$. An alternative interpretation of a toroidal arrangement in the two-dimensional case is shown in Fig. 6.4. Both these interpretations yield similar results in the limiting case of a large torus in which case local distances are not significantly affected by the curvature. We will focus on the former interpretation.

Suppose all nodes on a torus $\mathcal{T}_d(n)$ participate in an average consensus algorithm of the form (6.2) with a power allocation of P_m per node. The results for a disk connectivity model were provided in [19]. Here, we present results for the Manhattan connectivity model. Figure 6.5 describes the connectivity model for a set of transmitting nodes on a two-dimensional torus of $n = 25$ nodes and $m = 1$.

We now formally define the desired consensus graph $\mathcal{G}_{d,m} = (\mathcal{V}, \mathcal{E}_{d,m})$. The vertex set

$$\mathcal{V} = \{0, 1, \ldots, l - 1\}^d$$

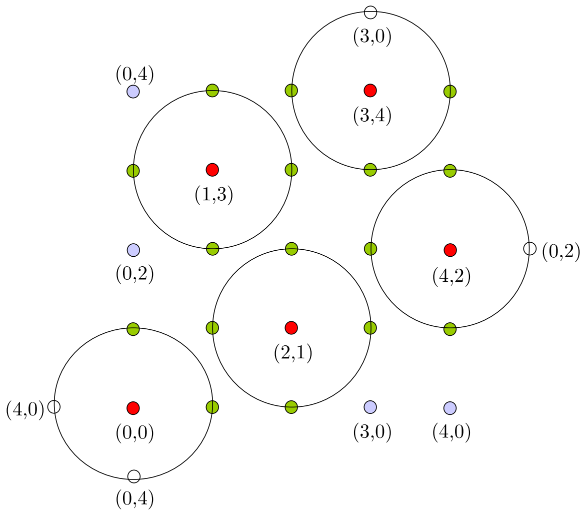


Fig. 6.5 Effect of network geometry on choosing the transmitter set, shown for $\mathcal{T}_2(25)$ and $m = 1$. The transmitters are at the center of each disk (e.g., the node labeled (0,0)). Note that the dimensionality is exploited to allow more concurrent transmissions. The nodes that do not lie in any of the disks (e.g., (4,0)) are covered by transmitters via their images (that have the same label)

is the set of all points in $\mathcal{T}_2(n)$. We note that if each node can transmit at power P_m, the Manhattan connectivity model places edges between two distinct nodes v and u if and only if the following conditions are satisfied:

1. The locations of v and u must differ in at most one coordinate; and
2. At the co-ordinate where they differ, the absolute difference between the respective values can be at most m/l

Denote the set of all these edges by $\mathcal{E}_{d,m}$.

In keeping with the notation developed for the one-dimensional case, we will denote the Laplacian and the update matrix for $\mathcal{G}_{d,m}$ by $L_{d,m}$ and $F_{d,m} \triangleq I - hL_{d,m}$, respectively. Assuming as before that each transmission occupies one time slot, we now study the convergence properties of the optimal MAC protocol that will form $\mathcal{G}_{d,m}$ in the smallest number of time slots. In this article, we set $d = 2$; the results can be generalized to tori of higher dimensions.

Using similar arguments as in the one-dimensional case, we can show that the optimal TDMA protocol assigns an equal power P_m to each chosen transmitting node i in any time slot. Denote by $T_2^*(m)$, or more compactly by T_2^* the number of time slots required by an optimal schedule to construct $\mathcal{G}_{2,m}$. The optimal MAC schedule places the *maximum* number $N_2^*(t)$ of non-interfering transmitters on the torus in every time slot $t = 1, 2, \ldots, T_2^*$.

Note that increased network dimensionality plays an important role in constraining N_2^* (and consequently, T_2^*). This makes the problem of analytically finding T_2^* nontrivial. This effect is illustrated in Fig. 6.5 for a 2-torus of $n = 25$ nodes and $m = 1$. The transmitters are chosen from the entire two-dimensional lattice. With power P_1, each node can reach its 4 nearest neighbors as shown. Since every node must transmit at least once, at least $4 + 1 = 5$ slots are necessary to form the consensus graph $\mathcal{G}_{2,1}$. Figure 6.5 shows the optimal transmitting set in the first time slot.

As before, we begin by characterizing the length T_2^* of the shortest TDMA schedule that constructs $\mathcal{G}_{2,m}$. Thereafter, we exploit the properties of the consensus algorithm along with the optimality of the MAC protocol and the constraints imposed by our problem to bound the convergence rates for a 2-torus.

As before, define P_m to be the transmit power that enables a node to form error-free links with m neighbors in the axial directions. Given that there are $\sqrt{n}$ nodes in either of the axial directions,

$$P_m \propto \left(\frac{m}{\sqrt{n}}\right)^{\alpha}$$

We now define the update matrix $F_{2,m}$. Define circulant matrix Q_m to be a circulant matrix with the first row $[1-4mh\ h\mathbf{1}_m^*\ 0\ \cdots\ 0\ h\mathbf{1}_m^*]_{1\times l}$. Further, define $R_m \triangleq h\mathbf{1}_m^* \otimes I = h[I\ I\ \cdots\ I]_{l\times lm}$.

If each node uses power P_m, the Manhattan connectivity model results in a block circulant update matrix $F_{2,m}$ with its first row being

$$[Q_m\ R_m\ 0\ \cdots\ 0\ R_m]_{l\times n} \tag{6.7}$$

We now bound the number of time slots required to form $\mathcal{G}_{2,m}$.

Lemma 2 *If each node transmits at power P_m and the optimal schedule over the 2-torus constructs $\mathcal{G}_{2,m}$ in T_2^* time slots, for $1 < m < \lfloor l/2 \rfloor$, T_2^* always satisfies*

$$T_l \le T_2^* \le T_u$$

where

$$T_l = m^2 + 2m + 2$$

and

$$T_u = 16m^2 + 8m + 1$$

Proof Using similar arguments as in Lemma 1, it is easy to show that the transmit power for any node should be at least P_m. Define a *feasible* TDMA schedule for a power allocation P_m per node as one that constructs $\mathcal{G}_{2,m}$ while satisfying the half-duplex and interference constraints described in Sect. 6.2.

Without loss of generality, suppose node $(0,0)$ transmits in the first time slot with power P_m. From the definition of a feasible schedule, during this time there cannot exist a transmitter inside the square with vertices $(1,1)$, $(1,m)$, (m,m), $(m,1)$. As a result, each node that is contained in the square requires time slot each. It is easy to see that there are $(m-1)^2$ such nodes. Moreover, any of the $4m$ nodes that are currently receiving a message from $(0,0)$ cannot transmit at this time. Therefore, accounting for the current slot, the length of a feasible schedule cannot be shorter than $T_l = (m-1)^2 + 4m + 1 = m^2 + 2m + 2$ slots. In particular, since T_2^* is the length of the shortest feasible schedule, $T_2^* \ge T_l$.

If T_u is number of time slots taken by any feasible schedule to form $\mathcal{G}_{2,m}$, $T_2^* \leq T_u$. Consider the following schedule: in the first time slot, choose $(0, 0)$ as a transmitter and schedule nodes $(0 + p(2m + 1), 0 + q(2m + 1))$ for $p, q = 1, \ldots, \lfloor \frac{l}{2m+1} \rfloor$ to transmit. In other words, we attempt to tile the torus with squares of side $\frac{2m+1}{\sqrt{n}}$. Clearly this is feasible, since each node receives from at most one transmitter. In every subsequent time slot, repeat this process by choosing some other node (i, j) inside the square of side $(2m + 1)/\sqrt{n}$ centered at the origin and schedule nodes $(i + p(2m + 1), j + q(2m + 1))$ for transmission. Repeat this process until all the $(2m + 1)^2$ nodes in this square have been chosen once. Using arguments similar to those used in Lemma 1, a maximum of $\lfloor \frac{l}{(2m+1)} \rfloor^2$ simultaneous transmissions can be scheduled per time slot. After $(2m + 1)^2$ time slots,

$$\begin{aligned} n - &\left\lfloor \frac{l}{(2m+1)} \right\rfloor^2 (2m+1)^2 \\ &= 2\left\lfloor \frac{l}{2m+1} \right\rfloor (2m+1)\,\mathrm{rem}\big(l, (2m+1)\big) + \big(\mathrm{rem}(l, 2m+1)\big)^2 \end{aligned}$$

nodes are yet to transmit. In the first term, we can schedule $\lfloor l/(2m + 1) \rfloor$ nodes in each of the $2\,\mathrm{rem}(l, 2m + 1) \leq 4m$ "rows", that require at most $4m(2m + 1)$ additional time slots. Scheduling one node per time slot, all the remaining $(\mathrm{rem}(l, 2m + 1))^2$ nodes can transmit in at most $4m^2$ time slots. Therefore, the schedule constructs $\mathcal{G}_{2,m}$ in $(2m + 1)^2 + 4m(2m + 1) + 4m^2 = 16m^2 + 8m + 1$ time slots. Thus, we conclude that $T_2^* \leq 16m^2 + 8m + 1$. □

As compared to the 1-torus, the optimal schedule for a 2-torus is bounded by two quadratic terms. This is due to the interference constraints to the given geometry of node placement. As we shall see, this quadratic—rather than linear—dependence on m is the key to understanding the effect of transmit power on the convergence behavior.

6.3.2.1 Bounding the Rate of Convergence

We now find the eigenvalues of $F_{2,m}$ by exploiting its block circulant property.

Lemma 3 *Let $\mathcal{G}_{2,m}$ be the consensus graph formed over $\mathcal{T}_2(n)$ using the Manhattan connectivity model. If $L_{2\,m}$ is its Laplacian then the eigenvalues of the $F_{2,m} = I - hL_{2,m}$ are*

$$\lambda_{a,b} = 1 - 2h(2m + 1) + hS_a^{(m,l)} + hS_b^{(m,l)}$$

where, as defined above $S_a^{(m,l)} = \frac{\sin(\frac{(2m+1)\pi a}{l})}{\sin(\frac{\pi a}{l})}$, $a = 0, 1, \ldots, l - 1$.

Proof From (6.7) $F_{2,m}$ is an $n \times n$ block circulant matrix with its first block row being

$$[Q_m \; R_m \; 0 \; 0 \; \cdots \; 0 \; R_m]_{l \times n}$$

where $R_m = h\mathbf{1}_m^* \otimes I$. As noted earlier, Q_m is circulant. Since the identity and the all-zero matrices are also circulant, all these matrices share the same eigenvectors. Using this in conjunction with the properties of block circulant matrices, we compute the eigenvalues $\mu_{r,s}$ of $F_{2,m}$ as

$$\mu_{r,s} = \sum_{t=0}^{l-1} \eta_{r,t} e^{-j\frac{2\pi st}{l}} \tag{6.8}$$

where $\eta_{r,t}$ is the rth eigenvalue of $Q_m \forall r, s = 0, 1, \ldots, l-1$. Using the eigenvalues for the 1-torus and simplifying yields the eigenvalues of $F_{2,m} = I - hL_{2,m}$ are

$$\lambda_{a,b} = 1 - 2h(2m+1) + hS_a^{(m,\sqrt{n})} + hS_b^{(m,\sqrt{n})} \tag{6.9}$$

which is the desired result. □

We are now in a position to bound the rate of decay for the case of the torus. We have the following result.

Theorem 2 *Consider a consensus algorithm of the form* (6.2) *on* $\mathcal{G}_{2,m}$. *If each node transmits at* P_m *for* $1 \le m < \lfloor \frac{\sqrt{n}}{2} \rfloor$, *the rate of convergence* β *of an optimal MAC schedule on* $\mathcal{G}_{2,m}$ *that drives* $\delta(k) = x(k) - \mathbf{1}_n x_{av}$ *to zero is bounded as*

$$\lambda_1^{\frac{1}{m^2+2m+2}} < \beta < \lambda_1^{\frac{1}{16m^2+8m+1}}$$

where

$$\lambda_1 = \left(1 - h(2m+1) + h(2m+1)S_1^{(m,\sqrt{n})}\right).$$

Proof From Lemma 3 above, the eigenvalues of the update matrix are

$$\lambda_{a,b} = 1 - 2h(2m+1) + hS_a^{(m,\sqrt{n})} + hS_b^{(m,\sqrt{n})}$$

The largest eigenvalue is obtained by maximizing both the sinc terms, i.e., by choosing $(a,b) = (0,0)$. The second largest eigenvalue is doubly degenerate and obtained for $(a,b) = (0,1)$ or $(a,b) = (1,0)$. Any of these choices will simplify (6.9) to

$$\lambda_{0,1} = \lambda_{0,1} = 1 - h(2m+1) + hS_a^{(m,\sqrt{n})} = \lambda_1$$

From Lemma 1, we know that the length T_2^* of the optimal schedule length is bounded as $T_l \le T_2^* \le T_u$, where $T_l = m^2 + 2m + 2$ and $T_u = 16m^2 + 8m + 1$.

The rate of convergence of $\mathcal{G}_{2,m}$ with the optimal schedule is $\lambda_1^{1/T_2^*} \leq \lambda_1^{1/T_u}$ since $T_2^* \geq T_u$. Similarly, $\lambda_1^{1/T_l} \leq \lambda_1^{1/T_2^*}$. Hence, the result follows. $\square$

To understand the effect of higher transmit power on the convergence rate in 2-tori, first simplify the expressions for λ_1 in Theorem 2 using $h = \gamma/(4m+1)$, $0 < \gamma < 1$:

$$\lambda_1 = 1 - \gamma \frac{(2m+1)}{(4m+1)} + \gamma \frac{\sin((2m+1)\pi/\sqrt{n})}{(4m+1)\sin(\pi/\sqrt{n})}$$

We can compare this to the 1-torus case with $h = \gamma/(2m+1)$, where

$$\rho_1 = 1 - \gamma + \gamma \frac{\sin((2m+1)\pi/n)}{(2m+1)\sin(\pi/n)}$$

Clearly, λ_1 is of similar form as ρ_1 in Theorem 1, except that the term $\sqrt{n}$ is replaced by n.

However, there is a significant difference between the two cases in the length of the optimal schedule. This length was shown to be $\Theta(m)$ for the 1-torus, and $\Theta(m^2)$ for the 2-torus. This means that the effect of interference depends on the geometry of node placement. High transmit powers reduce available network throughput by causing more interference. In TDMA-scheduling MAC protocols, this effect is reflected in longer schedules. For a 1-torus, this is still offset by the resultant long-range connections. However, this is no longer true for higher dimensions where

6.3.2.2 Tori in Arbitrary Dimensions

The results in Lemma 2 can be extended to higher-dimensional grids with toroidal boundary conditions. Similarly, to find the upper bound one can generalize the schedule described in Lemma 1 that was used to find an upper bound. It can be shown that the length of the optimal schedule will be $\Theta(m^d)$.

The results in Lemma 3 can also be generalized to $d-$dimensions as $\lambda_1 = 1 - h(2m+1) + hS_1^{(m,l)}$. Thus, the convergence rate increases with transmit power in geometries having dimension 2 or more.

6.4 Hierarchical Networks

Since the calculation of the rates of convergence for average consensus for arbitrary graphs is not possible even without the interference constraints, we do not expect to be able to extend our results for arbitrary graphs. In this section, we consider another useful class of graphs that allow us to state analytical results. We consider a variation on the random geometric graphs by adding a backbone of dedicated (long-distance) communication nodes. Thus, we consider a hierarchical network with N

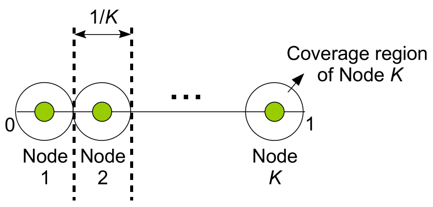


Fig. 6.6 Schematic of backbone nodes placed along a 1-dimensional torus. The ith backbone node is placed at $r_i = (i - \frac{1}{2})\frac{1}{K}$ from the origin, for $i = 1, 2, \ldots, K$

sensing nodes uniformly randomly placed on a torus in $[0, 1]^d$, and K^d identical regularly spaced *backbone* nodes on the torus, as shown in Fig. 6.6 for $d = 1$.

We assume that the backbone nodes do not participate in sensing, and only communicate with each other in the network. Each backbone node has a fixed exclusive region of coverage, i.e., it (alone) collects data from all the sensing nodes within a sphere of radius $r = 1/2K$. Initially, each backbone node collects and averages the data from all the sensing nodes in its region of coverage. All the backbone nodes now run an average consensus algorithm among themselves with their respective averaged data as initial values. We again assume the Manhattan connectivity model. It is possible to pass on this (global) average back to all the nodes within their coverage regions in $O(1)$ steps. Therefore for analyzing the rate of convergence, it is sufficient to analyze the time taken for collecting data by the backbone nodes and the time taken to reach consensus among these nodes.

We begin by characterizing the number of sensors reporting to each backbone node.

Lemma 4 *For large N and if the number of backbone nodes scales as $o(\frac{N}{\log N})$, asymptotically almost surely*[1] (*a.a.s.*) *the number of sensors per coverage area is $n = \frac{N\pi^{d/2}}{\Gamma(1+d/2)(2K)^d}$, where K is the number of nodes per dimension.*

Proof This can be proven by a variation of the argument used in the theory of random geometric graphs [20] to show regularity. Our approach here closely follows [21]. Consider a sphere $\mathcal{S}$ of radius r centered at point P on the torus. Marking the points $1, 2, \ldots, N$, we can associate with each point k a random variable X_k $(1 \le k \le N)$ defined as:

$$X_k = \mathbf{1}_{\mathcal{S}}(k)$$

where $\mathbf{1}(.)$ is the indicator function. That is, $\mathbf{1}_{\mathcal{S}}(k) = 1$ if $k \in \mathcal{S}$, and 0 otherwise. Since the sensing nodes are placed on the torus uniformly and independently of each other, $\{X_k\}$ are i.i.d. Bernoulli with success probability

$$p = \frac{\text{Vol. of sphere}}{\text{Vol. of torus}} = \frac{\pi^{d/2}}{\Gamma(1 + d/2)(2K)^d}$$

[1]In this article, a property U_N is said to hold asymptotically almost surely if and only if $\mathbb{P}(U_N$ is true) tends to 1 as $N \to \infty$.

The number of sensors inside the sphere is thus a binomial random variable

$$Y \triangleq \sum_{k=1}^{N} X_k$$

with $\mu \triangleq \mathbb{E}Y = Np$. We can now use the Chernoff bound:

$$\mathbb{P}\big(|Y-\mu| > \mu\delta\big) \le 2\exp\left(-\frac{\mu\delta^2}{2}\right)$$

For $0 < \delta < 1$, since the number of nodes $K^d = o(\frac{N}{\log N})$ we can always choose

$$\delta = \sqrt{\frac{4\log N}{N}\cdot \gamma K^d} = o(1), \quad N \to \infty$$

for some $\gamma > 0$. Plugging this value into the bound, we see that

$$\mathbb{P}\big(Y \notin \big[\mu(1-\delta), \mu(1+\delta)\big]\big) \le \frac{1}{N^{2\eta\gamma}} \le \frac{1}{N^3}$$

where $\eta = \frac{\pi^{d/2}}{2^d \Gamma(1+d/2)}$ and γ is chosen such that $\eta\gamma > 2$. So a.a.s.,

$$Y = Np\big(1 \pm o(1)\big)$$

Thus, the probability that any coverage area does not have $n(1 \pm o(1))$ nodes is

$$\begin{aligned}\mathbb{P}\left(\bigcup_k \big\{Y_k \notin \big[\mu(1-\delta), \mu(1+\delta)\big]\big\}\right) &\le \bigcup_k \mathbb{P}\big(Y_k \notin \big[\mu(1-\delta), \mu(1+\delta)\big]\big) \\ &= K^d \frac{1}{N^3} < N\frac{1}{N^3} = \frac{1}{N^2}\end{aligned}$$

where we have used the union bound. The result now follows readily. □

Since each of the K^d backbone nodes has n sensors in its coverage region a.a.s. when N is large, a total of nK^d sensing nodes will be covered by the backbone network. Therefore for large N, it is always possible to achieve consensus over a positive fraction

$$\kappa = \frac{nK^d}{N} = \frac{\pi^{d/2}}{2^d \Gamma(1+d/2)}$$

of the sensing nodes, independent of N and K.

We will now study specific cases for $d = 1$ and 2. To begin, we note that for $d = 1$,

$$\kappa_1 = \frac{\pi^{1/2}}{2\Gamma(3/2)} = 1$$

and for $d = 2$,

$$\kappa_2 = \frac{\pi}{4\Gamma(2)} = \frac{\pi}{4} \approx 78.5\%$$

As stated above, we assume that each backbone node collects data by polling all the sensing nodes in its coverage region. This is done in parallel over all backbone nodes, by a suitable choice of transmit power. Assuming each node transmits in one time slot, we need n time slots to initialize the consensus algorithm, where a.a.s.

$$n = \frac{\alpha N}{K^d}$$

During the consensus phase, the network topology is identical to the (regular) ring and torus topologies discussed previously. As before, let each node transmit at fixed power P_m to reach $m \leq \lfloor K/2 \rfloor$ neighbors per direction per dimension. Using the results in Theorems 1 and 2, we obtain that for $K^d = o(\frac{N}{\log N})$, a.a.s.,

$$\left(1 - h(2m+1) + hS_1^{(m,K)}\right)^{\frac{1}{4m+1}} \geq \beta \geq \left(1 - h(2m+1) + hS_1^{(m,K)}\right)^{\frac{1}{2m+1}}$$

for the ring or 1D torus and

$$\left(1 - h(2m+1) + hS_1^{(m,K)}\right)^{\frac{1}{16m^2+8m+1}} > \beta > \left(1 - h(2m+1) + hS_1^{(m,K)}\right)^{\frac{1}{m^2+2m+2}}$$

for the 2D torus.

6.5 Conclusions

We have introduced a framework that considers the effects of realistic communication constraints on average consensus algorithms. In particular, we analytically characterize the performance of the medium access control algorithm that maximizes the speed of convergence. We study the effect of transmit power on convergence in the presence of interference. In interference-limited wireless networks, the geometry of node placement plays a key role in deciding the fastest converging consensus graph. While forming long-range links (using more power) always improves the convergence on ring topologies, it is not so for higher-dimensional tori.

This work could be extended to other classes of graphs, like Cayley graphs and expander graphs that have good convergence properties [22]. Another issue is the effect of stochastic data loss through effects due to fading and interference, using a different framework as compared to [2, 10], to explicitly account for interference.

Acknowledgements The partial support of DTRA grant N00164-07-8510 and NSF grant 0846631 is gratefully acknowledged.

References

1. Blondel, V., Hendrickx, J., Olshevsky, A., Tsitsiklis, J.: Convergence in multiagent coordination, consensus and flocking. In: Proceedings of the 44th IEEE Conference on Decision and Control, pp. 2996–3000 (2005)
2. Boyd, S., Ghosh, A., Prabhakar, B., Shah, D.: Randomized gossip algorithms. IEEE Trans. Inf. Theory **52**(6), 2508–2530 (2005)
3. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Trans. Autom. Control **48**(6), 988–1001 (2003)
4. Fang, L., Antsaklis, P.: On communication requirements for multi-agents consensus seeking. In: Proceedings of Workshop NESC05: University of Notre Dame. Lecture Notes in Control and Information Sciences (LNCIS), vol. 331, pp. 53–68. Springer, Berlin (2006)
5. Olfati Saber, R., Murray, R.M.: Consensus problems in networks of agents with switcing topology and time-delays. IEEE Trans. Autom. Control **49**(9), 1520–1533 (2004)
6. Ren, W., Beard, R.W., McLain, T.W.: Coordination variables and consensus building in multiple vehicle systems. In: Proceedings of the Block Island Workshop on Cooperative Control. Lecture Notes in Control and Information Sciences, vol. 309, pp. 171–188. Springer, Berlin (2004)
7. Xiao, L., Boyd, S.: Fast linear iterations for distributed averaging. In: Proceedings of 42th IEEE Conference on Decision and Control, pp. 4997–5002 (2003)
8. Xiao, L., Boyd, S., Lall, S.: A scheme for robust distributed sensor fusion based on average consensus. In: Proceedings of International Conference on Information Processing in Sensor Networks, pp. 63–70 (2005)
9. Nedich, A., Olshevsky, A., Ozdaglar, A., Tsitsiklis, J.N.: On distributed averaging algorithms and quantization effects. LIDS Technical Report 2778, MIT, Lab. for Information and Decision Systems
10. Hovareshti, P., Gupta, V., Bars, J.S.: Average consensus over small world networks: a probabilistic framework. In: Proceedings of the IEEE Conference on Decision and Control (CDC' 08), pp. 375–380 (December 2008).
11. Huang, M., Manton, J.H.: Stochastic double array analysis and convergence of consensus algorithms with noisy measurements. In: Proc. American Control Conference, New York, pp. 705–710, July 2007
12. Huang, M., Manton, J.H.: Stochastic Lyapunov analysis for consensus algorithms with noisy measurements. In: Proc. American Control Conference, New York, pp. 1419–1424, July 2007
13. Nedich, A., Ozdaglar, A.: Convergence rate for consensus with delays. LIDS Technical Report 2774, MIT, Lab. for Information and Decision Systems
14. Desai, M.P., Rao, V.B.: A new eigenvalue bound for reversible Markov chains with applications to the temperature-asymptotics of simulated annealing. Proc. IEEE Int. Symp. Circuits Syst. **2**, 1211–1214 (1990)
15. Seneta, E.: Nonnegative Matrices and Markov Chains, 2nd edn. Springer, Berlin (1981)
16. Gupta, P., Kumar, P.R.: The capacity of wireless networks. IEEE Trans. Inf. Theory **46**(2), 388–404 (2000)
17. Liu, X., Haenggi, M.: Throughput analysis of fading sensor networks with regular and random topologies. EURASIP J. Wirel. Commun. Netw. **4**, 554–564 (2005). Special Issue on Wireless Sensor Networks
18. Xie, M., Haenggi, M.: Delay performance of different MAC schemes for multihop wireless networks. In: IEEE Global Communications Conference (GLOBECOM'05), St. Louis, MO, November 2005
19. Vanka, S., Gupta, V., Haenggi, M.: Power-delay analysis of consensus algorithms on wireless networks with interference. Int. J. Syst. Control Commun. **2**(1), 256–274 (2010)

20. Penrose, M.: Random Geometric Graphs. Oxford University Press, London (2003)
21. Boyd, S., Ghosh, A., Prabhakar, B., Shah, D.: Mixing times for random walks on geometric random graphs. In: SIAM Workshop on Analytic Algorithmics & Combinatorics (ANALCO), Vancouver, January 2005
22. Carli, R., Fagnani, F., Speranzon, A., Zampieri, S.: Communication constraints in the average consensus problem. Automatica **44**(3), 671–684 (2008)

Chapter 7
Analyzing the Theoretical Performance of Information Sharing

Paul Scerri, Prasanna Velagapudi, and Katia Sycara

Summary Individuals in large, heterogeneous teams will commonly produce sensor data that is likely useful to some other members of the team, but it is not precisely known to whom the information is useful. Some recent work has shown that randomly propagating the information performed surprisingly well, compared to infeasible optimal approaches. This chapter extends that work by looking at how the relative performance of random information passing algorithms scales with the size of the team. Additionally, the chapter looks at how random information passing performs when sensor data is noisy, so that individuals need multiple pieces of data to reach a conclusion, and the underlying situation is dynamic, so individuals need new information over time. Results show that random information passing is broadly effective, although relative performance is lower in some situations.

7.1 Introduction

Exciting applications are emerging that involve large, heterogeneous teams acting in complex environments. Examples include search and rescue [4], disaster response [13], and military applications [3]. In such domains, team members will often collect local information that is necessary or useful to other members of the team. For example, in urban search and rescue operations, an aerial robot might be able to locate a victim, but be unable to assess their condition or determine a route to them. A ground robot on the team could perform these tasks, but might be unable to locate the victim by itself. Efficiently getting such information from those collecting it to those requiring it is one of the keys to effective team performance. However,

P. Scerri (✉) · P. Velagapudi · K. Sycara
Carnegie Mellon University, Pittsburgh, PA 15217, USA
e-mail: pscerri@cs.cmu.edu

P. Velagapudi
e-mail: pkv@cs.cmu.edu

K. Sycara
e-mail: katia@cs.cmu.edu

teammates often have limited information about which, if any, team members require particular pieces of information. Thus, the team member collecting some piece of information needs to determine whether and where to send collected information with limited knowledge of who might need it and how important it is to them. At the same time, team members must also be careful about what they communicate as the volume of incoming information is typically dramatically higher than available communication bandwidth.

Recognizing the utility of such information and delivering it efficiently across a team has been the focus of much research, with proposed approaches ranging from flooding [2] to channel filters [1] and matchmakers [7]. Interestingly, random forwarding of information has been found to be a surprisingly effective information sharing approach in some domains [2]. In previous work, we investigated this phenomena in detail and showed that, in certain systems, random forwarding of information performs almost half as well as a globally optimal approach [15].

In small teams or static environments, a variety of approaches have been applied to the information sharing problem. One example, STEAM [14], requires team members to keep others informed of their current state, allowing members to intelligently reason about which teammates need which information. Approaches using matchmakers [7] allow team members to keep some central point informed of their state, while the control point is responsible for directing information as required. More recently, for applications such as sensor networks, algorithms drawing on intuitions of how human gossip [2] works have been shown to be effective, but often wasteful with bandwidth. Token-based algorithms have also been shown to be effective for large-scale team coordination [17] and belief sharing [15] in some domains.

An interesting feature of both gossip and token algorithms is that little knowledge of the team is known or assumed. Nearly random communication coupled with local reasoning is sufficient to produce surprisingly competitive results [16]. It is this surprising effectiveness of lightweight, decentralized, and largely random algorithms that is the focus of this paper. The intention in this work is to understand and quantify when and how these simple strategies will be effective.

In previous work, we established an upper-bound on average case performance of information sharing in large teams and showed that in certain circumstances random policies can achieve a significant portion of that performance [16]. By adding simple heuristics to avoid redundant communications, it was possible to improve the performance of a purely random policy significantly. This means that in domains where network and utility distributions are similar to these cases, random information sharing policies may present an efficient and robust information sharing solution.

This paper extends that previous work in two important ways. First, it looks at how the relative performance of the random policies vary with the size of the network. Two opposing forces influence the performance. On the one hand, bigger networks allow random strategies to revisit the same agent less often, improving their relative performance, but bigger networks give more options for an intelligent strategy to exploit, reducing the relative performance of random strategies. Second, this paper looks at cases where agents might need more than one piece of information, either because the information is noisy or because the environment is dynamic.

In one case, there are multiple, overlapping, noisy sensor readings generated by the team over time. Here, even team members with very high interest in a particular piece of information will not need to know every reading, only a subset that allows the underlying features or state to be inferred correctly. In another case, the underlying environment changes over time and agents need to track features. Results show that random information propagation is relatively better in the noisy case, but relatively worse in the dynamic case.

7.2 Information Sharing

In this section, we formally describe the information sharing problem. Consider a team of agents, A, working to achieve some goal. Suppose there is a piece of information η obtained by team member $a \in A$. If any other member $b \in A$ were to obtain that information, it would impact their ability to perform the team goal. Define the utility $\xi_a(\eta)$ as the quantification of this change in performance. In order for b to get the information, it must be communicated across the network, N. This requires some members of the team to spend resources such as time and power on communicating the information. Thus, there is some communications cost $\kappa(a, b, \eta)$ associated with transmitting η from a to b.

The best team performance is achieved when information is shared with the set of team members that have a higher utility for the information than the cost of communicating it to them. If the communication cost is expressed in the same units as the utility, this is represented by the maximization:

$$A^* = \arg\max_{A \subseteq T} \sum_{a \in A} \big(\xi_a(\eta) - \kappa(\cdot, a, \eta)\big) \tag{7.1}$$

In order to address the specific problem of information sharing in a large, dynamic team, we make several key assumptions. First, communications are assumed to be peer-to-peer and of a constant cost per transmission. Rather than representing the communications cost of every pair of agents, $\kappa(a, b, \eta)$ can be compactly represented as some fixed cost κ when agents are neighbors in the network, and infinite when they are not. This is reasonable for domains with peer-to-peer communications, as transmission between distant teammates in a network can be decomposed into a sequence of transmissions to their intermediate neighbors. The only solutions that are lost in this decomposition are solutions where teammates forward information along but do not make use of it, a rare case for a team.

Given this assumption, it is possible to condition the performance of an information sharing algorithm by the number of communications it has used. In many domains, the tradeoff between communication cost and utility is not well characterized or fixed. Avoiding it allows results to be generalized by removing the second term from (7.1) and allowing us to compare performance across algorithms which make multiple communications per time step.

Second, while the utility of some information will change over time, we assume that communication is sufficiently fast that utility is constant while a single piece of information is being shared across the network. For example, in a search and rescue domain, the utility of knowing where a fire is will change if the fire spreads or team members move relative to it. However, the speed of these changes is orders of magnitude slower than the millisecond scale transmission speeds of a modern wireless network connecting the team members.

Finally, in a large team, rather than modeling the utility explicitly for each member, we assume that it can be summarized in a *utility distribution* over all agents. This distribution represents the probability that team member A has some utility ξ for a piece of information η. For a given domain, the utility distribution can be computed empirically by conditioning on relevant variables and sampling utility as information is shared in the team. For analytic purposes, we can approximate this distribution by a number of canonical probability distributions such as normal, uniform, and exponential distributions.

7.2.1 Token Algorithms

Given these assumptions, we consider two extremes of token-based algorithm design in addressing this problem. In these algorithms, a token is created that contains some information η. This token is atomically passed from teammate to teammate. When a team member receives the token, it can make use of the information inside, then decide to either forward the token to a neighbor or delete it. Since tokens use exactly one communication per time step, token algorithms can control the number of communications by using tokens that are deleted after a fixed number of steps.

If we take advantage of all possible knowledge of agent utility and network properties, the optimal approach is to directly solve the maximization in (7.1). This is done using an exhaustive search of all possible network paths of length t. We call this a t-step *lookahead* approach.

On the other hand, if we ignore all available knowledge of agent utility, we can propose a simple algorithm of randomly passing information from neighbor to neighbor. This equates to simply performing a random walk across the network. We therefore call this the *random walk* approach.

Given that no knowledge of utility is used in routing a random walk, its efficiency is primarily determined by its coverage of the network. We therefore introduce two intermediate algorithms that are equally naïve with regards to utility, but significantly more intelligent about coverage. A token will maximize coverage if it never revisits the same agent in a network. Thus, a straightforward improvement to the random walk approach is the addition of a history of nodes carried within the token. As the token moves around the network, visited agents are marked in this history, and when the token is being routed, this history is used to exclude visited agents from selection. If all neighbor nodes are visited, the algorithm selects a link at random. This approximates a self-avoiding walk over the network, so we term this the *random self-avoiding walk* approach.

This approach is reasonable when the size of the history is expected to be bounded to a reasonable size. However, in very large teams or systems where tokens visit many agents of the team, this is not a practical solution. In these cases, if there are a bounded number of tokens in existence at any given time, an alternative solution is to maintain a local history at each agent for each active token, consisting of its previously used incoming and outgoing network connections. Similarly to the random self-avoiding approach, agents that receive a token multiple times will attempt to send it to different neighbors each time, selecting randomly from the outgoing edges if all of them have been previously used. We designate this the *random trail* approach.

7.3 Experimental Results

A highly abstracted information sharing token simulator was created to empirically test the performance of token-based information sharing methods. The simulation consists of a network of agents that are assigned utilities for a given piece of information from a specified distribution. A token representing that information is initialized at a randomly chosen agent within the network. The agents propagate the token around the network according to some routing policy, and the accumulated utility is recorded at each step until the simulation executes some fixed number of steps. Results are averaged over 20 runs.

Five canonical network types were examined: small-worlds, scale-free, hierarchical, lattice, and random. Unless otherwise stated, each was generated to contain 1000 nodes with an average degree of 4. The small-worlds network was generated by adding random links to a doubly-connected ring. The scale-free network was generated using a tunable variant of the Barabási–Albert algorithm [9]. The hierarchical network was formed by adding nodes evenly to a balanced tree. The lattice was a four-connected 2D grid with wraparound, and the random network was created by adding random links until the average degree was reached. In most cases, results for random networks were analogous to those of scale-free networks, thus some results for random networks were omitted for brevity.

Three canonical distributions were examined: uniform, normal, and exponential. The uniform distribution was over the interval [0, 1]. The exponential distribution had a rate parameter of $\lambda = 1.0$, but was scaled by a factor of 0.2. In the case of the normal distribution, the mean and variance of the distribution were sometimes altered for various trials, but the nominal parameters were $\mu = 0.5$, $\sigma = 0.2$.

Four information sharing methods were considered: optimal, random walk, random trails, and self-avoiding walks. The optimal policy was approximated using a finite lookahead policy with global knowledge. Every m-steps, an exhaustive search of paths of length m was executed to determine an optimal path. This path was executed fully, followed by another m-step planning phase. Ideally, this stage would consist of a single path search of the final path length, but computing this path is extremely expensive due to the nonMarkovian nature of the utility function (as agents are visited, their utility drops to zero, so the joint distribution of utility is always dependent on complete network state). Instead, smaller values of m were chosen empirically from the results of early experiments.

7.3.1 *Optimality of the Lookahead Policy*

In order to determine a sufficient approximation of optimality, an experiment was conducted in which the depth of the lookahead policy was varied over the four network types with a normal utility distribution ($\mu = 0.5, \sigma = 0.2$), and the utilities of the resulting communication paths computed. The results of these tests can be seen in Fig. 7.1, where the utility obtained by each token is plotted against the number of communications the token was allowed. As lookahead depth increases, the

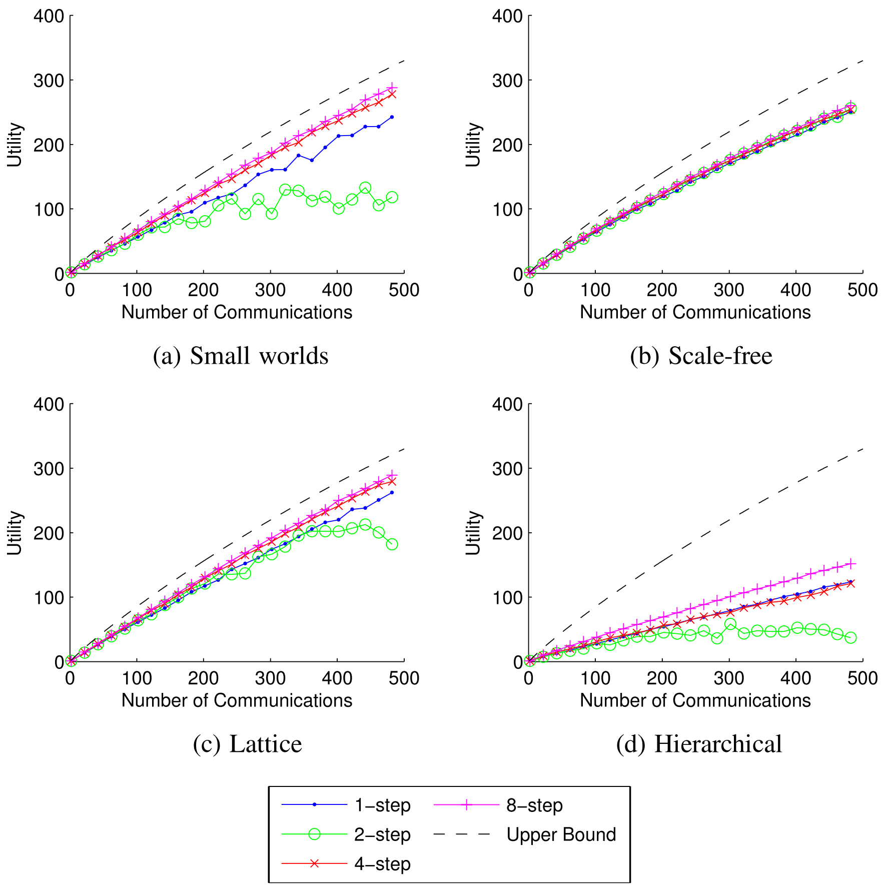


Fig. 7.1 Optimality of n-step lookahead over four network types with a normal utility distribution ($\mu = 0.5, \sigma = 0.2$). The utility obtained by each token is plotted against the number of communications used

obtained utility converges asymptotically to the optimal. Interestingly, while lookahead depths of 1, 4 and 8 converge toward an asymptote, 2-step lookaheads, denoted by the circle symbols, appear to perform pathologically poorly. It is possible that this is due to a negative interaction between the width of the networks and the depth of the search pattern, where the lookahead policy may consistently make myopic routing decisions. From these results, a lookahead policy with a depth of 4 was selected as a baseline for future experiments, as a compromise between computational complexity and optimality of performance.

It is also possible to evaluate the optimality of token-based lookahead policies against the upper bound established in [16]. The dashed lines in Fig. 7.1 correspond to these bounds. Two key characteristics are immediately evident. First, the lookahead policies often converge very closely to the upper bound on performance, suggesting that in the ideal case, token routing methods can perform very close to optimal. Second, while the bound is independent of network structure, clear differences are visible in the optimality of the lookahead policy over different network types. Most notably, in Fig. 7.1(d), the hierarchical network performs much worse than the upper bound, suggesting that information propagation via tokens in this type of structure is either highly inefficient or requires an extremely deep lookahead depth.

7.3.2 Optimality of the Random Policies

The four information sharing methods were tested in normal and exponential distributions. Figures 7.2 and 7.3 show the results of these experiments. It is evident that performance is heavily influenced by network type, with all policies performing significantly worse in hierarchical networks with both distributions and random policies performing proportionally much worse in the small worlds and hierarchical networks (Figs. 7.2(d) and 7.3(d)). Random policies perform best in the scale-free and lattice networks, with purely random walks attaining almost half the utility of the lookahead policy in the scale-free network with a normal utility distribution. The addition of self-avoiding heuristics appears to improve random policy performance significantly, primarily in the lattice and scale-free networks. The similar performance of the random trail and self-avoiding walk policies suggests that they are comparably effective at avoiding previously visited agents when covering the network. Further experiments focused on the random trail policy, as it typified the performance of the heuristic random policies.

As an example of the surprisingly efficient performance of random policies, consider Fig. 7.2(b). In it, we find that a utility of 175 is attained by a lookahead policy using an average of 300 communications. The same utility is obtained by a random self-avoiding policy in 425 communications. However, the random self-avoiding policy has no computational or structural overhead as it is *completely unaware of utility.* This suggests that under these conditions, if the cost of maintaining the necessary knowledge to perform an optimal strategy is on par with the cost of the extra

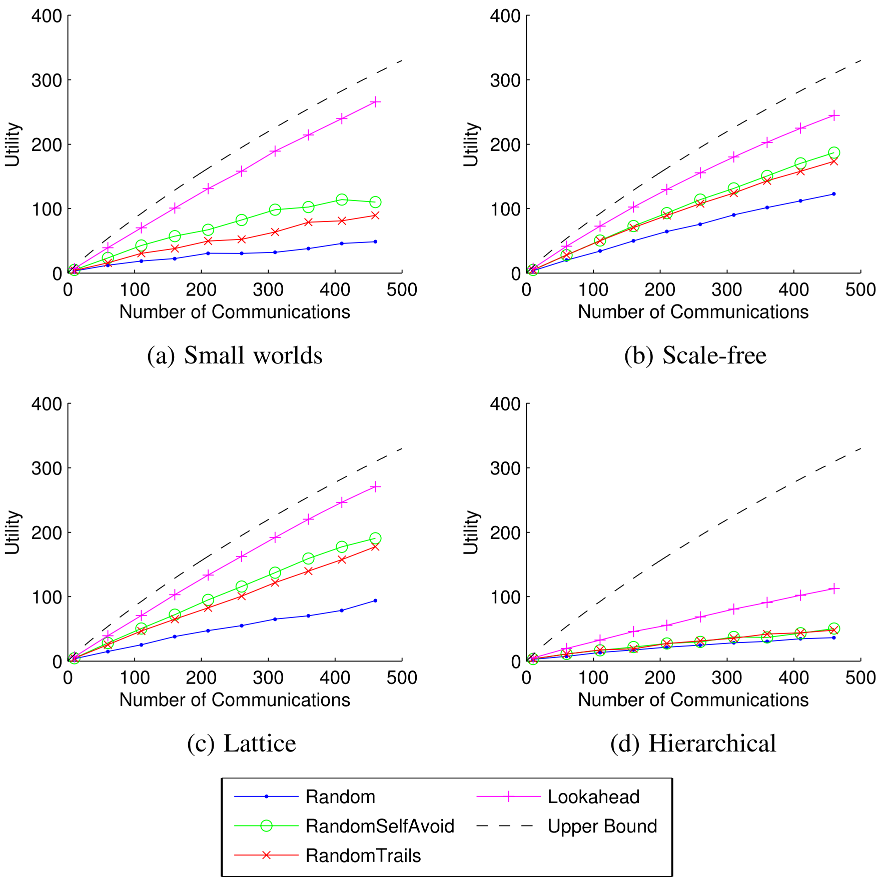


Fig. 7.2 Performance of random and lookahead policies over four network types with a normal utility distribution ($\mu = 0.5, \sigma = 0.2$). The utility obtained by each token is plotted against the number of communications used

125 communications, a randomized policy is in fact a competitive strategy for information sharing.

7.3.3 Effects of Noisy Estimation

Exploring this tradeoff further, we examine the effects of noisy estimates of utility on performance of the lookahead policy. Gaussian noise was introduced into the utility estimates of unvisited team members used by the lookahead policy. The utility of visited members was fixed at zero and not affected by this noise. To simulate the compounded inaccuracy of estimating the utility of team members further away in the network, the standard deviation of the noise (γ) was scaled exponentially by the

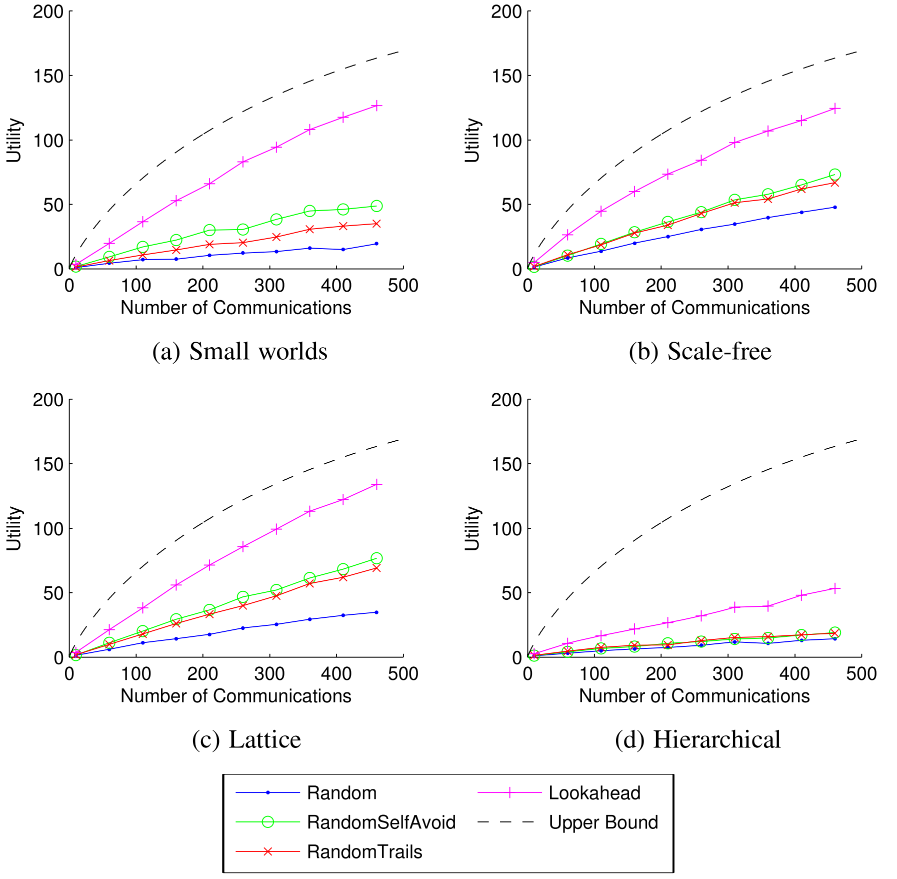


Fig. 7.3 Performance of random and lookahead policies over four network types with an exponential utility distribution ($\lambda = 1.0$, scale factor of 0.2). The utility obtained by each token is plotted against the number of communications used

network distance between teammates ($d_{a,b}$) using the following equation

$$\gamma_{a,b} = (\gamma + 1.0)^{d_{a,b}} - 1.0 \tag{7.2}$$

As seen in Fig. 7.4, as the amount of noise was increased, the performance of the lookahead policies degraded. This suggests that even when using an ideal routing policy, incorrect estimates of utility can disrupt intelligent routing policies. However, at $\gamma = 1.0$, the noise was so large that estimates of utility were approximately random. The only usable information available in this condition was that the utility of visited teammates was fixed at zero. Without the ability to discern high- and low-utility team members, the remaining difference in performance between the lookahead and random policy in the high noise condition cannot be attributed to the selection of higher utility paths. However, it may be the result of the lookahead

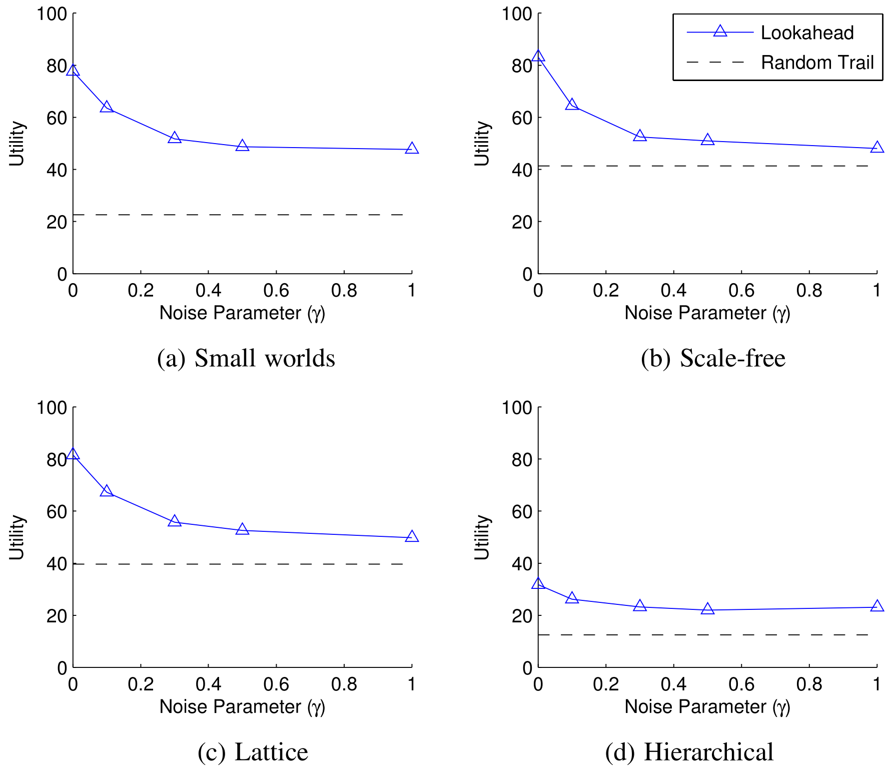


Fig. 7.4 Effects of noise on lookahead and random trail policy over four network types with a normal utility distribution ($\mu = 0.5, \sigma = 0.2$). The utility obtained by each token is plotted against a noise scaling parameter γ

policy's ability to avoid myopic routing decisions that might force future communications to pass through visited teammates.

7.3.4 Properties Affecting Optimality

Another observation was that the proportional gap between the lookahead policy and the random trail policy varied repeatably over networks and utility distributions across trials, suggesting that a combination of network structure and utility distribution properties affect the efficiency of the random trail policy. To explore this further, a wide array of networks and utility distributions were tested across a constant number of communications of $t = 250$ to study how the optimality of the random trail policy was affected by various properties. A cross section of these results can be found in Figs. 7.5 and 7.6. It was found that certain characteristics clearly affected optimality, while most had negligible effects.

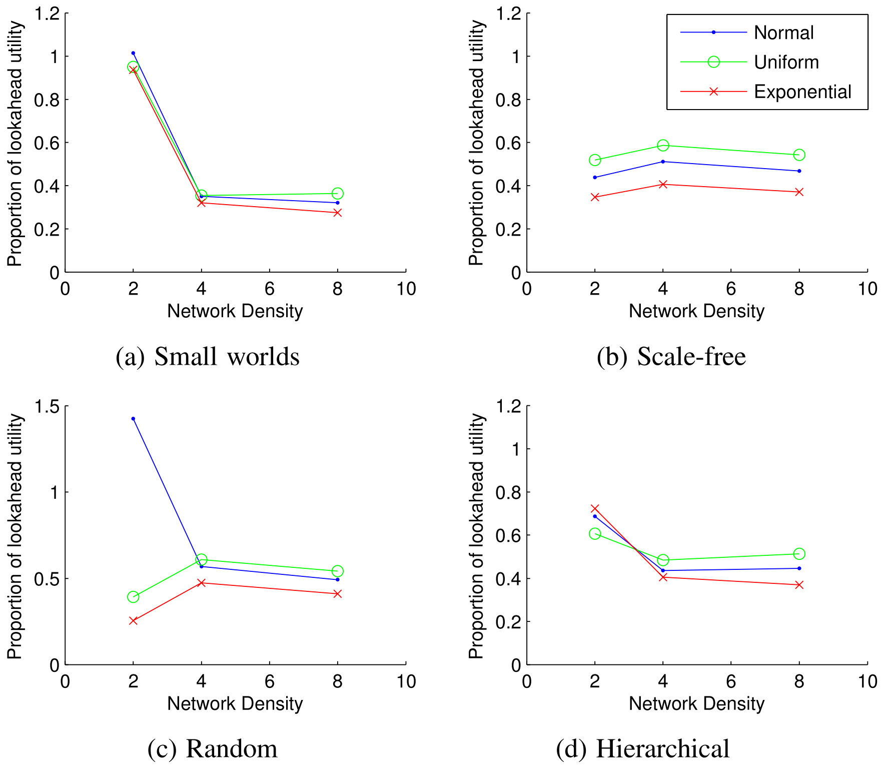


Fig. 7.5 Effects of network density on optimality of random trail policy over four network types with a normal ($\mu = 0.5, \sigma = 0.2$), exponential ($\lambda = 1.0$, scale factor of 0.2), and uniform utility distribution. The proportion of lookahead utility refers to the utility of the random trail policy scaled by that of a 4-step lookahead policy

The type of network had a clear impact on optimality. Interestingly, small worlds and hierarchical networks were similar in performance and contrasted with scale-free networks. In this experiment, particularly interesting results were obtained for random networks, so these results are presented in lieu of the grid results.

The *network density* was another property that showed a clear effect on optimality. At low network densities ($\rho = 2$), the average case performance of the random trail algorithm matched or exceeded the optimal policy on the small worlds and random networks. The consistency of this result, and its specificity, suggest that certain combinations of network structure, utility distribution, and network density are pathological for the lookahead policy. At higher densities, the optimality seemed to converge to a constant value dependent on network type.

Aside from inconsistent behavior at low network densities, the variance of the utility distribution also affected optimality, with the random trail policy performing better as variance was decreased. This makes sense, as a perfect self-avoiding policy over a network of members with constant utility (no variance) will always take an optimal path.

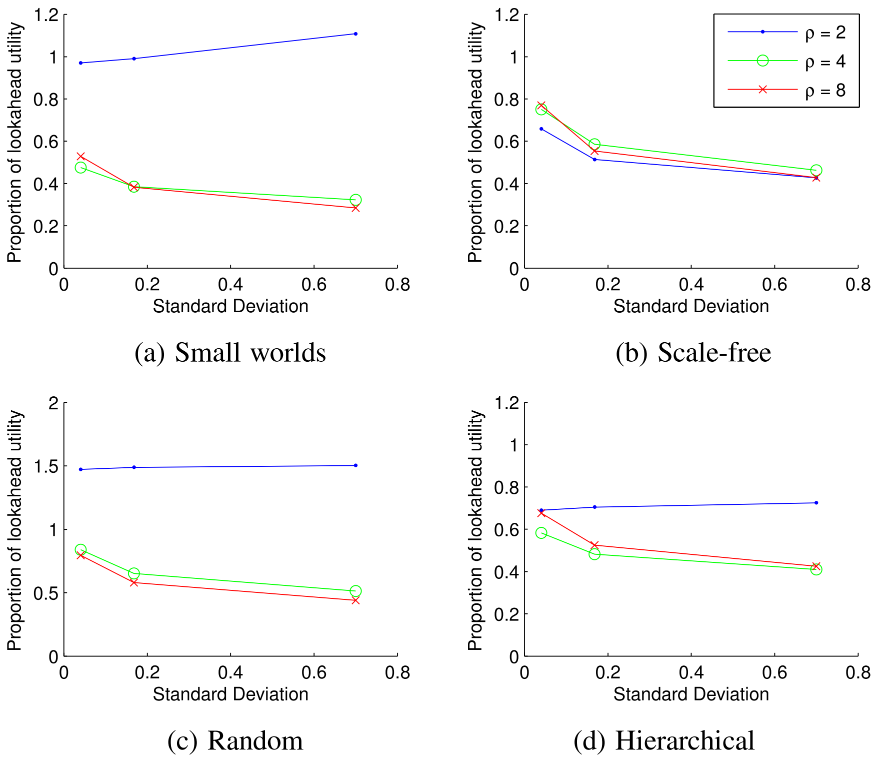


Fig. 7.6 Effects of variance on optimality of random trail policy over four network types with a normal utility distribution ($\mu = 0.5, \sigma$ varied) and varying network densities. The proportion of lookahead utility refers to the utility of the random trail policy scaled by that of a 4-step lookahead policy

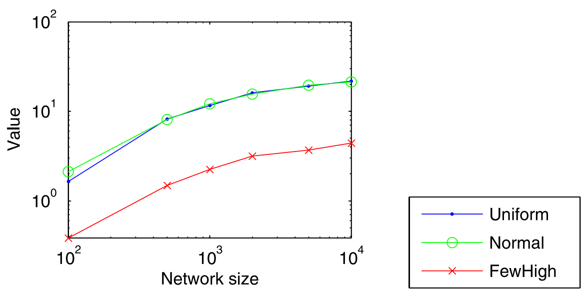


Fig. 7.7 Effects of varying value distribution as a scale free network is scaled up on relative optimality of random policies

7.3.5 *Scaling Network Size*

In the initial set of scaling experiments, distribution type, routing type and network type were varied as the size of the network was scaled up. The results are shown in Figs. 7.7–7.9. Each of the figures use log scaling on both the x- and y-axes. In

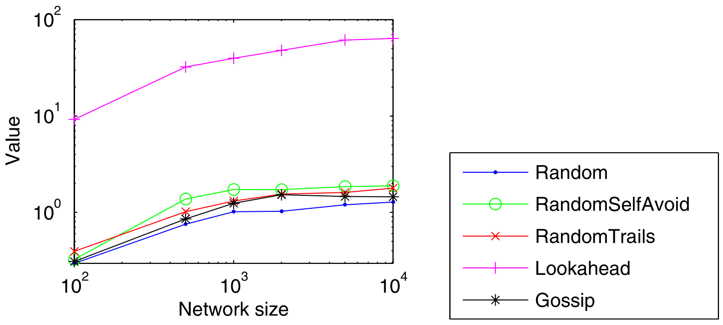


Fig. 7.8 Effects of varying routing type as a scale free network is scaled up on relative optimality of random policies

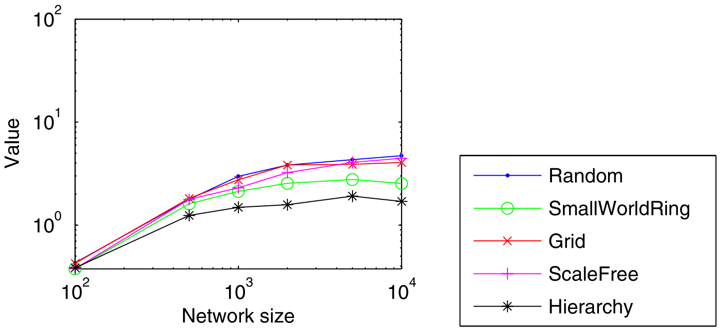


Fig. 7.9 Effects of varying network type as a scale free network is scaled up on relative optimality of random policies

the first of the experiments, varying the distribution type, random trails routing was used on a scale free network (Fig. 7.7). In all cases, overall value for a fixed token propagation length is shown. The slight improvement in value received with bigger networks was due to the larger network resulting in less situations where the token unavoidably retraced its own path. Notice that this effect lessens as the network gets bigger and will asymptotically reach some limit when it never retraces it path. In the second of the experiments, the routing type was varied on a scale free network, with an exponential value distribution, while the network size was scaled up. Notice that the lookahead policy, which knows the whole network improves its value much more with the increasing network size than any of the random policies. This is because the larger network offers more opportunities for the lookahead to exploit. Since the scale-free network has a small worlds property, the width of the network, i.e., the average distance between any two nodes in the network increases only slowly, thus offering many more opportunities for the lookahead to exploit, at relatively low cost. While the random policies benefit from retracing their steps less often, they do not proactively exploit the new opportunities. In the third scaling experiment, the network type is varied, with random trails routing and an exponential

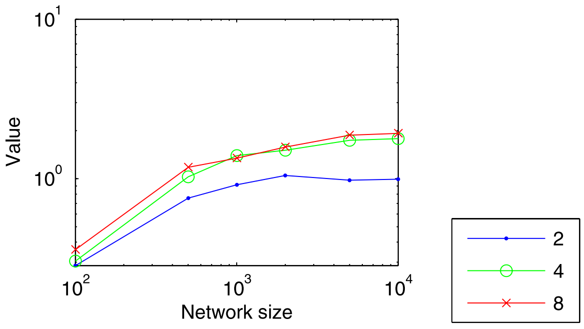


Fig. 7.10 Confirming the impact of revisits on random policies by varying network density of a scale free network with network size scale up

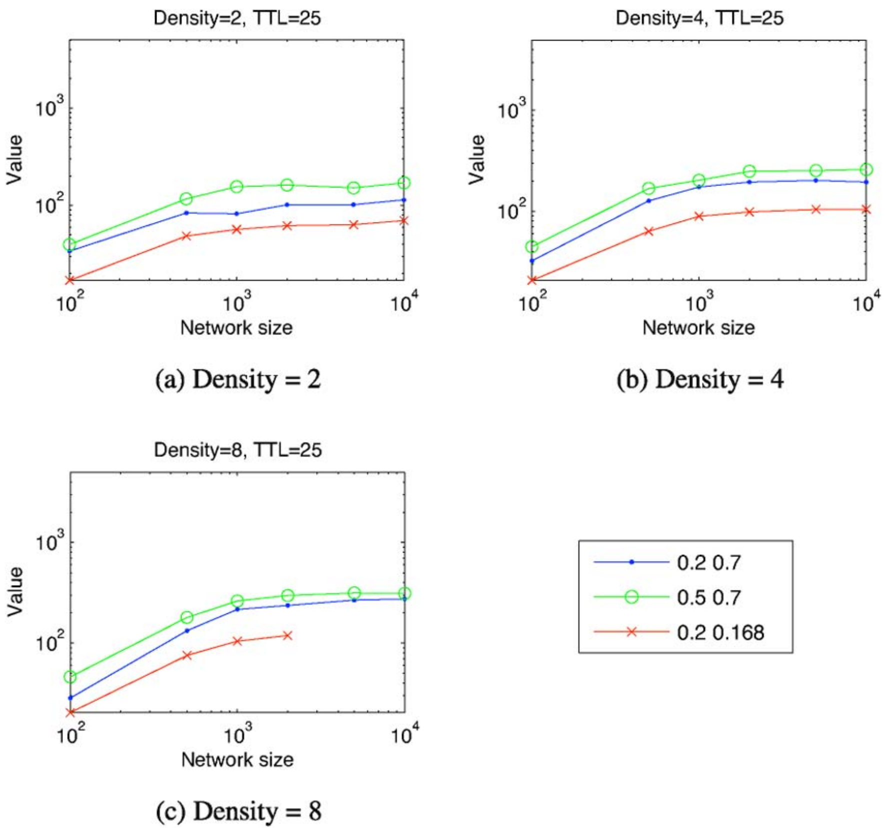


Fig. 7.11 The performance of the lookahead algorithm on small worlds networks of varied density, with TTL = 25

value distribution, while the network size is scaled up. The improvement in value for the bigger networks was much less pronounced for the hierarchy, because the structure of that network does not make it easy for the token to avoid revisiting agents,

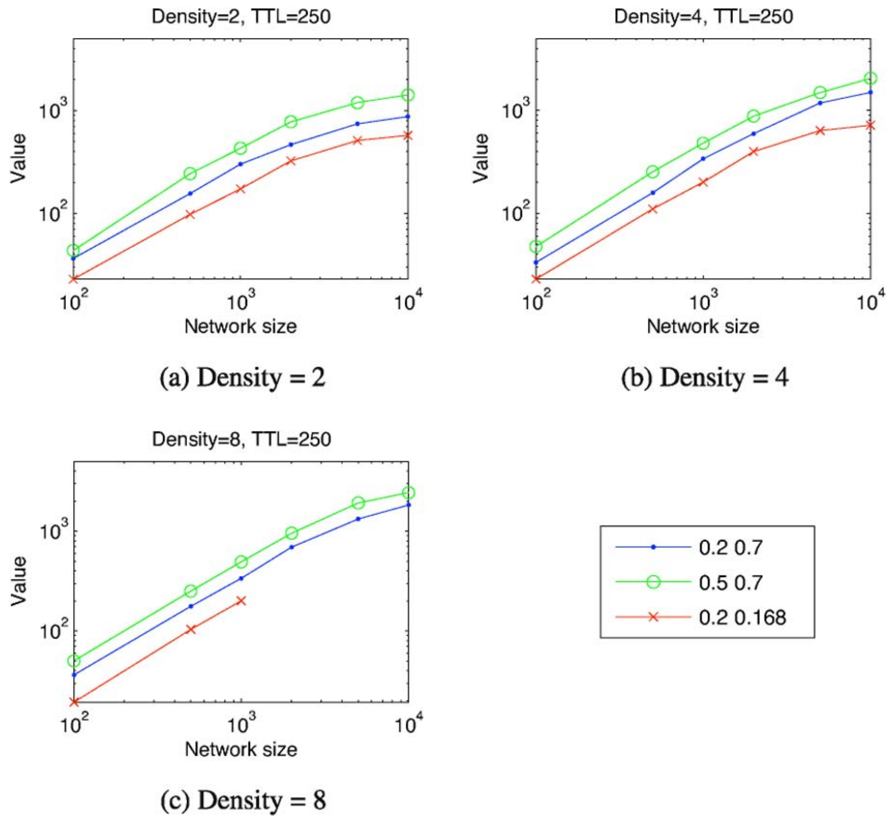


Fig. 7.12 The performance of the lookahead algorithm on small worlds networks of varied density, with TTL = 250

even in very big hierarchies. The other network types offer many more alternative paths, hence lead to a bigger value increase for the bigger network.

To confirm that it is the reduced need to revisit agents that allows the value increase with scale up for the random routing policies, we varied the network density of a scale free network and used random trails routing on the networks, while the overall network size was scaled up. The results are shown in Fig. 7.10. Notice that for the larger networks, random trails performed relatively better on denser networks (average density 4 or 8) than on sparser networks (average density 2), confirming the hypothesis. There is no significant difference in the average density 4 and average density 8 case, because the token already has sufficiently many options in the density 4 case to avoid most revisits.

The next set of experiments looked at the performance of the lookahead routing on small-worlds networks and hierarchies as the network density is increased (see Figs. 7.11–7.14). The network density makes relatively little difference to overall effectiveness, since these social networks have relatively low width, even when the density is quite low. The line labeled “0.2 0.7” uses a normal distribution of values

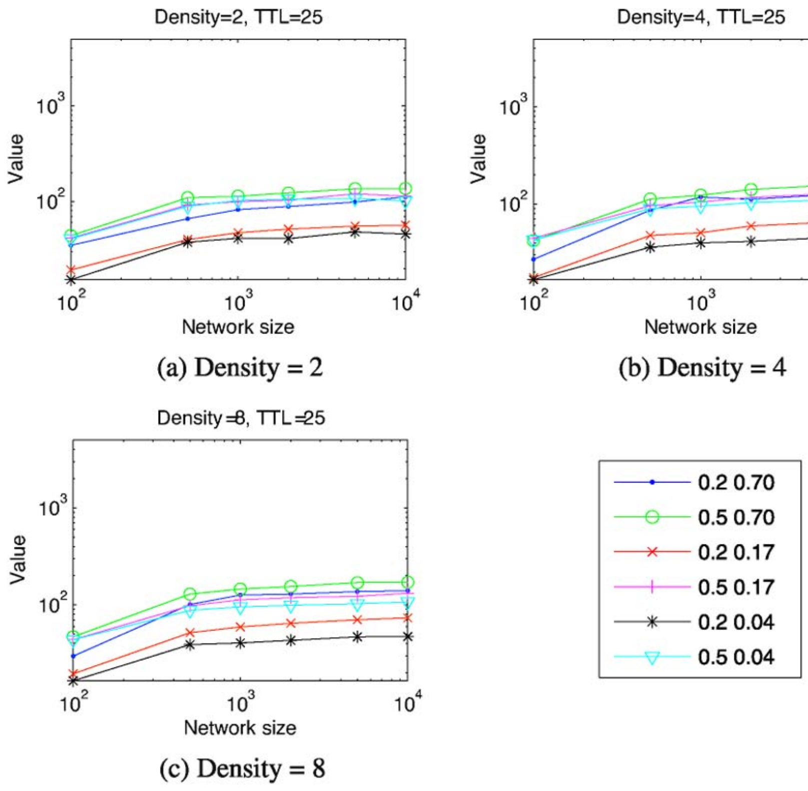


Fig. 7.13 The performance of the lookahead algorithm on hierarchies of varied density, with TTL = 25

with a mean value of 0.2 and a standard deviation of 0.7, and similarly for the other lines. As expected, when the mean is higher, the lookahead is able to find better agents to deliver the information to and when the standard deviation is higher, the lookahead exploits the high value agents efficiently.

Finally, we looked at how random routing performs when an agent might need multiple pieces of information. Two possibilities were modeled. First, when propagated information was noisy sensor readings, with agents potentially needing multiple readings to reduce uncertainty sufficiently. Second, when information was propagated about something that was changing over time, hence requiring new information occasionally. Measuring the average value received by the tokens was not an appropriate metric, in this case, because it incentivized behavior where there was maximal delay between delivering dynamic information to agents that needed it most, since that would be high value. Instead, the performance metric used was the sum of information needs over both agents and time. For this metric, lower is better, since lower implies agents needing information the most have received it earliest (and more often, in the dynamic case). Figure 7.15 shows the results. In the dynamic case, the relative advantage of lookahead over random policies increases

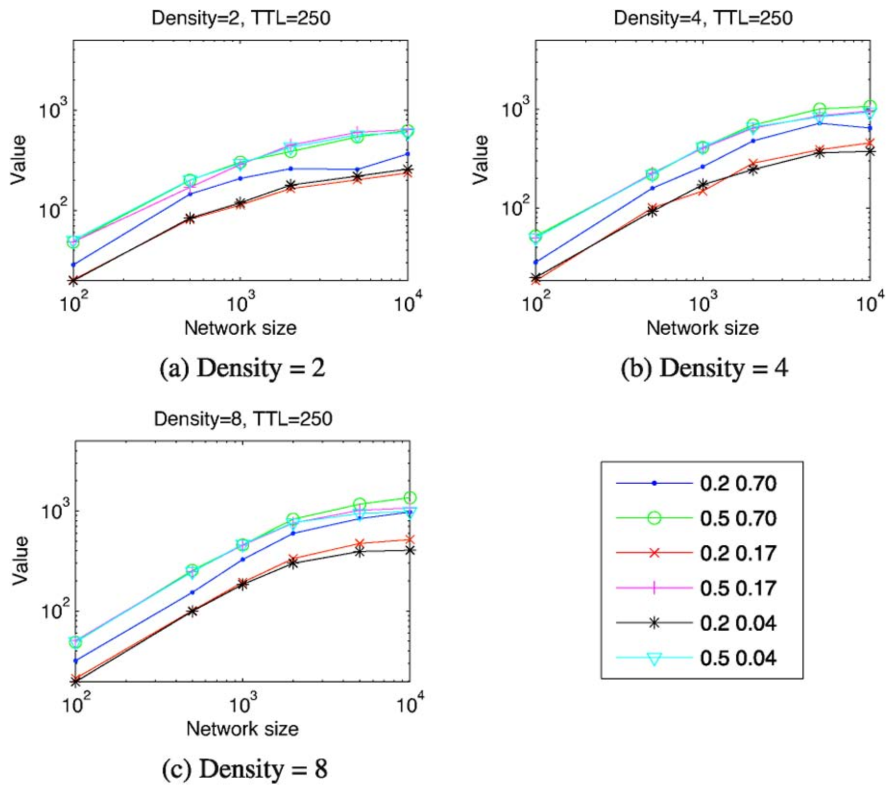


Fig. 7.14 The performance of the lookahead algorithm on hierarchies of varied density, with TTL = 250

almost linearly, since the lookahead policy can repeatedly exploit its search to find agents with highest value for the information. However, in the noisy case, the difference gets smaller over time because there is less value for the lookahead to exploit after the highest value agents have received the information. Notice that in this case, the random trails and random self avoid policies do not attempt to avoid repeated visits across tokens, only for the single token.

7.4 Related Work

The problem of communication in teams has been well-studied in a variety of fields. Approaches such as STEAM [14] and matchmakers [8] share knowledge about information requirements in order to reason about where to direct information. Gossip algorithms [2] and token passing algorithms [15, 17] use randomized local policies to share information and are thus particularly suited to large scale problems. To address the expense of synchronizing beliefs over teams, several techniques have been

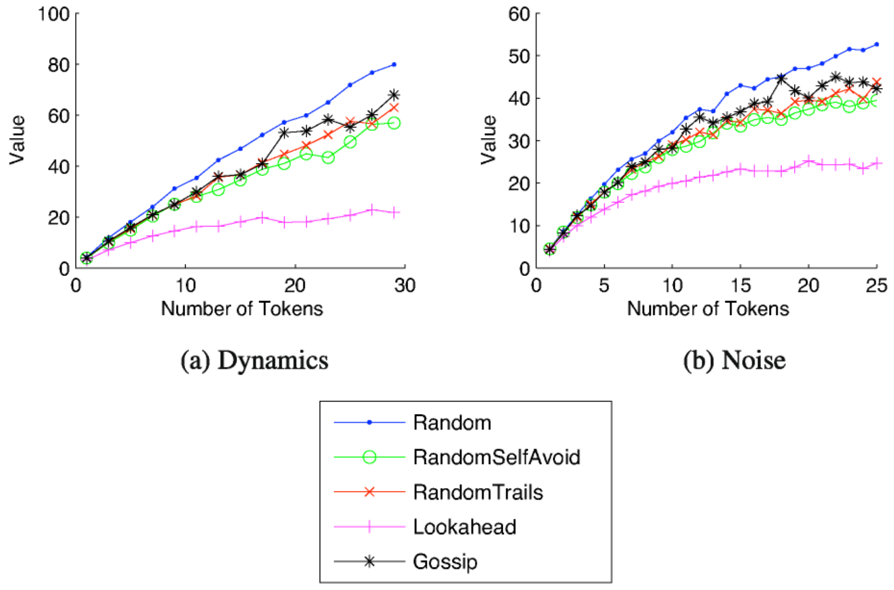


Fig. 7.15 The relative performance of random strategies when multiple pieces of information are required per agent, either due to noise or dynamics on a scale free network

developed in conjunction with decentralized Bayesian filtering techniques, including channel managers [1] and query-based particle filters [11].

A number of approaches to communication for multi-agent coordination have evolved around the concept of multi-agent partially observable Markov decision processes (POMDPs). Some approaches augment the decentralized problem with actions for synchronization [10], while others model communications as an explicit choice and seek to maximize the tradeoff between communication cost and reward [5, 18] to achieve goals such as minimizing coordination errors [12].

7.5 Conclusions and Future Work

This chapter extends previous work that established upper bounds on token-based propagation. Specifically, it was shown the that relative performance of random token-passing policies scaled well with overall network size and were actually relatively better when noisy. Overall random policies were found to perform relatively poorly on small-worlds networks, while performing well on scale-free and lattice networks. In addition, hierarchical networks were shown to be ill-suited to even optimal token-based information sharing algorithms. The empirical conclusions support the idea that for some domains, random information sharing policies will be a very reasonable approach to take.

In future work, we will apply these results to a variety of physical domains, including urban search and rescue and mobile mesh networking. Using these analysis methods, it should be possible to determine which information sharing methods are best suited to these domains, including if and when random policies should be used. In addition, by modeling the utility distributions of these domains, it may be possible to gain insight into the fundamental properties of real-world information sharing problems, in turn improving the information sharing algorithms that must address them. Further graph-theoretic and probabilistic analysis should yield tighter bounds on performance, and additional experiments can determine the optimality of other common information sharing algorithms such as classic flooding [6], gossiping [2], and channel filtering [1].

Acknowledgements This research has been funded in part by the AFOSR grant FA9550-07-1-0039 and the AFOSR MURI grant FA9550-08-1-0356. This material is based upon work supported under a National Science Foundation Graduate Research Fellowship.

References

1. Bourgault, F., Durrant-Whyte, H.: Communication in general decentralized filter and the coordinated search strategy. In: Proc. of FUSION'04 (2004)
2. Boyd, S., Ghosh, A., Prabhakar, B., Shah, D.: Randomized gossip algorithms. IEEE/ACM Trans. Netw. **14**(SI), 2508–2530 (2006). doi:10.1109/TIT.2006.874516
3. Chaimowicz, L., Kumar, V.: Aerial shepherds: Coordination among uavs and swarms of robots. In: 7th International Symposium on Distributed Autonomous Robotic Systems (2004)
4. Drury, J.L., Richer, J., Rackliffe, N., Goodrich, M.A.: Comparing situation awareness for two unmanned aerial vehicle human interface approaches. In: Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics (2006)
5. Goldman, C.V., Zilberstein, S.: Optimizing information exchange in cooperative multi-agent systems. In: Proceedings of the Second International Conference on Autonomous Agents and Multi-agent Systems (2003)
6. Heinzelman, W.R., Kulik, J., Balakrishnan, H.: Adaptive protocols for information dissemination in wireless sensor networks. In: Proc. of MobiCom'99 (1999)
7. Kuokka, D., Harada, L.: Matchmaking for information agents. In: Readings in Agents. Morgan Kaufmann, San Mateo (1995)
8. Kuokka, D., Harada, L.: Matchmaking for information agents. In: Readings in Agents. Morgan Kaufmann, San Mateo (1997)
9. Leary, C.C., Schwehm, M., Eichner, M., Duerr, H.P.: Tuning degree distributions: Departing from scale-free networks. Phys. A, Stat. Mech. Appl. **382**(2), 731–738 (2007)
10. Nair, R., Tambe, M.: Communication for improving policy computation in distributed pomdps. In: Proc. of AAMAS'04 (2004)
11. Rosencrantz, M., Gordon, G., Thrun, S.: Decentralized sensor fusion with distributed particle filters. In: Proceedings of the Conference on Uncertainty in AI (UAI) (2003)
12. Roth, M.: Execution-time communication decisions for coordination of multi-agent teams. PhD thesis, Robotics Institute, Carnegie Mellon Univ., Pittsburgh, PA (2007)
13. Schurr, N., Marecki, J., Tambe, M., Scerri, P., Levis, J., Kasinadhuni, N.: The future of disaster response: Humans working with multiagent teams using DEFACTO. In: AAAI Spring Symposium on Homeland Security (2005)
14. Tambe, M.: Agent architectures for flexible, practical teamwork. In: National Conference on AI (AAAI97), pp. 22–28 (1997)
15. Velagapudi, P., Prokopyev, O., Sycara, K., Scerri, P.: Maintaining shared belief in a large multiagent team. In: Proceedings of FUSION'07 (2007)

16. Velagapudi, P., Prokopyev, O., Sycara, K., Scerri, P.: An analysis of information sharing in large teams. In: Proceedings of AAMAS'09 (2009)
17. Xu, Y., Scerri, P., Yu, B., Okamoto, S., Lewis, M., Sycara, K.: An integrated token-based algorithm for scalable coordination. In: AAMAS'05 (2005)
18. Xuan, P., Lesser, V., Zilberstein, S.: Communication decisions in multi-agent cooperation: Model and experiments. In: Proceedings of the Fifth International Conference on Autonomous Agents (2001)

Chapter 8
Self-Organized Criticality of Belief Propagation in Large Heterogeneous Teams

Robin Glinton, Praveen Paruchuri, Paul Scerri, and Katia Sycara

Summary Large, heterogeneous teams will often be faced with situations where there is a large volume of incoming, conflicting data about some important fact. Not every team member will have access to the same data and team members will be influenced most by the teammates with whom they communicate directly. In this paper, we use an abstract model to investigate the dynamics and emergent behaviors of a large team trying to decide whether some fact is true. Simulation results show that the belief dynamics of a large team have the properties of a Self-Organizing Critical system. A key property of such systems is that they regularly enter critical states, where one additional input can cause dramatic, system wide changes. In the belief sharing case, this criticality corresponds to a situation where one additional sensor input causes many agents to change their beliefs. This can include the entire team coming to a "wrong" conclusion despite the majority of the evidence suggesting the right conclusion. Self-organizing criticality is not dependent on carefully tuned parameters, hence the observed phenomena are likely to occur in the real world.

8.1 Introduction

The effective sharing and use of uncertain information is key to the success of large heterogeneous teams in complex environments. Typically, noisy information is collected by some portion of the team and shared via the social and/or physical networks connecting members of the team [2]. Each team member will use incoming, uncertain information and the beliefs of those around them to develop their own beliefs about relevant facts. For example, in the initial reaction to a disaster, members of the response team will form beliefs about the size, scope, and key features of the disaster. Those beliefs will be influenced both by what they sense in the environment and what they are told by others in the team. Human teams have developed

This research has been sponsored by AFOSR grants FA95500810356 and FA95500710039.

R. Glinton (✉) · P. Paruchuri · P. Scerri · K. Sycara
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA
e-mail: rglinton+@andrew.cmu.edu

processes and strategies for managing this messy information and acting without necessarily having convergent beliefs [8]. However, in heterogeneous teams, where team members are not exclusively humans, but may be intelligent agents or robots, and novel network structures connect the team members, we cannot assume that the same tactics will help convergence or that undesirable and unexpected effects will not be observed. Thus, before such teams are deployed in important domains, it is paramount to understand and potentially mitigate any system-wide phenomena that affect convergence. There have been previous attempts in the scientific literature to describe the information dynamics of complex systems, however, due to the complexity of the phenomena involved, mathematical formulations have not been expressive or general enough to capture the important emergent phenomena.

To investigate the dynamics and emergent phenomena of belief propagation in large heterogeneous teams, we developed an abstracted model and simulator of the process. In the model, the team is connected via a network with some team members having direct access to sensors and others relying solely on neighbors in the network to inform their beliefs. Each agent uses Bayesian reasoning over beliefs of direct neighbors and sensor data to maintain a belief about a single fact which can be *true* or *false*. The level of abstraction of the model allows for investigation of team level phenomena decoupled from the noise of high fidelity models or the real-world, allowing for repeatability and systematic varying of parameters. Simulation results show that the number of agents coming to the correct conclusion about a fact and the speed of their convergence to this belief, varies dramatically depending on factors including network structure and density and conditional probabilities on neighbor's information. Moreover, it is sometimes the case that significant portions of the team come to have either no strong belief or the wrong belief despite overwhelming sensor data to the contrary. This is due to the occasional reinforcement of a small amount of incorrect sensor data from neighbors, *echoing* until correct information is ignored.

More generally, the simulation results indicate that the belief propagation model falls into a class of systems known as Self Organizing Critical (SOC) systems [1]. Such systems naturally move to states where a single local additional action can have a large system wide effect. In the belief propagation case, a single additional piece of sensor data can cause many agents to change belief in a cascade. We show that, over an important range of conditional probabilities, the frequency distribution of the sizes of cascades of belief change (referred to as *avalanches*) in response to a single new data item follows a power law, a key feature of SOC's. Specifically, the distribution of avalanche sizes is dominated by many small avalanches and exponentially fewer large ones. Another key feature of SOCs is that the critical behavior is not dependent on finely tuned parameters, hence we can expect this criticality to occur often, in real-world systems. The power law suggests that large avalanches are relatively infrequent, however when they do occur, if sparked by incorrect data, the result can be the entire team reaching the wrong conclusion despite exposure to primarily correct data. In many domains such as sensor networks in the military, this is an unacceptable outcome even if it does not occur often. Notice that this phenomena was not revealed in previous work, because the more abstract mathematical models were not sufficiently expressive. Finally, our simulation results show that the inclusion of humans resulted in fewer and smaller avalanches.

8.2 Self-Organized Criticality

Self Organized Criticality is a property of a large number of many body complex systems identified in the pioneering work of Bak et al., characterized by a power law probability distribution in the size of cascades of interaction between system constituents. In such systems, these cascades are frequently small and punctuated by very large cascades that happen much less frequently. Self Organized Criticality (SOC) has been used in an attempt to explain the punctuated equilibrium exhibited by many systems [3, 7]. The defining characteristic of systems that exhibit SOC Systems is an attractor in the system state space which is independent of parameter values and initial conditions. Such an attractor, typically called a *critical state*, is characterized by a lack of a characteristic scale in the interactions of system constituents. For a system in a critical state long range correlations exist, meaning that perturbations caused by individual system constituents can have system-wide effects. There are many systems which exhibit criticality, but require fine tuning of a global control parameter to enter the critical state. For a system that exhibits SOC, the system will spontaneously enter a critical state due to the intrinsic interactions within the system, independent of system parameters [1].

Systems which exhibit SOC share certain fundamental characteristics. Chief among these are large numbers of constituents interacting on a fixed lattice. Furthermore, for each unit in the system, the number of its neighbors on the lattice which it interacts with is typically a small percentage of the constituents of the system. There are three primary system influences that lead to systems which exhibit SOC. The first is an external drive which attempts to change the state of the individual. Examples include market forces which influence an individual's stock purchasing decisions in economic models or gravity acting on the particles of a sandpile. The second factor is a resistance of the individual to change. In the economic model, this would be the individual's cognitive resistance to purchasing a stock, while in the sandpile, friction would play this role. The last factor is a threshold in the local resistance at which the individual relents and changes (toppling grains in the sandpile or the point at which an individual is satisfied that a stock is a worth-while purchase).

These three factors interact to create the conditions necessary for the characteristic scale-free dynamics of SOC. For the power law dynamics of SOC, external events can have temporal effects on the system well beyond the instant at which the external influence acted, which in turn results in long range temporal and spatial synchronization. In a system which exhibits SOC the local resistance is naturally balanced such that, most of the time, most individuals are below the threshold to change and building towards it (they are remembering external events). The result is that most of the time only a few agents will change and propagate information. However, infrequently, many agents will simultaneously reach the threshold and a massive avalanche will occur. For a system in which the local resistance is too large, the system will quickly dissipate local perturbations. We refer to this as a system with a *static* regime. In the *static* regime, local perturbations do not lead to system-wide effects. In systems where, local resistance is too low, local perturbations are quickly propagated and massive avalanches are frequent. We refer to this as a system with an *unstable* regime.

To make the SOC concept clear, consider the concrete example of dropping sand onto different parts of a sandpile. If friction is low corresponding to the first regime, dropping sand on the sandpile would always result in many small scale local "avalanches" of toppling sand. Conversely, if friction is too high, grains of sand will not be able to start a flow and avalanches will never occur. However, when friction is balanced between these two extremes, some parts of the sandpile would initially resist toppling due to the addition of sand. Sand would collect on many different parts of the sandpile until a critical threshold was reached locally where the weight of the sand would exceed the friction forces between grains. The friction results in system "memory" which allows for many parts of the pile to reach this critical state simultaneously. As a result, a single grain of sand acting locally could set off a chain reaction between parts of the pile that were simultaneously critical and the entire pile would topple catastrophically. This is why power laws are typical in the statistics of "avalanches" in SOC systems.

8.3 Belief Sharing Model

Members of a team of agents, $A = \{a_1, \ldots, a_N\}$, independently determine the correct value of a variable F, where F can take on values from the domain $\{true, false, unknown\}$. Note that the term agent is generic in the sense that it represents any team member including humans, robots, or agents. We are interested in large teams, $|A| \geq 1000$, that are connected to each other via a network, K. K is a $N \times N$ matrix where $K_{i,j} = 1$, if i and j are neighbors and 0 otherwise. The network is assumed to be relatively sparse, with $\sum_j K_{i,j} \approx 4, \forall i$. There are six different network structures we consider in this paper: (a) Scalefree, (b) Grid, (c) Random, (d) Smallworld, (e) Hierarchy and (f) HierarchySLO. A comprehensive definition of networks a–d can be found in [6]. A *hierarchical* network has a tree structure. A *hierarchySLO* has a hierarchical network structure where the sensors are only at the leaves of the hierarchy. Some members of the team, $H \subset A$ are considered to be humans. Certain members of the team, $S \subset A$ with $|S| \ll |A|$, have direct access to a sensor. We assume that sensors return observations randomly on average every second step. A sensor simply returns a value of *true* or *false*. Noise is introduced into sensor readings by allowing a sensor to sometimes return an incorrect value of F. The frequency with which a sensor returns the correct value is modeled as a random variable R_s which is normally distributed with a mean μ_s and a variance σ_s^2. Agents that are directly connected to sensors incorporate new sensor readings according to the belief update equation, given by (8.1), this is the version used to calculate the probability that F is true to incorporate a sensor reading that returns *false*:

$$P'(b_{a_i} = true) = \frac{A}{B + C} \tag{8.1}$$

$$A = P(b_{a_i} = true)P(s_{a_i} = false|F = true) \tag{8.2}$$

$$B = (1.0 - P(b_{a_i} = true))_t P(s_{a_i} = false|F = false) \tag{8.3}$$

$$C = P(b_{a_i} = true)P(s_{a_i} = false|F = true) \tag{8.4}$$

Agents use (8.5) to incorporate the beliefs of neighbors:

$$P'(b_{a_i} = true) = \frac{D}{E + G} \tag{8.5}$$

$$D = P(b_{a_i} = true)P(b_{a_j} = false|F = true) \tag{8.6}$$

$$E = (1.0 - P(b_{a_i} = true))P(b_{a_j} = false|F = false) \tag{8.7}$$

$$G = P(b_{a_i} = true)P(b_{a_j} = false|F = true) \tag{8.8}$$

where $P(b_{a_i})$ gives the prior belief of agent a_i in the fact F and $P(s_{a_i}|F)$ gives the probability that the sensor will return a given estimate of the fact (*true* or *false*) given the actual truth value of the fact. Finally, $P(b_{a_j}|F)$, referred to interchangeably as the belief probability or the conditional probability or *CP*, gives a measure of the credibility that an agent a_i assigns to the value of F received from a neighbor a_j given F. Humans are assigned a much larger credibility than other agents. That is, for $a_h \in H$ and $a_k \notin H$, $P(b_{a_h}|F) \gg P(b_{a_k}|F)\ \forall_{i,h,k}$. Humans also have a higher latency between successive belief calculations.

Each agent decides that F is either *true*, *false*, or *unknown* by processing its belief using the following rule. Using T as a threshold probability and ϵ as an uncertainty interval the agent decides the fact is *true* if $P(b_{a_i}) > (T + \epsilon)$, *false* if $P(b_{a_i}) < (T - \epsilon)$, and *unknown* otherwise. Once the decision is made, if the agents decision about F has changed, the agent reports this change to its neighbors. Note that in our model, neither $P(b_{a_i})$, the probability that F is *true* according to agent a_i, or the evidence used to calculate it, is transmitted to neighbors. Communication is assumed to be instantaneous. Future work will consider richer and more realistic communication models.

8.4 System Operation Regimes

This section gives insight into the relationship between the local parameter regimes discussed in Sect. 8.2 and the parameters of the belief model introduced in Sect. 8.3.

When $P(b_{a_j}|F)$, the credibility that agents assign to neighbors, is very low, agents will almost never switch in response to input from neighbors so no propagation of belief chains occurs. This is the *static* regime. The *unstable* regime occurs in the belief sharing system for high values of $P(b_{a_j}|F)$. In this regime, agents take the credibility of their neighbors to be so high that an agent will change its belief

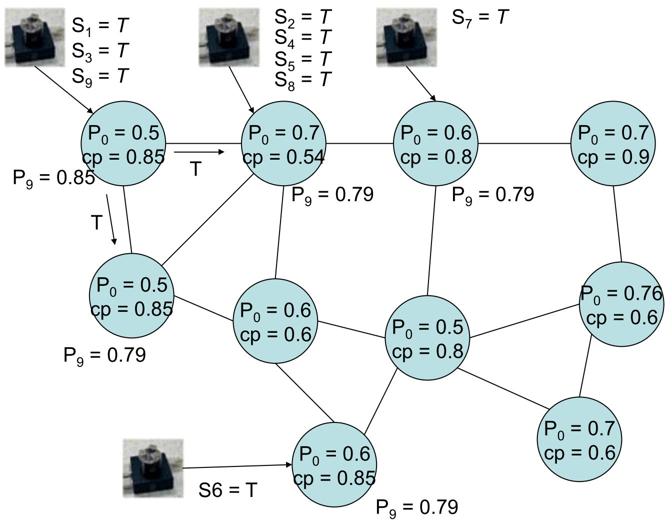


Fig. 8.1 The system in a critical state

based on input from a single neighbor and large chains of belief propagation occur frequently.

The SOC regime occurs for values of $P(b_{a_j}|F)$ between the two other ranges. Within the SOC regime, most of the time there are small avalanches of belief propagation. Much more infrequently there are large system wide avalanches. The large avalanches occur because in this regime $P(b_{a_j}|F)$ acts as a local resistance to changing belief when receiving a neighbors belief. As described in Sect. 8.3 for an agent in the *unknown* state, there are thresholds on $P(b_{a_i})$ above which the agent will change to the *true* state, (and vice versa). In certain circumstances given the data received by agents thus far, many agents will simultaneously reach a state where they are near the threshold. In this case, a single incoming piece of data to any of the agents will cause an avalanche of belief changes. Figures 8.1 and 8.2 give an example. In the figure, P_0 gives the prior on each agent and S_t a sensor reading received at time step t. Figure 8.1 shows a critical state where many agents are simultaneously near the threshold of $P = 0.8$ as a result of sensor readings $S_1 \ldots S_9$. Figure 8.2 shows what happens when one additional sensor reading, S_{10} arrives at time $t = 10$. The numbered arrows shows the ordering of the chain of changes as every message passed causes each agent to change its belief since they all need one additional message to rise above the threshold. The result is a system-wide propagation of belief changes.

8.5 Simulation Results

We performed a series of experiments to understand the key properties and predictions of the system. First, we conducted an experiment to reveal that the system

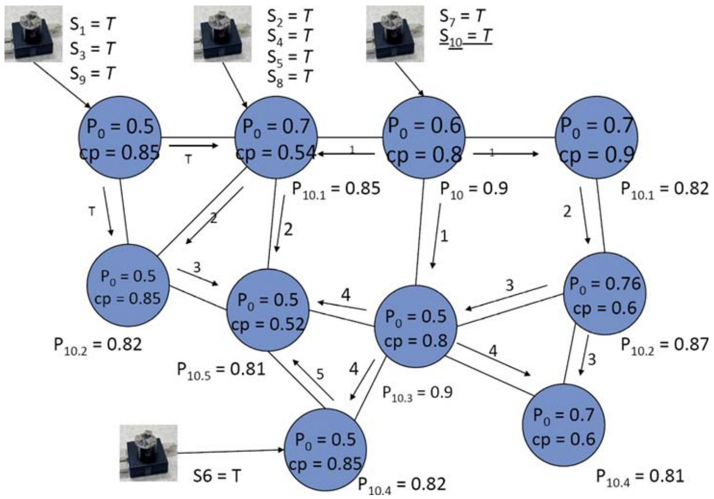


Fig. 8.2 One additional reading causes a massive avalanche of changes

exhibits SOC. The result of this experiment is shown in Fig. 8.3. The x-axis gives the length of a cascade of belief changes and the y-axis gives the frequency with which a chain of that length occurred. The plot is a log-log plot so we expect a power law to be evident as a straight line. The underlying network used to produce the figure was a scale-free network. The 7 lines in the figure correspond to *CP* values (judgement of a neighbors credibility) of 0.52, 0.54, 0.56, 0.58, 0.6, 0.7, and 0.8. Each point in the graph is an average over 500 runs. While *CP* values from 0.52 to 0.6 exhibit a power law, values greater than 0.6 do not. This is because if agents have too much faith in their neighbors (large values of *CP*) then all agents change their beliefs immediately upon receiving input from a neighbor and the system converges quickly. This is the *unstable* regime. While we produced this figure solely for a scalefree network due to space constraints, our experiments show that this trend is true across all networks.

The existence of the SOC regime for a particular range of credibility is important for real world systems because the massive avalanches of belief changes could be propagating incorrect data. Furthermore, this occurs rarely and would be difficult to predict. It would be desirable to avoid or mitigate such an eventuality. Understanding the particular parameter responsible and the ranges over which this phenomena occurs allows system designers to reliably do this.

Having revealed the SOC regime in the model, we wanted to obtain a deeper understanding of the relationship between the parameters of the model and the dynamics of belief propagation in this regime. We concisely capture the dynamics of belief cascades using a metric that we refer to as center of mass. Center of mass is defined as:

$$\frac{\sum_{i=1toN}(AvalancheFrequency * AvalancheSize)}{\sum_{i=1toN} AvalancheFrequency}$$

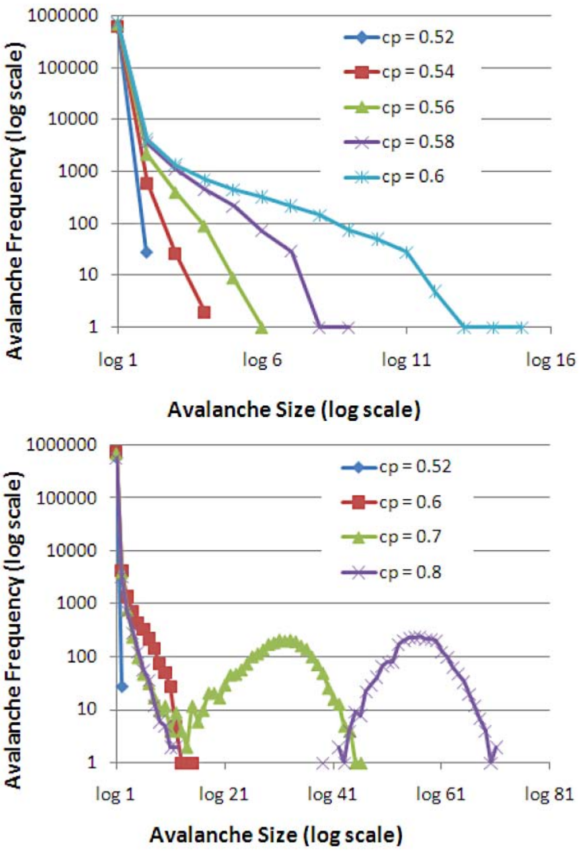


Fig. 8.3 The *straight line* plots mean that the avalanche distribution follows a power law for the associated parameter range

and can be thought of as the expected length of a chain of belief changes. The next experiment studies the relationship between *CP* (assessment of a neighbors credibility) and center of mass across the six different network types. Experiments had 1000 agents and 6 different networks. Results are averaged over 500 runs. Figure 8.4 gives the result of these experiments. The x-axis gives *CP* and the y-axis captures the averaged center of mass. The figure shows a uniform trend across all the networks that when agents have more faith in their neighbors, the center of masses of the avalanches become larger. However, this effect is much less extreme for the networks that are hierarchies. This is because in a hierarchy there are relatively few paths between random pairs of nodes along which belief chains can propagate (all paths go through the root node for example). The effect is particularly prevalent in the networks like the Scalefree and Smallworld networks, because these networks all have a large number of paths between random pairs of nodes.

Next, we conducted an experiment to understand the effect of increasing the number of sensors, and thus the amount of information available to the system, on belief dynamics. The x-axis of Fig. 8.5, represents the number of agents that have sensors (out of 1000) while the y-axis represents the center of mass. Each point in the graph

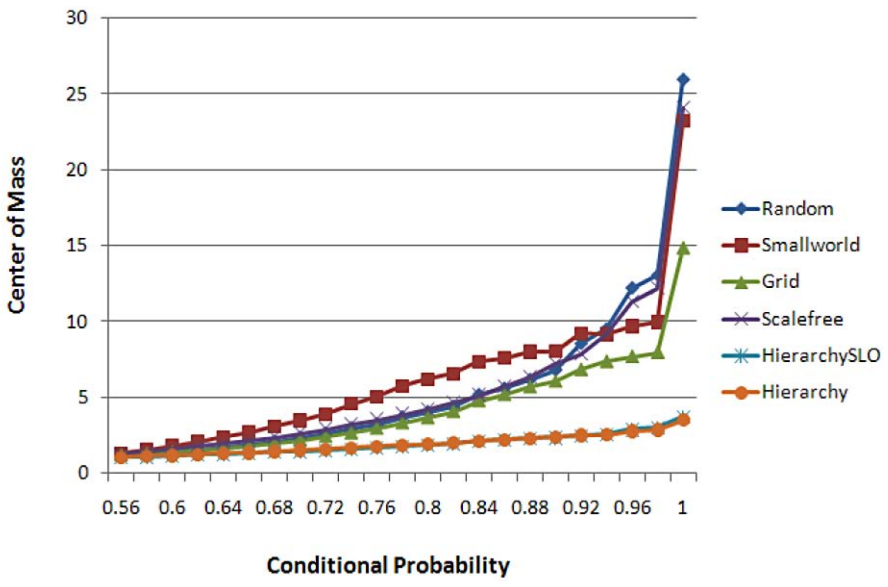


Fig. 8.4 Conditional Probability vs Center of Mass Graph

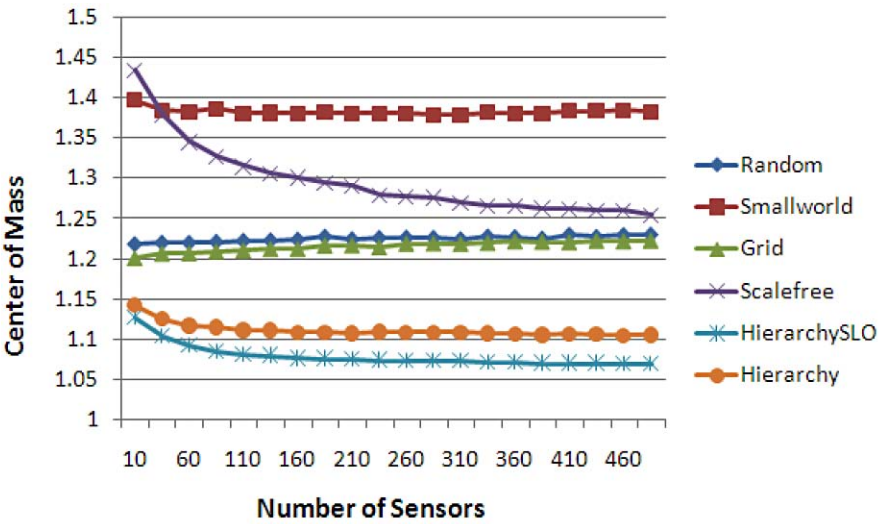


Fig. 8.5 Effect of the increasing the number of sensors on center of mass

represents an average of 500 runs over 6 different networks. For this experiment, *CP* is set to 0.56. The graph shows that as the number of sensors increases the center of mass across all the networks types decreases (or increases by only a small amount). A particularly sharp drop is observed for scalefree networks. This is because agents

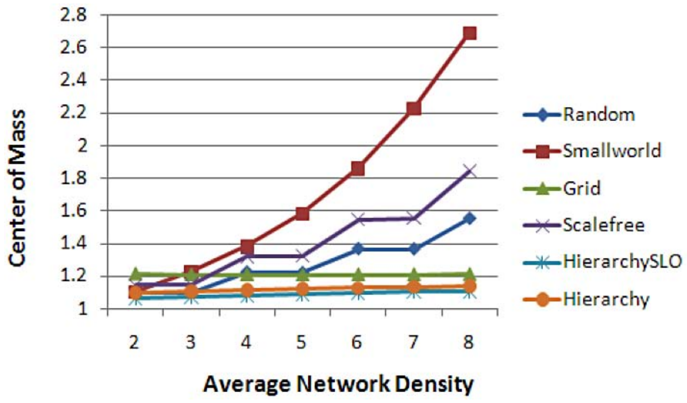


Fig. 8.6 Density vs. Center of mass plot

with sensors have more access to information and are thus more certain in their own beliefs and less likely to change belief in response to information from neighbors. This means that cascades of belief changes are smaller. Conventional wisdom dictates that if information received is primarily accurate then more information is better. We see here, however, that increasing the number of individuals that have access to a sensor can discourage the desirable sharing of information. This could be a problem if some sensors in a system are receiving particularly bad data and thus need information from their neighbors.

The next experiment looks at the impact of the average network density on the center of mass where average network density $= \frac{\sum_{a_i} K_{i,j}}{|A|}$, which is the average number of links that an agent has in the network. The x-axis of Fig. 8.6 shows the average network density increasing from 2 to 8 and the y-axis gives the center of mass. As the network density increases, the center of mass either increases (for scalefree, smallworld, and random) or remains constant (for grid, hierarchySLO and hierarchy). This is due to the fact that each agent can influence more neighbors and hence lead to the formation of bigger avalanches resulting in a higher or at least equal center of mass.

Next, we conducted an experiment to study the effect that humans have on belief propagation within the system. The model of a human used is described in Sect. 8.3. Human agents have higher credibility and are slower to change beliefs. The x-axis of Fig. 8.7 shows the percentage of humans in the network increasing from 0 to 100% and the y-axis gives the center of mass. Figure 8.7 shows that at first increasing the percentage of humans in a team up to about 20% increases the center of mass of the length of avalanches. The length of avalanches decreases when more than 20% of the agents are human. This effect was noted across all network types. The higher credibility of humans tends to encourage changes in others who receive human input, and thus humans can encourage avalanches. Conversely, the long change latency of humans has the effect of impeding avalanche formation. When the percentage of humans is below 20%, the avalanche enhancing effect of the high human

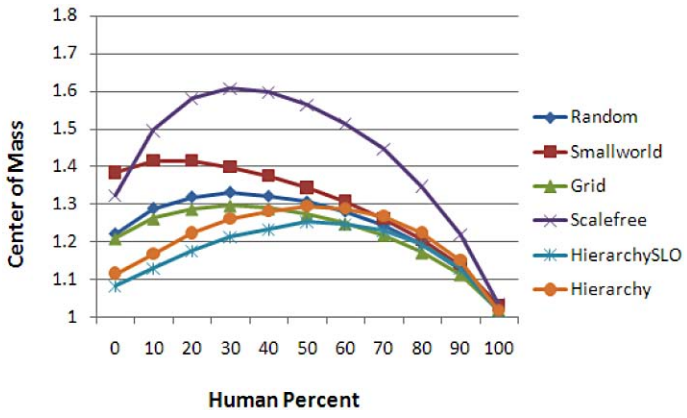


Fig. 8.7 Human percent vs. Center of mass plot

CP predominates, while above this value the impedance effect caused by the long latency predominates.

The previous experiments were aimed at understanding the dynamics of belief propagation and the parameters that impact it. However, they did not reveal the nature of the convergence. That is, how many agents actually converge to the correct belief. Our next experiment was conducted to study the impact of network type on the nature of convergence. There are three graphs presented in Fig. 8.8 corresponding to random, Small world and Scale free networks, respectively. Each graph represent a total of 2500 simulations of the system. The sensor reliability in all the three graphs is fixed to 0.75. A sensor reliability of 0.75 would mean that the sensor gives a true reading with a 0.75 probability. The x-axis gives sizes of clusters of agents that reached a correct conclusion and the y-axis gives the number of simulations out of the 2500 in which such a cluster occurred. The plot corresponding with the scale free network shows a particularly interesting phenomena. There are two humps in the graph. The larger of the two shows that large numbers of agent converge to the correct conclusion quite often. However, the smaller hump shows that there is a smaller but still significant number of simulations where only very small clusters of agents converge to the correct result. This is a startling result because it says that despite having the same number of sensors with the same reliability the system can come to drastically different conclusions. This is probably due to the nature of the information (correct or incorrect) that started an SOC avalanche which led to convergence. This has important implications in many domains where scale free networks are becoming increasingly prevalent.

The previous experiment gave us insight into the nature of convergence of the system and its relationship to system parameters and network type. However, all previous experiments were conducted using 1000 agents and we wanted to find out if the scale of the system, in terms of number of agents, had any effect on information propagation. In addition, we wanted to understand the sensitivity of the system to changes in system scale, given a particular communication network structure. To this

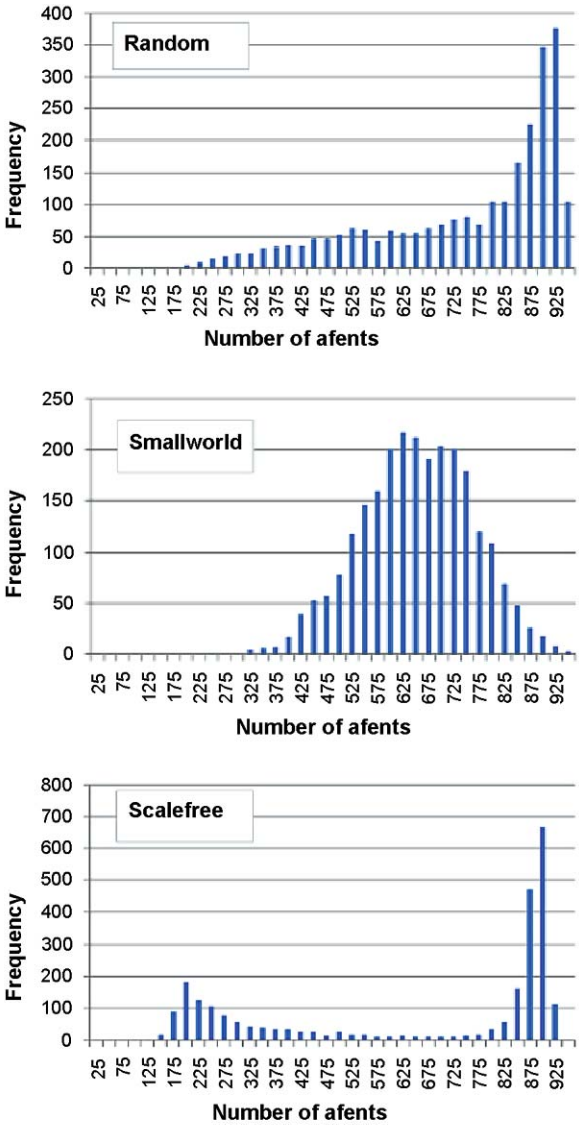


Fig. 8.8 Correct conclusions as a function of network type

end, we conducted an experiment in which the size of the agent team and network type were both varied and the percentage of agents within the system that converged to *true* was recorded. For this experiment, 10% of the agents had a sensor up to a maximum of 200 sensors. Sensor reliability was set to 0.75 with $CP = 0.8$, $\epsilon = 0.3$ and $P(b_{a_i}) = 0.5$. Figures 8.9–8.14 give the results of this experiment. For all graphs, the x-axis gives the percentage of agents within the system that converged to *true* and the y-axis gives the number of simulation runs (out of 500) in which the corresponding percentage of agents converged to true. Note that there is an implied

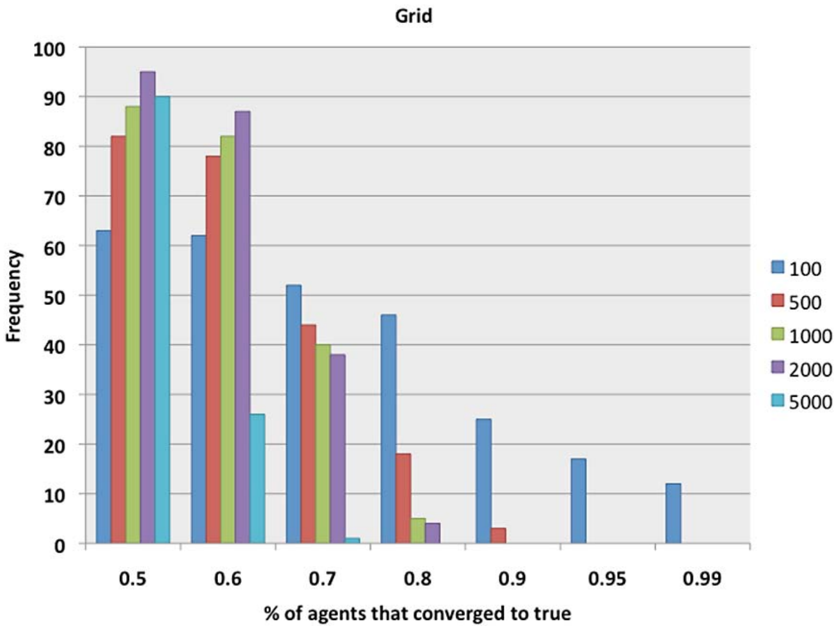


Fig. 8.9 Frequency for which a certain percentage of agents converge to true over different network sizes. The agents use a Grid communication network

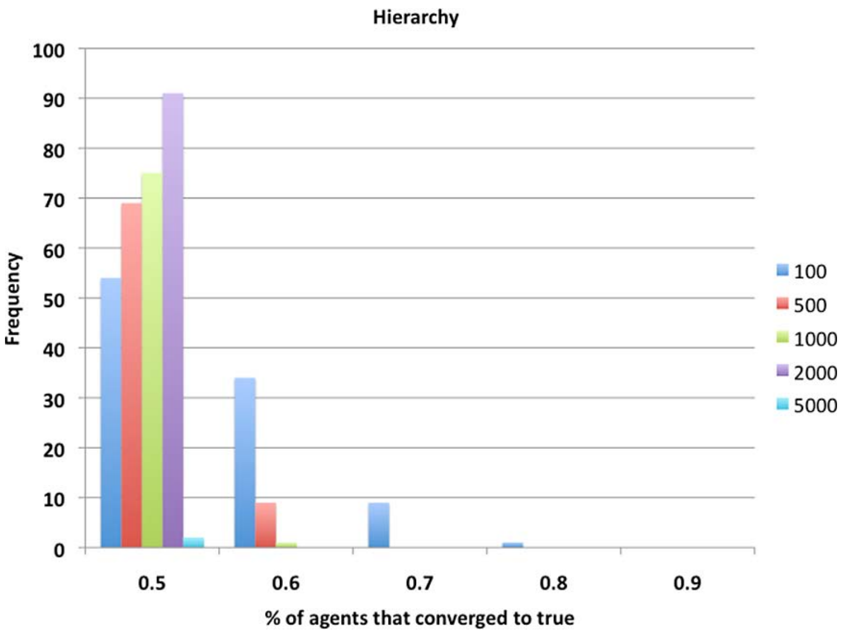


Fig. 8.10 Frequency for which a certain percentage of agents converge to true over different network sizes. The agents use a Hierarchy for communication

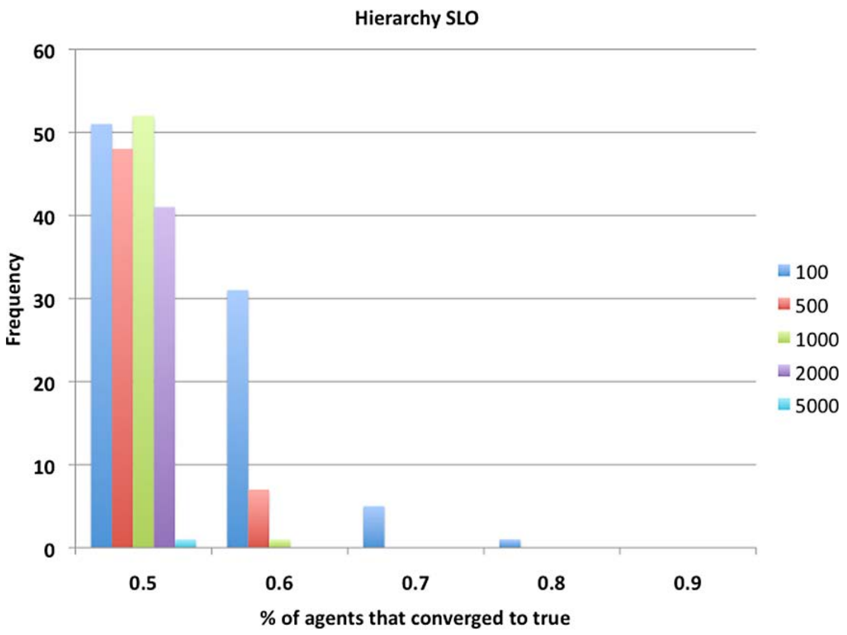


Fig. 8.11 Frequency for which a certain percentage of agents converge to true over different network sizes. The agents use a HierarchySLO communication network

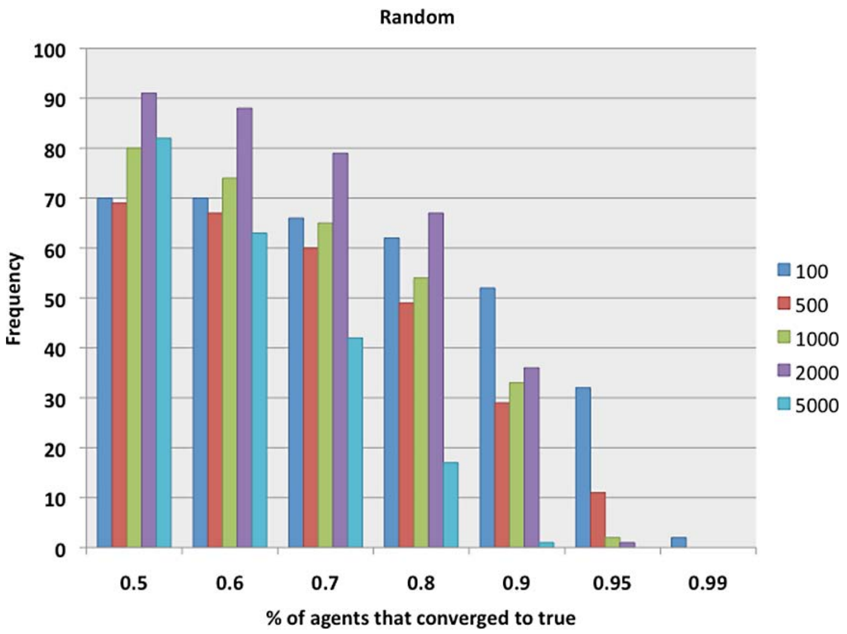


Fig. 8.12 Frequency for which a certain percentage of agents converge to true over different network sizes. The agents use a Scalefree communication network

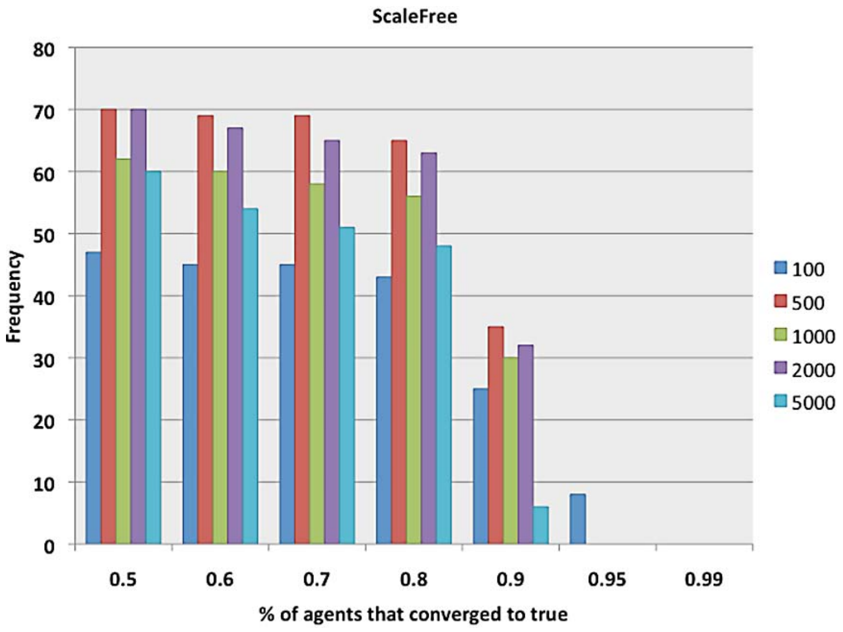


Fig. 8.13 Frequency for which a certain percentage of agents converge to true over different network sizes. The agents use a Random network

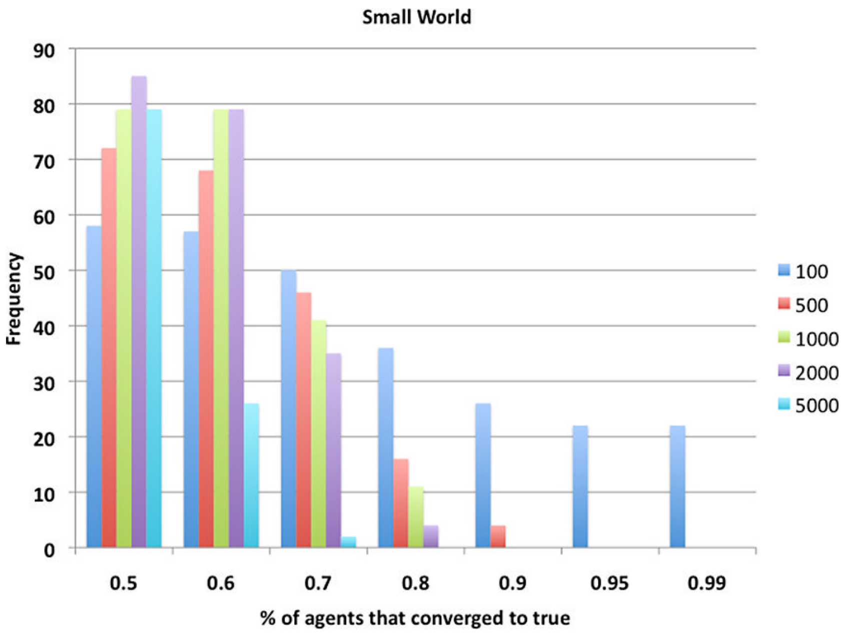


Fig. 8.14 Frequency for which a certain percentage of agents converge to true over different network sizes. The agents use a small-world ring communication network

temporal progression in the graphs. Specifically, if in a particular simulation run the maximum number of agents that converged to true was 90%, this implies that there was an intermediate time step within the simulation run at which 50% of the agents had converged to *true*. For each unit on the x-axis, the corresponding histogram bars correspond (from left to right on the plot) to 100, 500, 1000, 2000, and 5000 agents.

Figure 8.9 gives the results of the scale experiment when the agents communicated using a Grid network (network types are discussed in Sect. 8.3). This graph shows that scale has a significant impact when a grid is used as a communication network. For the larger systems, with greater than 100 agents, the system infrequently has more than 70% of the agents converging to *true*. This is not surprising because on a grid each agent can only influence a small number of other agents in the system and hence information propagation cascades are easily dampened. What is surprising is that this trend is reversed below 70%. That is below this threshold the larger the system, the more frequently it attains the corresponding percentage of convergence. This cannot be explained by the network structure and must then be a result of the large-scale interaction of the agents.

Figures 8.10 and 8.11 give the results for the two communication networks that are organized as tree structures (Hierarchy and HierarchySLO). Both graphs show that scale has a profound effect on information propagation through a hierarchy. This is because a tree structure isolates large portions of the system from each other. In the extreme case, any information that originates in roughly one half of the nodes in the system must pass through the root node to reach nodes in the other half of the system. This bottleneck acts to dampen the majority of the cascades in the system because the root node is solely responsible for passing this information between the two halves of the system. This means that if the root node makes a mistake, it has a disproportionately large effect. This effect is more pronounced at larger scales because there is always a single root node in all cases but the size of the isolated sections of the system do scale.

To produce Fig. 8.12 a communication network with a Scalefree structure was used. This figure shows, as intuition would dictate, that scale has little effect on convergence for this type of network. This is because the hubs (few agents with high connectivity) act as cascade multipliers, widely disseminating information in the system. However, it is surprising that for the Grid structure (Fig. 8.9) the system more frequently reaches higher percentages of agents converging to *true*. This can be again explained by the hubs present in the system. Just as in the case of the hierarchy networks, the negative decisions of the hubs have a disproportionately large effect on the system resulting in lower convergence to the correct result.

Figure 8.13 shows the results when a Random network was used. The results are very similar to those for when a Scalefree communication network was employed. This is not surprising because they have similar properties, for example similar average network width. Random connections mean that independent of scale there is a high probability of a short path between any two nodes in the system independent of scale.

Figure 8.14 was produced for a system employing a communication network with the properties of a small-world ring network. The sensitivity of the system employing the small-world ring system to scale seems to share properties with both the

Grid-network and the Random network. Like the system employing the Grid network, the percentage of agents converging to *true*, seems to be much higher for 100 agents than when greater numbers of agents are used. However at other scales the small world ring system behaves in a similar fashion to the Random network. This is reasonable because the small-world ring consists of primarily structured connections similar to the Grid network, however, unlike the grid network the small world ring contains of a few random connections.

8.6 Related Work

Many studies have been conducted to discover systems which exhibit SOC. SOC has provided an explanation for the distribution of earthquake intensity [7]. Drossell showed that forest fire models exhibit SOC dynamics [3]. SOC has been used to explain the avalanche dynamics of a wide range of "pile" models including, sandpiles [1], and rice grains [4]. In addition, a number of researchers have attempted to use SOC to describe models of evolution [10].

In all of these treatments, simulations of simple models of networked nodes are used to investigate the self-organized criticality of the respective systems. In these treatments, the nodes are homogeneous and change there state according to a simple rule that is the same for all nodes. The model described in this paper differs from these others through the use of nodes which are heterogeneous in their decisions about changing state. The heterogeneity is a result of the distribution of priors across the system. In addition, these distributions of probabilities introduces a dynamic thresholding that is not present in these other models. This means that in the model described in this paper, each node makes a decision about changing state using a unique rule.

The model used in the paper, due to Watts [9], is similar to our own in that it also uses heterogeneous nodes which change state based on a distribution of thresholds to change. There are, however, a number of key differences between our work and that of Watts. The first is that, in our model, we introduce a simple model of a human. The technical difference introduced by the introduction of human nodes is that some nodes in our system have increased latencies before changing with respect to changes made by their neighbors. Unlike Watts, we explore the effect of these latencies which includes novel system dynamics which fall outside of self-organized criticality. Another key difference is that our model is a three state model (*true*, *false*, *unknown*), while Watts uses a two state model (*1*, *0* equivalent to *true*, *unknown* in our model). The result of this is an increase in complexity of the decision rules used by nodes which also introduces regimes of system dynamics outside of self-organized criticality. Finally, in the Watts model once an agent changes to the *true* state, it remains in that state for the remainder of the simulation, this is not the case in our model. This has a dramatic effect on the information propagation in the system as well as on the types of emergent effects that are possible. Consequently, in the case of Watts' model only avalanches of *true* states are possible. In [5], Motter discusses cascading failures in electrical networks. Motter investigates cascades due to removing nodes as opposed to cascades of decisions as in our work. Unlike the work discussed here, Motter does not examine the relationship between system properties and the distribution and sizes of cascades.

8.7 Conclusions and Future Work

This work demonstrated that a system of belief sharing agents exhibits SOC dynamics for a particular range of agent credibility. SOC dynamics are likely to exist in real-world systems because of the relatively wide parameter range over which this regime occurs. In addition, we discovered that humans can have a dramatic effect on information propagation. Small numbers of humans encourage information propagation in the system while large numbers of humans inhibit information propagation. Finally, we found that for scale free networks, SOC dynamics often propagate incorrect information widely even when much correct information is available.

In the future, we plan to investigate the effects of using a more complex model of the belief maintained by an agent, including multivariate beliefs. Of particular interest are multivariate beliefs where variables are conditionally dependent. In real world systems, beliefs are often abstracted as they move up a hierarchy, we plan to include this effect in future belief models and to study the resulting effect on the convergence of the system. We also plan to investigate richer models of the relationship between agents and the resulting effects on belief propagation, including authority relationships, trust, and reputation. Finally, we plan to investigate the effects of malicious manipulation by an intelligent adversary on the convergence of the system.

References

1. Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality: an explanation of 1/f noise. Phys. Rev. Lett. **59**, 381–384 (1983)
2. Bellur, B., Lewis, M., Templin, F.: An ad-hoc network for teams of autonomous vehicles. In: Proceedings of the First Annual Symposium on Autonomous Intelligence Networks and Systems (2002)
3. Clar, S., Drossel, B., Schwabl, F.: Scaling laws and simulation results for the self-organized critical forest-fire model. Phys. Rev. E **50**, 1009–1018 (1994)
4. Malte-Sorensen, A., Feder, J., Christensen, K., Frette, V., Josang, T., Meakin, P.: Surface fluctuations and correlations in a pile of rice. Phys. Rev. Lett. **83**, 764–767 (1999)
5. Motter, A.E., Lai, Y.-C.: Cascade-based attacks on complex networks. Phys. Rev. E **66**, 065102 (2002)
6. Newman, M.E.J.: The structure and function of complex networks. SIAM Rev. **45**, 167 (2003)
7. Olami, Z., Feder, H., Christensen, K.: Self-organized criticality in a continuous, nonconservative cellular automaton modeling earthquakes. Phys. Rev. Lett. **68**, 1244–1247 (1992)
8. Sukthankar, G., Sycara, K., Giampapa, J., Burnett, C.: A model of human teamwork for agent-assisted search operations. In: NATO Research and Technology Organisation: Human Factors and Medicine Panel Symposium (NATO HFM-142/RTO Symposium). Neuilly-sur-Seine Cedex, April 2008. NATO RTA/HFM
9. Watts, D.J.: A simple model of global cascades on random networks. Proc. Nat. Acad. Sci. **99**, 5766–5771 (2002)
10. Zhang, Y.: Scaling theory of self-organized criticality. Phys. Rev. Lett. **63**(5), 470–473 (1989)

Chapter 9
Effect of Humans on Belief Propagation in Large Heterogeneous Teams

Praveen Paruchuri, Robin Glinton, Katia Sycara, and Paul Scerri

Summary Members of large, heterogeneous teams often need to interact with different kinds of teammates to accomplish their tasks, teammates with dramatically different capabilities to their own. While the role of humans in teams has progressively decreased with the deployment of increasingly intelligent systems, they still have a major role to play. In this chapter, we focus on the role of humans in large, heterogeneous teams that are faced with situations, where there is a large volume of incoming, conflicting data about some important fact. We use an abstract model of both humans and agents to investigate the dynamics and emergent behaviors of large teams trying to decide whether some fact is true. In particular, we focus on the role of humans in handling noisy information and their role in convergence of beliefs in large heterogeneous teams. Our simulation results show that systems involving humans exhibit an enabler-impeder effect, where if humans are present in low percentages, they aid in propagating information; however when the percentage of humans increase beyond a certain threshold, they seem to impede the information propagation.

9.1 Introduction

As intelligent systems get increasingly deployed in the real world, humans need to work in cooperation with various other entities such as agents and robots. The effective sharing and use of uncertain information in decision-making is the key

P. Paruchuri (✉) · R. Glinton · K. Sycara · P. Scerri
Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: paruchur@cs.cmu.edu

R. Glinton
e-mail: rglinton@cs.cmu.edu

K. Sycara
e-mail: katia@cs.cmu.edu

P. Scerri
e-mail: pscerri@cs.cmu.edu

to the success of such large heterogeneous teams in complex environments. Typically, noisy information is collected by some portion of the team and shared with others via a network [3]. Each team member will then use this uncertain information and the beliefs of those around them to develop their own beliefs about relevant facts. For example, consider the case of Network Centric Warfare [6, 13]. In this domain, a large number of agents and humans must do a variety of tasks, such as planning, information fusion and resource allocation. All these tasks need the exchange of potentially uncertain information between the different team members to arrive at a decision required to achieve joint goals. In this chapter, we focus on the role of humans in handling noisy information and their role in convergence of beliefs in large heterogeneous teams that include machine agents.

We developed an abstracted model and simulator for studying the dynamics and interaction between the various members of large, heterogeneous teams. In our model, the team is connected via a network with some team members having direct access to sensors, while others relying solely on neighbors in the network to provide inputs for their beliefs. Each agent then uses Bayesian reasoning over the beliefs of direct neighbors and sensor data to maintain it's own belief about a single fact which can be *true*, *false*, or *unknown*. Simulation results indicate that the number of agents coming to a correct conclusion about a fact and the speed of their convergence to this belief depends on various factors including network structure, the percentage of humans in the team and the conditional probability of belief in neighbor's information.

Our experiments show that our belief propagation model exhibits the property that a single additional piece of data can cause a cascade of belief changes, referred to as *avalanches*. In particular, we observed that the frequency distribution of these avalanches follows a *power law* over a certain conditional probability range, a key feature of *SOC*'s (Self-organizing critical systems). Our results also indicate that humans have a significant effect on the way these avalanches occur. In particular, we uncovered a new effect due to the presence of humans which we call the enabler-impeder effect which holds across a general range of experimental settings.

The rest of this chapter is organized as follows. Section 9.2 provides a brief description of the Self Organized Criticality property. Section 9.3 describes the *enabler-impeder* effect, a new effect that we uncovered in this work. Section 9.4 provides the actual model of a heterogeneous team that we use for purposes of simulation. Section 9.5 provides the results of the various experiments we performed using our simulator. In particular, we present extensive simulation results that shows the influence of humans under different system parameters on the behavior of large scale heterogeneous teams. Our results uncover the enabler-impeder effect due to the presence of humans and the SOC nature of such teams. Section 9.6 provides the related work while Sect. 9.7 summarizes the chapter and provides a list of possible extensions for this work.

9.2 Self-Organized Critical Systems

There has been a lot of focus on identifying systems that exhibit Self Organized Criticality (SOC) as a property [1, 5, 7, 8, 12]. The main reason that SOC behavior is considered important is that most of the observed phenomenon in SOC systems occur for a large range of parameters [1], as opposed to the dependency on fine-tuning of parameters in general complex systems. Systems which exhibit SOC share the following fundamental characteristics. All SOC systems consist of a large number of interacting constituents which interact with a relatively small percentage of the other system elements. The behavior of these constituents are dominated by three factors: (a) An external drive that changes the state of the individual; (b) Resistance of the individual to change; (c) Threshold in the local resistance at which the individual changes its belief or behavior. In our experimental section, we show that the system modeled here indeed exhibits these features over a large parameter range. A key feature of SOC's is that the avalanches caused by a single additional piece of data exhibits a power law property over a certain conditional probability range. The power law implies that the distribution of avalanche sizes is dominated by many small avalanches and exponentially fewer large ones. Identifying this behavior is important since the critical behavior of SOC's is not dependent on finely tuned parameters, and hence we can expect this criticality to occur often in real-world systems too. The power law suggests that large avalanches are relatively infrequent. However, when they do occur, when sparked by incorrect data, the result can be that the entire team reaches a wrong conclusion despite exposure to primarily correct data. In many domains, such as our Network Centric Warfare, this is an unacceptable outcome even if it does not occur often.

9.3 The Enabler-Impeder Effect

Addition of humans to large heterogeneous teams result in changing the interaction dynamics of the existing system. In particular, we uncovered a new team dynamic which we refer to as the enabler-impeder effect. This effect means that up to a certain percentage, humans enable belief propagation and above that percentage they actually impede the propagation. Thus, big avalanches occur most often in teams with some humans, but less often in teams with few or many humans. The reason behind the enabler-impeder effect is as follows: The enabler effect is due to the fact that agents have a higher belief in humans while the impeder effect is due to the fact that humans have a high latency between successive belief calculations. More details of our model of humans is presented in Sect. 9.4.

We now provide a brief discussion on the effect of the enabler-impeder effect on the formation of avalanches. In general, occurrence of avalanches is good as it implies that agents are able to convey their beliefs to many other agents in the network. Bigger avalanches imply that each agent is able to convey and influence the beliefs of a larger set of agents. However, really large avalanches can potentially be harmful since agents can start imposing their beliefs on others or manipulate the network.

We mention the word potentially here because there can be situations where we may want one agent's opinion to be instantaneously propagated across the network. Such situations can include an emergency where we want an instantaneous propagation of beliefs or it can be that the new input is the truth, and hence propagating it instantaneously is the correct thing to do. Considering the fact that avalanches are good up to moderate sizes and potentially harmful thereafter, the enabler-impeder effect of humans can be classified as a positive effect on the team. This is because while a moderately high percentage of humans in a team helps in forming avalanches, addition of humans thereafter impedes the formation of big avalanches that are potentially harmful.

9.4 Model of Information Dissemination in a Network

We now introduce the network model similar to the one presented in [7]. In this model, there is a team of agents, $A = \{a_1, \ldots, a_N\}$, where an agent is a generic term to imply humans, robots, or agents. We are interested in large teams, $|A| \geq 1000$, that are connected to each other via a network, K. K is represented by $|A| \times |A|$ matrix where $K_{i,j} = 1$, if i and j are neighbors and 0, otherwise. The network is assumed to be relatively sparse, with $\sum_j K_{i,j} \approx 4, \forall i$. There are six different network structures we consider in this chapter: (a) Scalefree, (b) Grid, (c) Random, (d) Smallworld, (e) Hierarchy and (f) HierarchySLO (a hierarchy network with sensors only at the leaf nodes). A comprehensive definition of these networks can be found in [11].

Some members of the team, $H \subset A$ are considered to be humans. Certain members of the team, $S \subset A$ with $|S| \ll |A|$, have direct access to a sensor. A sensor simply returns a value of *true* or *false* with a particular probability. Sensor readings are noisy, so the sensor could return incorrect values with some probability. The frequency with which a sensor returns the correct value is modeled as a random variable R_s which is normally distributed with a mean μ_s and a variance σ_s^2. Agents that are directly connected to sensors incorporate new sensor readings according to the following belief update equation.

$$P'(b_{a_i} = true) = \frac{A}{B + C}$$
$$A = P(b_{a_i} = true) P(s_{a_i} = false / F = true)$$
$$B = (1.0 - P(b_{a_i} = true)_t P(s_{a_i} = false / F = false)$$
$$C = P(b_{a_i} = true) P(s_{a_i} = false / F = true)$$

Agents use the following equation to incorporate the beliefs of neighbors:

$$P'(b_{a_i} = true) = \frac{D}{E + G}$$
$$D = P(b_{a_i} = true) P(b_{a_j} = false / F = true)$$

$$E = (1.0 - P(b_{a_i} = true)P(b_{a_j} = false/F = false)$$
$$G = P(b_{a_i} = true)P(b_{a_j} = false/F = true)$$

where, $P(b_{a_i})$ gives the prior belief of agent a_i in the fact F and $P(s_{a_i}/F)$ gives the probability that the sensor will return an estimate of the fact (*true* or *false*) given the actual truth value of the fact. We refer to, $P(b_{a_j}/F)$, interchangeably as the *belief probability* or the *conditional probability* or *CP*. $P(b_{a_j}/F)$ gives a measure of the credibility that an agent a_i assigns to an estimate of the value of F received from a neighbor a_j given F.

Each agent decides that F is either *true*,*false*, or *unknown* by processing its belief using the following rule. If T is a threshold probability and ϵ an uncertainty interval, the agent decides that the fact is *true* if $P(b_{a_i}) > (T + \epsilon)$, *false* if $P(b_{a_i}) < (T - \epsilon)$, and *unknown* otherwise. Once the decision is made, if the agents decision about F has changed, the agent reports this change to its neighbors. Note that in our model, neither $P(b_{a_i})$, the probability that F is *true* according to agent a_i, or the evidence used to calculate it, is transmitted to neighbors. Instead, the only parameter that is transmitted is the actual value of F when there is a change in belief (i.e., either *true*, *false* or *unknown*).

We use a distinct model for the humans in our simulator. In particular, we make the following two assumptions with respect to humans: (a) Humans are given much larger credibility than other agents. In other words, agents have a higher belief in human's estimation of the value of F than in the value of F given to them by other agents. That is, for $a_h \in H$ and $a_k \notin H$, $P(b_{a_h}/F) \gg P(b_{a_k}/F)\ \forall_{h,k}$. This is to abstract away the fact that humans are good at analyzing (given particular information) and hence agents believe in humans more. (b) Humans have a higher latency between successive belief calculations. Effectively, this means that humans are slower to sense the environment or update their beliefs. This is to account for the fact that humans do not just believe others at face value, but change their beliefs using more complicated reasoning models without getting into the detailed modeling issues. We believe that these two assumptions help in achieving our goal of building more realistic human models.

9.5 Simulation Results

We performed a series of seven experiments to understand the effects of humans on the key properties and predictions of the system. All our experiments were performed with networks of 1000 agents with six different network topologies [11]. These six different topologies as mentioned earlier are: (1) Scalefree, (2) Grid, (3) Random, (4) Smallworld, (5) Hierarchy and (6) HierarchySLO. Before presenting the details of our experiments, we introduce a metric named center of mass which is useful to concisely capture the dynamics of belief cascades. Center of mass is defined as:

$$\frac{\sum_{i=1\ldots N}(AvalancheFrequency_i * AvalancheSize_i)}{\sum_{i=1\ldots N} AvalancheFrequency_i}$$

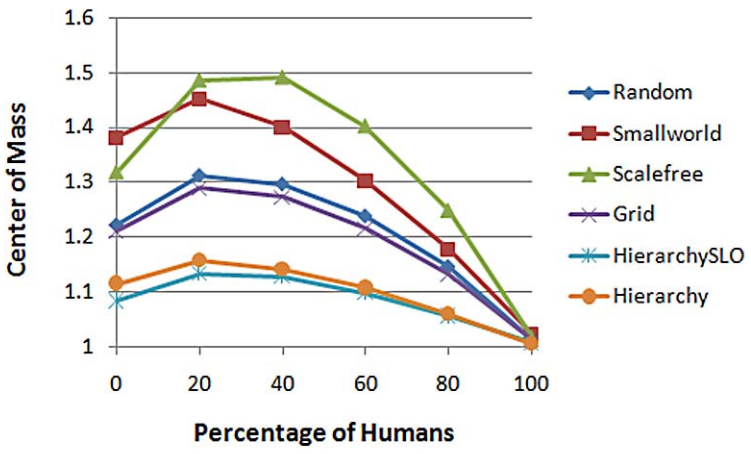


Fig. 9.1 Enabler–Impeder effect of humans

where $AvalancheSize_i = i$, $AvalancheFrequency_i$ is the frequency of avalanche size i and N is the maximum possible avalanche size (which is 1000 in our experiments since there are 1000 agents in our simulation).

Our first experiment studies the effect that humans have on belief propagation within the system. Recall that our modeling assumption about humans is that: (a) They have higher credibility, i.e., an agent believes a human with higher probability than it believes another agent and (b) Humans are slower to make decisions i.e., they have a higher latency between successive belief calculations. In our first experiment, we varied the percentage of humans in the team from 0 to 100 in increments of 20 across the six different network topologies and measured the center of mass of the avalanches obtained. The x-axis of Fig. 9.1 shows the percentage of humans in the network increasing from 0 to 100% and the y-axis represents the center of mass. The figure shows the enabler–impeder effect due to the presence of humans in the team. In particular, we observe that the center of mass increases till the percentage of humans in the team reaches 20%, the enabler effect, and then decreases thereafter, the impeder effect. This effect was noted across all the network types. The higher credibility of humans tends to encourage a change of belief in others who receive human input and thus humans can encourage or enable the formation of avalanches. Conversely, the long latency of humans in changing their beliefs has the effect of impeding the avalanche formation. When the percentage of humans is below 20% the avalanche enhancing effect of the high human *CP* predominates, while above this value the impedance effect caused by the long latency predominates.

In our second experiment, we show that the network convergence rate is a function of percentage of humans in the team for a Scalefree network. Figure 9.2 shows the varying percentages of humans on the x-axis and the frequency of convergence to the ground truth on the y-axis. The total number of runs for this experiment is 5000 and the ground truth is *True*. Each line in the graph corresponds to a particular percentage or greater number of agents converging to the ground truth. Note that

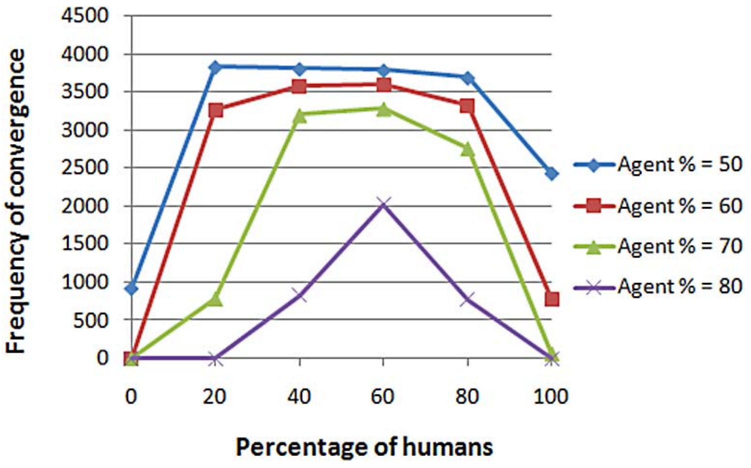


Fig. 9.2 Network convergence rate as a function of human percentage

greater than 90% of agents, never converged to the ground truth and hence the graph shows only four lines corresponding to greater than or equal to 50, 60, 70, and 80 percent. An example data that can be obtained from this graph is: If the percentage of humans in the team is 20, then at least 50% of the agents have converged to the ground truth 3833 times out of the total 5000 runs. When there are no humans in the team more than 60% of the agents never converged to the ground truth.

Figure 9.2 confirms our observation that there is an enabler–impeder effect. In particular, addition of humans to the team helped in formation of avalanches, thus implying that humans play a key role in propagating the ground truth. This observation is true here since more than 50% of the team converges to the ground truth in almost all the settings of the graph. However, once the percentage of humans crosses 60, the frequency of convergence to the ground truth decreases since humans impede the propagation of information and hence the propagation of the ground truth.

Our third experiment shows that our system exhibits SOC behavior with a varying *CP* and percentage of humans in the team. In particular, we performed the experiment varying the *CP* values from 0.52 to 0.58 in increments of 0.02 and the human percentage from 0 to 100 in increments of 20. The results of this experiment are shown in Fig. 9.3. The five plots in the figure correspond to varying the human percentage in the system. The x-axis in all the plots denotes the size of the avalanches and the y-axis denotes the frequency with which avalanches of various sizes occur. Both the axes are plotted on a log–log scale, and hence we expect a power law to be evident as a straight line. The underlying network used to produce this figure was a scale-free network. The 4 lines in the figure correspond to *CP* values of 0.52, 0.54, 0.56, and 0.58. Each point in the graph is an average over 5000 runs. In general, *CP* values greater than or equal to 0.6 did not exhibit the power law. This is because if agents give their neighbor's estimates high credibility (large values of *CP*), all the agents change their beliefs immediately upon receiving input from a neighbor. Hence, the bigger avalanches occur frequently without having a distribution over

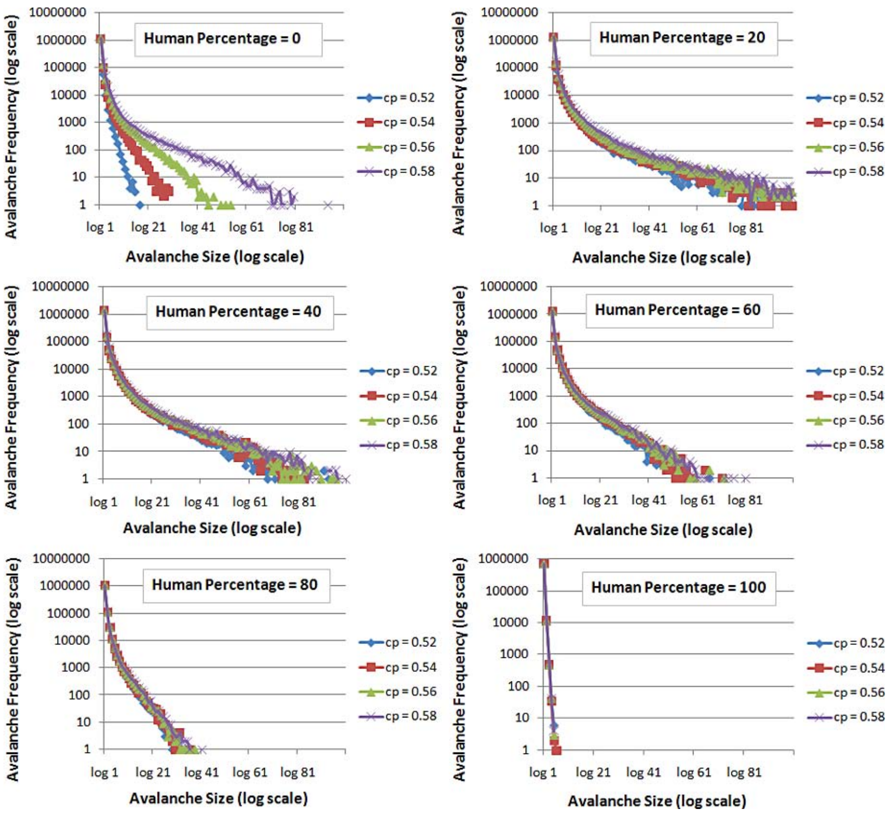


Fig. 9.3 Avalanche size versus frequency across various percentages of humans

the small and medium size avalanches which led to the power law. While we show this figure solely for a scalefree network, our experiments show that this trend is true across all the network topologies we examined.

From Fig. 9.3, we notice that most of the plots follow the power law in the *CP* range from 0.52 to 0.58. However, the addition of humans has a significant effect. When the percentage of humans was zero, the plot was more spread out i.e., *CP* had a greater effect and hence there is a greater spread between the lines even with small *CP* increments. However, the effect of *CP* decreases even with the addition of a small percentage of humans in the team i.e., there is not much spread between the various *CP* lines when the percentage of humans in the team is greater than zero. The explanation for this is as follows. When there are no humans in the team, the main parameter affecting the avalanche sizes was the belief in neighbors. Hence, when *CP* increases there is a great variation in the occurrence of various avalanche sizes. However as humans get added to the team (even by a small percentage), given that agents have a higher belief in humans, they have a much greater effect on the dynamics of avalanche formation. Thus, increasing the *CP* in teams with humans

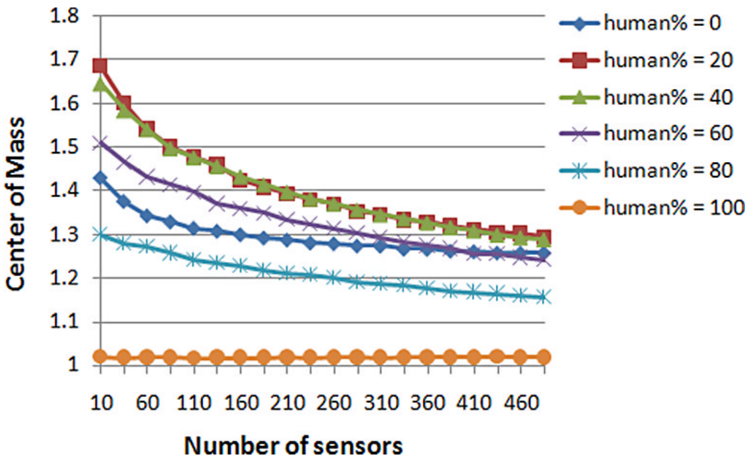


Fig. 9.4 Number of sensors versus Center of Mass

has a small effect on the dynamics of avalanches formed as opposed to the large effects seen when there are no humans.

Our fourth experiment looks at the relationship between the number of sensors and the center of mass for a Scalefree network as the percentage of humans in the team is varied. A similar trend holds for the other network topologies considered in this chapter. Figure 9.4 shows the number of sensors varying from 10 to 485 on the x-axis and the center of mass on the y-axis. We assume that sensors return observations randomly on an average of every second step. The six lines in the graph correspond to human percentages of 0, 20, 40, 60, 80, and 100. Each point in the graph is generated using 1000 agents and averaged over 1000 runs. The graph shows that as the number of sensors increase the center of mass of avalanches decreases (or increases by a negligible amount), across the various human percentages. This is because agents with sensors have access to more information and are thus more certain in their own beliefs. Hence, they are less likely to change their beliefs in response to information from neighbors and hence the cascades of belief changes are likely to be smaller. This effect of low probability of beliefs in neighbors can be a problem if some sensors in the system give out particularly bad data since such agents are likely to ignore potentially correct information from their neighbors.

Another interesting point to note from this graph is the enabler–impeder effect described in our earlier experiment. In Fig. 9.4, let us consider the center of mass values when the number of sensors is fixed to 210. The center of mass values at this point are $\langle 1.29, 1.39, 1.40, 1.33, 1.21, 1.02 \rangle$ corresponding to the following percentages of humans: $\langle 0, 20, 40, 60, 80, 100 \rangle$. The enabler effect is predominantly visible here since the center of mass increases until the percentage of humans reaches 40. However, above 40% the center of mass values decrease. This shows that varying the human percentage has a strong effect on the outcome even if the other parameters of the experiment change significantly. A similar pattern is observed in all our experiments presented below even though a variety of parameters are changed, thus

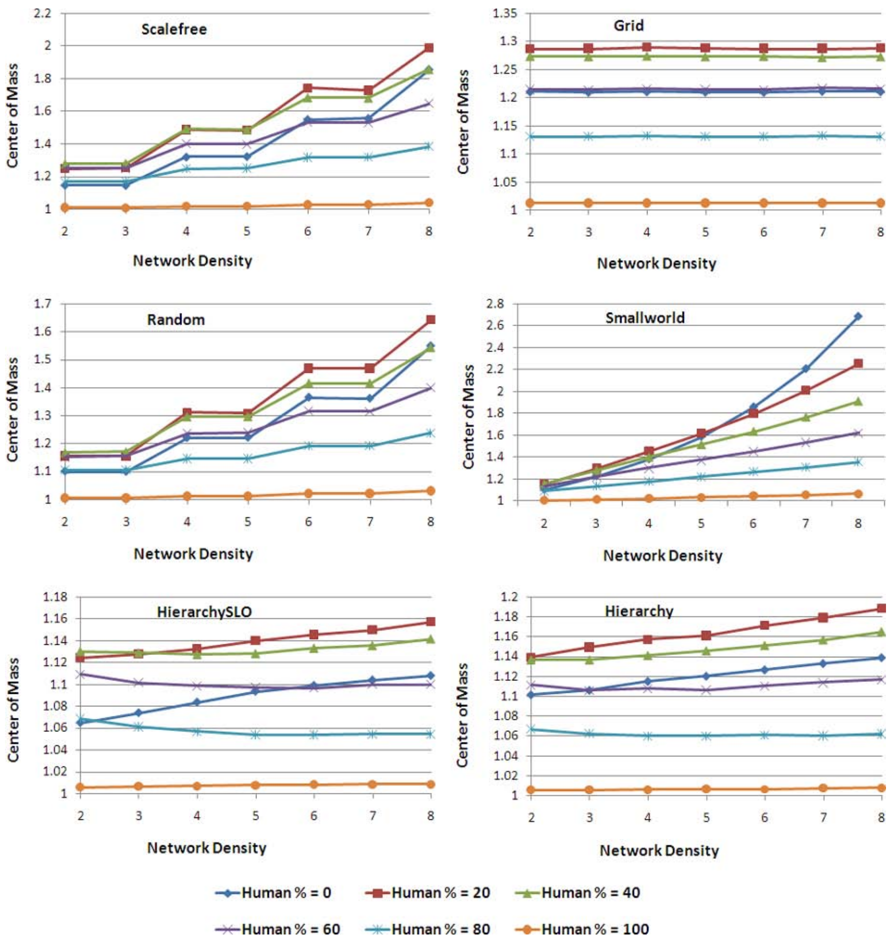


Fig. 9.5 Density versus Center of Mass

showing the generality of this effect. To avoid repetition, we indicate the occurance of this effect in the rest of the experiments without repeating the description.

Our fifth experiment looks at the relationship between the network density and the center of mass values across six different network topologies as the percentage of humans in the team is varied. Figure 9.5 shows six plots each corresponding to a different network type. In each plot, the network density is plotted on the x-axis and the center of mass on the y-axis. The six lines in each plot correspond to the six different percentages of humans in the team. Each point in the graph is generated using 1000 agents averaged over 1000 runs. The plots show that as the network density increases the center of mass in general increases except for the grid network where the center of mass mostly remains constant. This is due to the fact that each agent can influence more neighbors and hence lead to the formation of bigger avalanches resulting in a higher or at least equal center of mass. As mentioned

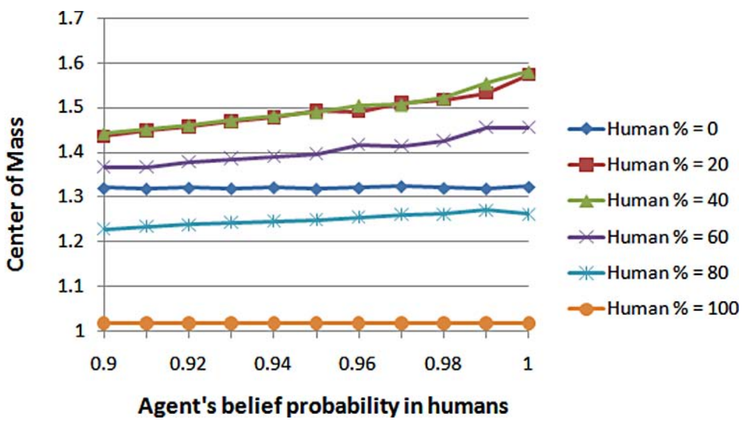


Fig. 9.6 Effect of increase in agent's belief in humans

earlier, the enabler–impeder effect of humans remains a predominant effect in this experiment too across all the networks except the smallworld network.

Our next experiment looks at the effect of the increase in agent's belief in humans over the center of mass as the percentage of humans in the team is varied. Figure 9.6 shows the agent's belief in humans on the x-axis and the center of mass on the y-axis. The six lines in the figure correspond to the human percentage in the team varied from 0 to 100. From the plot we can see that, as the agent's belief in humans increase, the center of mass in general increases across all percentages of humans in the team except when the percentage is 0 or 100. When the percentage of humans is 0, since there are no humans in the team the parameter belief probability in humans has no effect as expected. Similarly, when the percentage of humans is 100, i.e., when there are no agents in the team, the size of the center of mass remains constant since the parameter cannot have any effect. Similar to all our earlier plots, the enabler–impeder effect is observed in this plot too.

Our last experiment looks at the effect of increase in human's belief in other humans on the center of mass as the percentage of humans in the team is varied. Figure 9.7 shows the belief probability of humans in humans varying from 0 to 1 on the x-axis and the center of mass on the y-axis. The six lines in the figure correspond to the percentage of humans in the team varying from 0 to 100. When there are no humans in the team the belief probability parameter does not affect the simulation and hence a straight line is obtained as expected. For the rest of the lines, when the belief probability is zero it means that humans do not believe other humans and hence disregard other human's inputs. As the belief probability increases, humans start considering others beliefs, but still remain skeptical. This trend continues till 0.5 where the situation becomes equivalent to information coming from an unbiased coin toss. Due to this skeptical nature of humans in other humans, they effectively impede the formation of avalanches and hence lower the center of mass. As the belief probability in other humans crosses 0.5 and moves towards 1, humans start believing other humans more, enabling the formation of bigger avalanches and

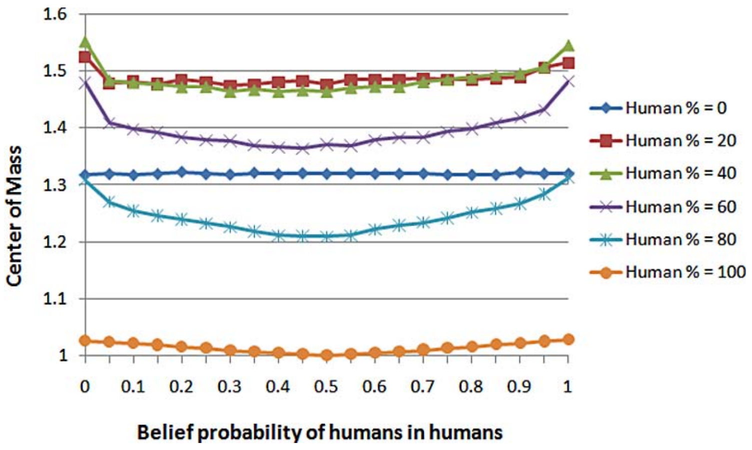


Fig. 9.7 Effect of increase in agent's belief in humans

hence increase the center of mass. As obtained in all our earlier plots the enabler–impeder effect is observable across the various human percentages in the team.

9.6 Related Work

Many studies have focused on the belief propagation models for large heterogeneous teams. While most of these studies focus on the cascade effects of belief propagation, not much effort has been put into studying the specific effect of humans on such teams. For example, in [14] a belief propagation model is used for a large heterogeneous team. Using this model, the paper shows that the cascades generated by their system follows a power law and hence exhibits SOC dynamics. Our model is similar to theirs, in the sense that they too use a heterogeneous nodes model with state change based on a distribution of thresholds. However, they do not introduce an explicit model of the human. The main difference introduced due to the presence of humans is that humans have a higher credibility and increased latency for changing as compared to other agents. We then explored the effects of these changes and uncovered two key properties: (a) The presence of humans results in the enabler–impeder effect, a new effect that has not been uncovered earlier. (b) The belief dynamics of our system across various human percentages exhibits the properties of a Self-Organizing Critical system. In [9], Motter discusses cascading failures in electrical networks. In particular, Motter investigates cascades due to removal of nodes as opposed to cascades of decisions as in our work. Furthermore, there is no explicit model of humans considered in their work.

In contrast, [2, 4], and [10] study the properties of heterogeneous human teams. While each of these papers presents a different way in which the team members update their beliefs, all these works uncover the cascading effects in large scale teams with humans. For example, [10] studies the cascading effects in heterogeneous teams where the individuals are modeled as naive Bayesian norm followers. Using this model, they uncovered the herding behavior in human societies, a direct result of the cascading effect of belief propagation. However, none of these works uncover the enabler–impeder effect that humans bring in nor the SOC nature of such teams.

9.7 Conclusions and Future Work

Our conclusions from this work are the following. We built an abstract model and a simulator that studies the dynamics and interactions between the various members of a heterogeneous team. We then performed various experiments to study the properties of the simulator. In particular, we focused on the role of humans in all our experiments. Our main results are as follows: (a) Humans can have a dramatic effect on information propagation which we characterized as the enabler–impeder effect. In particular, the effect means that small percentages of humans encourage information propagation in the system while large percentages inhibit the information propagation. (b) We also found that the enabler–impeder effect is consistent even if other parameters of the domain change significantly. (c) We demonstrated that our system of belief sharing agents with varying percentages of humans in the team exhibits SOC dynamics for certain parameter ranges, thus providing a basis for why our results might hold in a real situation.

In the future, we plan to build more realistic models of humans. In particular, we would like to model the fact that humans may not maintain exact numbers for beliefs but a more complicated model. This complicated model could be maintaining beliefs as intervals or as probability distributions. We also plan to attribute psychological factors for humans such as optimism, pessimism etc. and other personal attributes such as motivations and emotions. Effectively, such a modeling can bring out the notion that humans are decision makers rather than just accumulators of belief. We would also like to model the fact that humans do not just communicate belief probabilities but also the reasoning behind them. This would require us to develop richer communication models.

Acknowledgement This research was funded by AFOSR grants FA95500810356 and FA95500710039.

References

1. Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality. Phys. Rev. A **38**(1), 364–374 (1988)
2. Banerjee, A.: A simple model of herd behavior. Q. J. Econ. **107**(3), 797–817 (1992)
3. Bellur, B., Lewis, M., Templin, F.: An ad-hoc network for teams of autonomous vehicles. In: Proc. of First Annual Symposium on Autonomous Intelligence Networks and Systems (2002)

4. Bikhchandani, S., Hirshleifer, D., Welch, I.: A theory of fads, fashion, custom, and cultural change as informational cascades. J. Polit. Econ. **100**(5), 992–1026 (1992)
5. Clar, S., Drossel, B., Schwabl, F.: Scaling laws and simulation results for the self-organized critical forest-fire model. Phys. Rev. E **50**(2), 1009–1018 (1994)
6. Dekker, A.H.: Centralisation and decentralisation in network centric warfare. J. Battlef. Technol. **6**, 23–28 (2003)
7. Glinton, R., Paruchuri, P., Scerri, P., Sycara, K.: Self organized criticality of belief propagation in large heterogeneous teams. In: Hirsch, M.J., Pardalos, P., Murphey, R. (eds.) Dynamics of Information Systems: Theory and Applications. Springer Optimization and its Applications. Springer, Berlin (2010)
8. Malthe-Sørenssen, A., Feder, J., Christensen, K., Frette, V., Jøssang, T.: Surface fluctuations and correlations in a pile of rice. Phys. Rev. Lett. **83**(4), 764–767 (1999)
9. Motter, A., Lai, Y.: Cascade-based attacks on complex networks. Phys. Rev. E **66**(6), 065102 (2002)
10. Neill, D.: Cascade effects in heterogeneous populations. Ration. Soc. **17**(2), 191–241 (2005)
11. Newman, M.E.J.: The structure and function of complex networks. SIAM Rev. **45**, 167–256 (2003)
12. Olami, Z., Feder, H.J.S., Christensen, K.: Self-organized criticality in a continuous, nonconservative cellular automaton modeling earthquakes. Phys. Rev. Lett. **68**(8), 1244–1247 (1992)
13. Sukthankar, G., Sycara, K., Giampapa, J.A., Burnett, M.C.: A model of human teamwork for agent-assisted search operations. In: Proceedings of the NATO Human Factors & Medicine Panel Symposium on Adaptability in Coalition Teamwork. NATO RTA/HFM (2008)
14. Watts, D.: A simple model of global cascades on random networks. Proc. Nat. Acad. Sci. USA **99**(9), 5766–5771 (2002)

Chapter 10
Integration of Signals in Complex Biophysical Systems

**Alla Kammerdiner, Nikita Boyko, Nong Ye,
Jiping He, and Panos Pardalos**

Summary There is clear evidence of fusion processes exhibited by biophysical systems, such as the brain. One simple example is the way a human brain processes visual information. In fact, one of the consequences of normal integration of the visual information from two retinas in the visual cortex is ability for depth perception. In actuality, a primate brain is capable of integrating visual, auditory, cutaneous, and proprioceptive signals in order to extract crucial information that may not otherwise be fully present in any single type of signal.

Our analysis of neural data collected from primates during sensory-motor experiments shows a clear presence of transient fusion of neural signals. In particular, the activity in the brain regions responsible for motor planning and control exhibit cointegration among the instantaneous phase measures, which is associated with generalized phase synchronization of neural activity.

A. Kammerdiner (✉) · N. Ye
Department of Industrial Engineering and Operations Research, Arizona State University, Tempe, AZ 85287-5906, USA
e-mail: akammerd@asu.edu

N. Ye
e-mail: nongye@asu.edu

N. Boyko · P. Pardalos
Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611-6595, USA

N. Boyko
e-mail: nikita.boyko@gmail.com

P. Pardalos
e-mail: pardalos@ufl.edu

J. He
Department of Biomedical Engineering, Arizona State University, Tempe, AZ 85287-9709, USA
e-mail: jiping.he@asu.edu

10.1 Introduction

When viewed as a self-organized cooperative biological information system, the brain can be used as a model for developing biologically inspired approaches to intelligent information fusion. The brain of primates is an incredibly complex biophysical system that consists of an extremely large number of interacting neurons (e.g., estimated 50–100 billion neurons in a human brain) grouped into functionally diverse areas and is capable of simultaneous processing and continuous integration of large amounts of multi-modal information.

A process of transmission, fusion, aggregation and distribution of information in the brain is reflected in spatio-temporal patterns of excitation spread over a large number of neurons. The brain handles large volumes of sensory information in parallel via coordinated dynamical interaction of large number of neurons that are distributed within and across different specialized brain regions. Amazingly, such complex neuronal interactions allow for very high, efficient and flexible information processing. The brain's ability of creating an internal model of the environment through learning and self-organization enables effective behavior in response to external changes.

The visual information processing in the human brain provides an example of fusion processes exhibited by biophysical systems. In normal vision, the inputs from both eyes are fused by sensory systems in the brain into a single percept. In particular, this neural process, known as binocular integration, is responsible for our depth perception ability. Because of the distributed organization of sensory systems, the representation of real-world objects calls for integration of neural activity across various cortical areas [22]. Since many objects are multisensory in their nature and, as such, may possess a combination of visual, auditory, haptic and olfactory characteristics, this integration must be supported by the apparatus for fusion of signals across different modalities. In fact, as indicated in [22], at all levels of sensory processing, the neuronal activity is modulated by top-down attentional mechanisms that allow to dynamically select and fuse sensory signals in a context specific way [3]. Additionally, flexible dynamic binding of neural activity in sensory and motor cortical regions is necessary for the sensory-motor coordination [18]. Short-term synchronization of neuronal discharges has been long attributed as a mechanism that facilitates dynamical integration of widely distributed collections of neurons into functionally coherent groups that guides execution of cognitive and motor tasks [6, 19]. Moreover, neural synchronization appears to be involved in large-scale integration of distributed neural activity, which occurs across distant cortical regions [18].

A number of studies have found clear evidence that neural synchronization plays an important role in a variety of cognitive functions, such as sensory-motor integration [9, 24]. Specifically, large-scale frequency-specific synchronization of neural activity has been linked with information integration in associative learning [14], meaningful perception [17], perceptual awareness [13, 21], and internal cortical interaction and top-down information processing [25]. Interestingly, a recent study [9] discovered a close relationship between the noise-induced changes in behavioral performance and the respective changes in phase synchronization between widely separated brain areas. The authors conclude that these findings imply that noise-induced large-scale neural synchronization may play a significant role in the transmission of information in the brain.

Furthermore, importance of neural synchronization processes for normal cognitive functioning is highlighted by the studies of synchronization in the brain activity of the patients affected by several neurological disorders [22]. In fact, abnormal neural synchrony patterns are present in certain brain disorders, such as epilepsy, autism, Alzheimer's disease, schizophrenia, and Parkinson's Disease.

10.2 Methods for Analysis of Phase Synchronization

Synchronization can be loosely defined as a process of active adjustment in the rhythms of the oscillating systems (or subsystems) due to some interaction that can be characterized by an emergence of the stable relationship between the rhythms of the systems. The synchronized systems, i.e., the systems in which such stable relationship between their respective rhythms has been achieved, are often said to exhibit *phase-locking*.

Since the first studies describing presence of synchronization in biophysical systems, such as the heart [23] or the brain, a number of different types of synchronization have been introduced, including identical synchronization [4], generalized synchronization [1], phase synchronization [15] and, most recently, generalized phase synchronization [8]. Here, we will focus our attention on the concepts of phase synchronization and generalized phase synchronization.

In the last decade, phase synchronization has been used extensively to investigate large-scale integration between neural signals [24]. Most of the approaches for studying phase synchronization utilize instantaneous phase in some way or another.

10.2.1 Instantaneous Phase

A general approach for measuring instantaneous phase has originally been developed by Gabor [5] and is based on the notion of analytical signal. For an arbitrary continuous real-valued function $x(t)$ of time parameter t, its *analytical signal* $\xi_x(t)$ is a complex-valued function of t defined as

$$\xi_x(t) = x(t) + i\,\tilde{x}(t) \tag{10.1}$$

where i is an imaginary unit and $\tilde{x}(t)$ is the Hilbert transform of $x(t)$, which is calculated using the following formula:

$$\tilde{x}(t) = C.P.V.\int_{-\infty}^{+\infty} \frac{1}{\pi}\frac{x(s)}{t-s}ds \tag{10.2}$$

where the abbreviation $C.P.V.$ symbolizes that the integral is taken in the sense of Cauchy principal value [2].

Since $\xi_x(t)$ is complex-valued, it can alternatively be written via its polar form as follows:

$$\xi_x(t) = r_x(t)\exp\{i\ \phi_x(t)\} \tag{10.3}$$

where $r_x(t)$ and $\phi_x(t)$ denote the instantaneous amplitude and instantaneous phase of the function $x(t)$, respectively. By combining formulas (10.1) and (10.3), it is easy to see that the instantaneous phase can be directly computed from $x(t)$ as

$$\phi_x(t) = \arctan\left\{\frac{\tilde{x}(t)}{x(t)}\right\} \tag{10.4}$$

An alternative approach for calculating the instantaneous phase of a given function, which was proposed by Lachaux et al. [10], is based on the wavelet transform and is simply analogous to the Hilbert transform approach, which was presented earlier. In the wavelet transform method, the instantaneous phase is extracted from $x(t)$ by means of convolution of $x(t)$ with a complex Morlet wavelet (also called Gabor function):

$$\varphi(t, f) = \exp\left\{-\frac{t^2}{2\sigma^2} + i\ 2\pi f t\right\} \tag{10.5}$$

where f is a frequency parameter at which $\varphi(t, f)$ is centered at, and σ is a chosen rate of decay. Since the result of the convolution of $x(t)$ with the complex wavelet $\varphi(t, f)$ for a fixed value f_0 of frequency parameter is also a complex-valued function of t, like $\xi_x(t)$ in the Hilbert transform approach, then the instantaneous phase can be obtained analogously to (10.4) as:

$$\phi_x(t) = \arctan\left\{\frac{\Re(W_x(t))}{\Im(W_x(t))}\right\} \tag{10.6}$$

where $\Re$ and $\Im$ denote real and imaginary parts, respectively of the convolution of $x(t)$ with the wavelet $\varphi(t, f_0)$:

$$W_x(t) = \int_{-\infty}^{+\infty} \varphi(s, f_0)\, x(t-s)ds \tag{10.7}$$

Not only are these two approaches for extracting the instantaneous phase analogous, but also the instantaneous phases obtained by the Hilbert and the wavelet transform methods are shown to be closely connected, which was first demonstrated experimentally in [11] and later explained theoretically in [16]. Loosely speaking, the phase determined using the wavelet transform approximately corresponds to the phase computed via the Hilbert transform applied to the band-pass filtered data. Basically, the wavelet transform limits the extracted information to the main frequency band centered at a given frequency f_0 with the rate of decay σ, whereas the Hilbert transform is parameter free and therefore preserves the entire power spectrum of the data.

Although both approaches could be used depending on whether one is interested in a narrow band or a broad band synchronization, in our analysis of experimental data, we focused our attention on phase synchronization in a broad band sense.

10.2.2 Phase Synchronization

As mentioned above, synchronized systems exhibit a stable relationship between their respective rhythms called phase-locking. In terms of the instantaneous phases, the phase-locking relationship between $x(t)$ and $y(t)$ can be defined as

$$n\,\phi_x(t) - m\,\phi_y(t) = const \tag{10.8}$$

where $\phi_x(t)$ and $\phi_y(t)$ are the instantaneous phases of $x(t)$ and $y(t)$ respectively, and n and m are integers that define the phase-locking ratio.

Several techniques for examining phase synchronization have been proposed. One of the most commonly used among these approaches is based on the concept of the phase locking value [10]. The phase locking value (PLV) at time t is defined via the average of the phase differences among N trials as follows:

$$PLV(t) = \frac{1}{N}\left|\sum_{n=1}^{N} \exp\left\{i\left(\phi_x^{(n)}(t) - \phi_y^{(n)}(t)\right)\right\}\right| \tag{10.9}$$

where $\phi_x^{(n)}(t)$ and $\phi_y^{(n)}(t)$ denote the instantaneous phase estimates in trial n for $x(t)$ and $y(t)$, respectively, and i symbolizes an imaginary unit.

The PLV quantifies the internal variability of the difference of instantaneous phases at time t on average across all the trials. To detect statistically significantly different PLV values, the test is constructed using the surrogate data technique, developed in [20].

There are two main disadvantages of the PLV approach to measuring phase synchronization. First, it requires performing multiple trials while assuming the phase-locking ratio of 1:1. Second, the PLV method is inherently bivariate and cannot be easily extended to study phase synchronization among multiple systems. On the contrast, an approach based on the idea of generalized phase synchronization that has been recently developed in [8] can be applied in either bivariate or multivariate case.

10.2.3 Generalized Phase Synchronization

The concept of generalized phase synchronization is based on modeling of the instantaneous phases of multiple systems by means of cointegrated vector autoregressive processes.

Suppose the measurements of dynamical changes in K interacting systems are collected. Let $\phi_i(t)$ denote the instantaneous phase of system i $(1 \le i \le K)$ at time t. Combining single time series of $\phi_i(t)$ together for all K systems into a common K-dimensional process, we obtain $\phi(t) = (\phi_1(t), \phi_2(t), \dots, \phi_K(t))^\top$. Using vector autoregressive (VAR) processes, multiple series $\phi(t)$ of instantaneous phases can be modeled as follows:

$$\phi(t) = \mu + A_1\phi(t-1) + A_2\phi(t-2) + \cdots + A_p\phi(t-p) + \varepsilon(t), \\ t = 0, \pm 1, \pm 2, \dots \tag{10.10}$$

where $\mu = (\mu_1, \mu_2, \dots, \mu_K)^\top$ is a fixed $K \times 1$ vector of process mean $E\phi(t)$, A_i, $i = 1, \dots, p$ are fixed $K \times K$ real matrices of regression coefficients, and $\varepsilon(t) = (\varepsilon_1(t), \varepsilon_2(t), \dots, \varepsilon_K(t))^\top$ is a K-dimensional white noise process.

The conditions of stability and stationarity along with Gaussian distributed noise are the usual assumptions for vector autoregressive processes [12]. In practice stability and stationarity assumptions are often too restrictive. In fact, many real-life time series, including those representing biophysical signals such as electroencephalogram, have more complex nature and, therefore, are better fit by unstable, nonstationary processes. The stationarity assumption can easily be relaxed by successively estimating the model parameters on relatively short time intervals. The stability condition assumes that the reverse characteristic polynomial $RCP(z)$ has no roots on or inside the complex unit circle:

$$RCP(z) = \det\bigl(I_K - A_1 z - A_2 z^2 - \cdots - A_p z^p\bigr) \neq 0 \quad \text{for complex } z, \quad |z| \le 1 \tag{10.11}$$

where I_K is a $(K \times K)$ identity matrix.

A special class of autoregressive processes for which this condition is violated are integrated processes. Here, we assume that some of the univariate components in the multiple series $\phi(t)$ of instantaneous phases in (10.10) may be integrated.

The generalized phase synchronization is then defined by introducing the relationship among univariate components of phase series, which extends the bivariate condition (10.8) as follows:

$$c\phi(t) = c_1\phi_1(t) + c_2\phi_2(t) + \cdots + c_K\phi_K(t) = \eta(t) \tag{10.12}$$

where $c = (c_1, c_2, \dots, c_K)^\top$ is a $(K \times 1)$ real vector, and $\eta(t)$ is a stationary stochastic process, which represents the deviation from the constant relation (equilibrium). The relationship (10.12) also implies that the multiple time series $\phi(t)$ of instantaneous phases are cointegrated.

A cointegrated vector autoregressive process $\phi(t)$ is said to be cointegrated of rank r, if the correspondent matrix $\Pi = I_K - A_1 - A_2 - \cdots - A_p$ has rank r. Cointegration rank r is an important characteristic of cointegrated processes. As shown in [8], the cointegration rank can be interpreted as a measure of the generalized phase synchronization in the multiple time series.

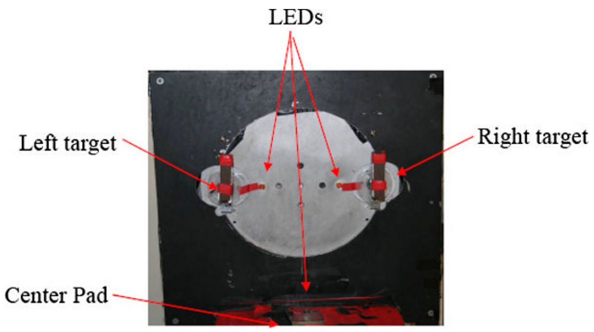


Fig. 10.1 The setup, in which a monkey repeatedly performed the reach to grasp task

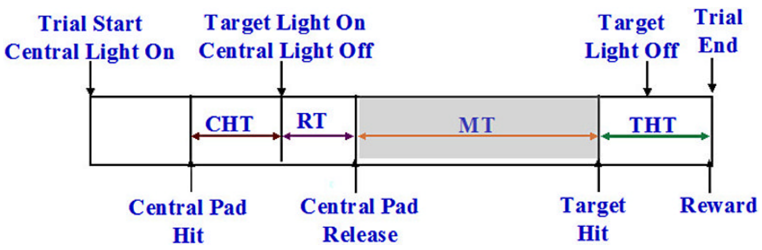


Fig. 10.2 The five major stages of a successful trial, where CHT, RT, MT, and THT denote time intervals between Center Pad Hit and Central Light Off/Target Light On, Target Light On and Central Pad Release, Pad Release and Taget Hit, and Target Hit to Trial End, respectively

10.3 Analysis of the Data Collected During Sensory-Motor Experiments

This section describes some peculiar patterns of the instantaneous phases of the data collected from motor and premotor cortical areas during the sensory-motor experiments. In particular, the computational results show transient integration in subgroups of neural channels.

10.3.1 Sensory-Motor Experiments and Neural Data Acquisition

To investigate synchronization of neural activity in the brain of primates during planning, execution and control of upper limb movements, local field potential (LFP) data was collected using chronically implanted microwire arrays in the following experiments. A rhesus macaque was trained to perform reach and grasp tasks in response to available visual clues. As depicted in Fig. 10.1, the setup, in which the monkey performed the tasks, included LED lights (to signal visual clues), a center pad and two targets (left and right). Although the experiment focused on the five key stages, the neural signals were recorded continuously, including in between consecutive trials.

Each successful trial can be subdivided into the five stages (or phases) according to the intervals shown in Fig. 10.2. The monkey was trained to touch the central holding pad when the central LED was turned on. The first stage of each trial began with the central LED light on and lasted until the animal hit the central pad. As soon as the animal placed its hand on the pad, a target LED would light up, providing the monkey with a visual cue as to which target, left or right, must be reached for. So, the second stage was the time immediately following the center pad hit and preceded the central light being shut off and instead one of two target LEDs being turned on. In fact, after a short delay, the central LED would shut off indicating to the animal to begin reaching for the target according to which of two target LEDs was activated. That is when stage three began, which lasted until the monkey released the central pad. The period from the time the animal released the central pad and until the target was grabbed by the monkey was defined as the fourth stage. Upon grasping the target in a power grip, the animal were trained to hold the target for a predefined period of time after which the animal would be rewarded. So, the fifth and final stage spanned the time from the target hit to the animal receiving the reward, which concluded the trial.

LFP data was recorded using two chronically implanted microwire arrays. Each array included two rows of eight microwire electrodes. The length of wire varied as shown on Fig. 10.3 and the array was designed in such a way that each wire had a length of 1.5 mm, 2 mm, or 2.5 mm from the tip of the electrode to the array's land. The purpose of the land was to keep the electrodes evenly spaced and to limit the wire depth during an insertion. The diameter of each electrode was 50 microns, while the spacing between electrodes in each row was 250 microns and the spacing between each row was 500 microns.

Recordings were performed from the motor and premotor cortical regions, which have been shown to contribute to planning and execution of upper limb movement. In particular, neurons located in the motor area are responsible for activation of muscles in the upper limb, whereas the neurons in the premotor regions can be involved in planning of a motor action.

10.3.2 Computational Analysis of the LFP Data

The collected experimental data was analyzed statistically to investigate presence of generalized synchronization during normal sensory-motor activity. Since two microwire arrays, each having two 8-electrode rows, were used during recordings and the data was sampled at 20 kHz, we obtained extended 32-dimensional time series of LFP measurements. Most of the computations were coded and implemented in the R 2.8.1 statistical software. First, the corresponding instantaneous phases were computed for each single time series based on the Hilbert transform approach as described in Sect. 10.2.1. Next, individual univariate time series of instantaneous phases for different electrode channels were grouped together into respective multiple series according to specific rows in the arrays. As a result, we obtained four

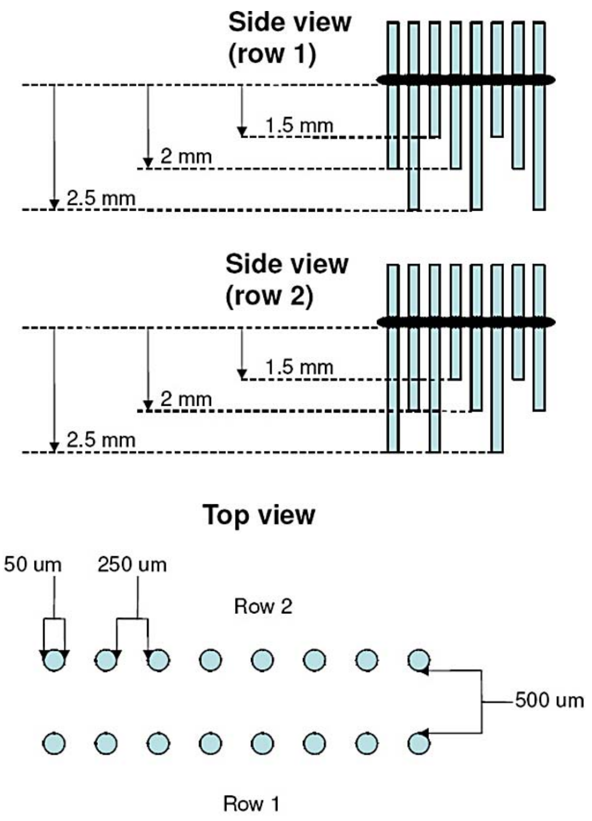


Fig. 10.3 Illustration of microwire array placement for neural data acquisition. Each microwire array had 16 electrodes in two rows. Side view for row 1, side view for row 2, and top view of both rows are displayed (from *top* to *bottom*)

8-dimensional time series of the instantaneous phases, each representing neural activity measured from the eight electrodes in a given row of two microwire arrays. Finally, VAR modeling and testing were performed separately on each of four grouped multiple time series.

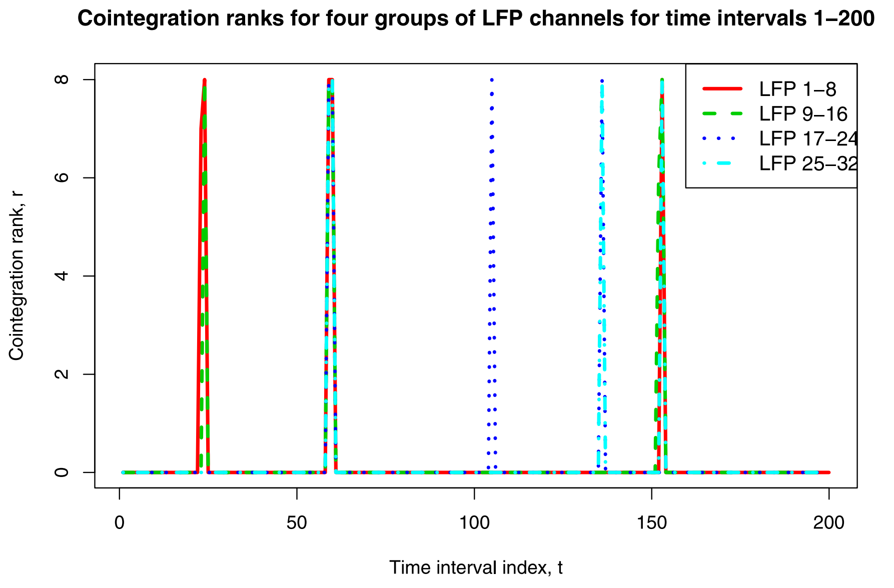
Four 8-dimensional time series of the instantaneous phases corresponded to four groups of channels from which local field potentials were collected, i.e., the first series included LFP 1–8, the second combined LFP 9–16, the third LFP 17–24, and the fourth LFP 24–32. To study synchronization in each of these multiple series, we applied the generalized phase synchronization approach described in Sect. 10.2.3. Since neural signals are nonstationary and we focused on transient synchronization, the time was subdivided into relatively small intervals of 0.25 seconds with 500 sample points. This procedure produced over 2000 time intervals. The time scale was the same for all four multiple series. Next, for each 8-dimensional time series we estimated the VAR model parameters p, A_i, and μ on each time interval. By applying Johansen–Juselius cointegration rank procedure [7] with p-values of 0.01 to test the rank of the estimated VAR, we found the cointegration rank values for each multiple time series of phases on every time interval.

Figures 10.4 and 10.5 illustrate the dynamic changes in the generalized phase synchronization of four different groups of eight microwire electrodes by plotting the respective cointegration rank of the respective 8-dimensional phase series against time interval values. The top picture in Fig. 10.4 and the bottom picture in Fig. 10.5 include a legend that specifies which group (or row) of channels is represented by each line. Four plots are displayed, each containing 200 consecutive time intervals. The horizontal axis represents time interval indexes, and the vertical axis denotes the value of cointegration rank. In Fig. 10.4, the top plot depicts cointegration rank values for time intervals 1–200, and the bottom plot shows cointegration ranks on intervals 201–400. In Fig. 10.5, the top and the bottom plots correspond to intervals 401–600 and 601–800, respectively.

As can be seen from Figs. 10.4 and 10.5, most of the time all four groups of channels exhibit a zero cointegration rank, which implies the absence of generalized phase synchronization. On the other hand, at certain times there are sudden short spikes in cointegration rank that can be detected in all four 8-channel rows. Interestingly, the patterns of the sharp transient increases in the rank values are complex, characterized by a number of different combinations of rows often displaying a simultaneous increase in their respective cointegration ranks.

Furthermore, the increased values of cointegration ranks range between 4–8, with the cointegration rank of 8 being observed the second most often after the prevalent rank of 0. The patterns in the range of cointegration rank values were further studied using scatter plots of the experimental stage against the cointegration rank values during the respective stage. Recall from Sect. 10.3.1 that each successful trial in the experiment consisted of five distinct stages, and note that the intervals in between trials were designated as experimental stage 0.

Figures 10.6 and 10.7 display the scatter plots of experimental stage versus the respective cointegration ranks for each of the four groups of LFP channels. The indexes of experimental stages, ranging from 0 to 5, are represented by the horizontal axis, and the cointegration rank values found during each stage are displayed on the vertical axis and range from 0 to 8. Although all the coordinates (the stages and the ranks) are integers, we added a random jitter to the cointegration rank values in order to be able to display multiple points having the same coordinates. The four scatter plots in Figs. 10.6–10.7 correspond to four different groups of channels, with the top left plot displaying stage indexes versus cointegration ranks for LFP 1–8, the top right presenting the scatter plot for LFP 9–16, and the bottom left and bottom right plots illustrating the stage/rank relationship for LFP 17–24 and LFP 25–32 groups, respectively. Figures 10.6 and 10.7 highlight some interesting similarities among the four different groups of electrode, each of which represents a single row of the microwire array. In particular, whereas no synchronized activity in terms of generalized phase synchronization can be detected at stages 2 and 3 (after the center pad is hit and until it is released) in all four groups with a single instance exception of stage 3 for LFP 17–24, stages 0, 1, 4, and 5, are all characterized by a short but strong presence of generalized phase synchronization. Although the rank results for stage 0 (between trials) are difficult to interpret with certainty, the cointegration of the instantaneous phases during experimental stage 1 may be attributed to the motor planning induced by activation of the central light at the beginning of each trial and/or execution of motor action that leads to central pad hit. Similarly, for experimental stage 4, the positive and large rank values may be associated with execution



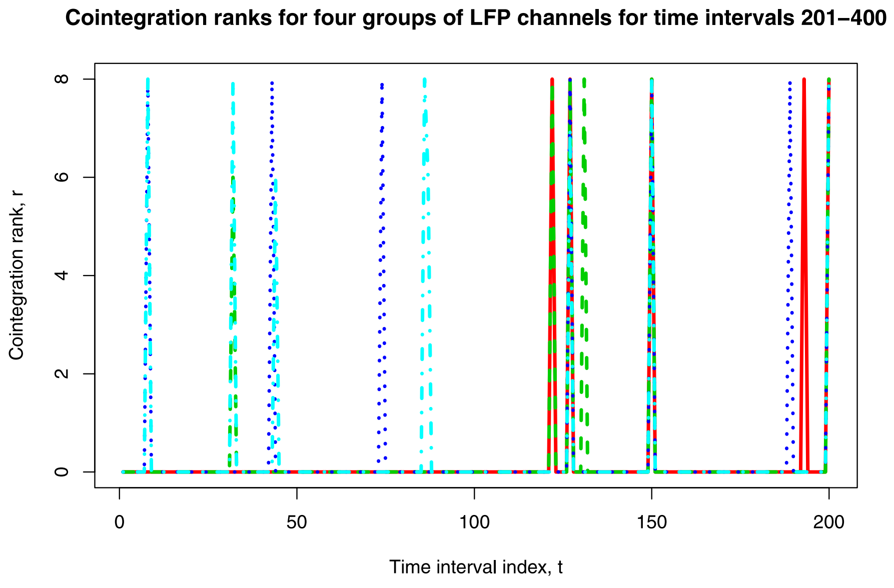
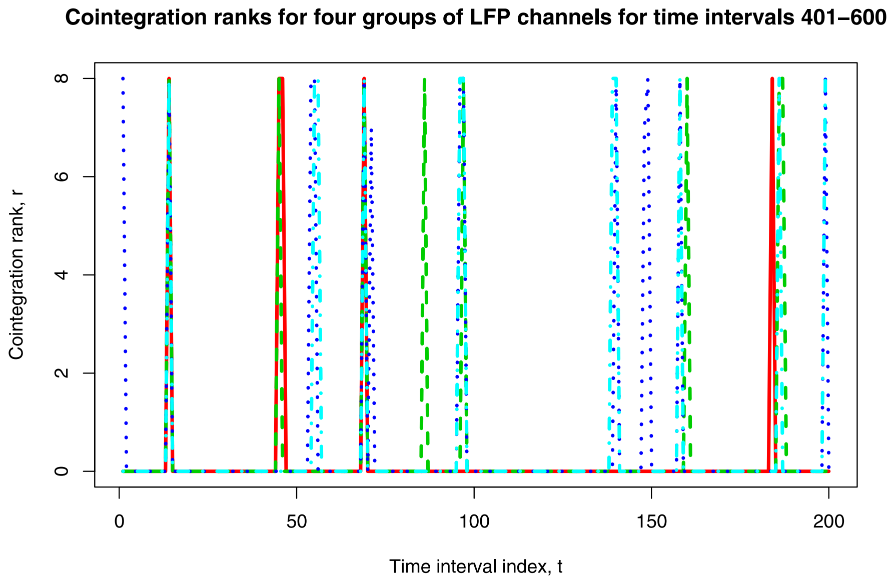


Fig. 10.4 Linear plots of cointegration rank values on 200 consecutive time intervals for four groups of LFP channels. *Top*: First 200 time intervals (i.e., intervals 1–200). *Bottom*: Time intervals 201–400



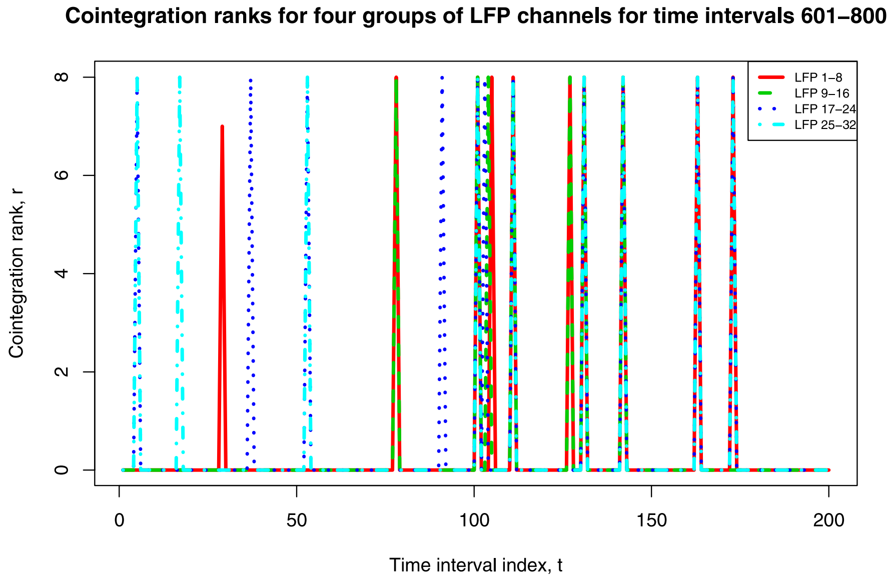
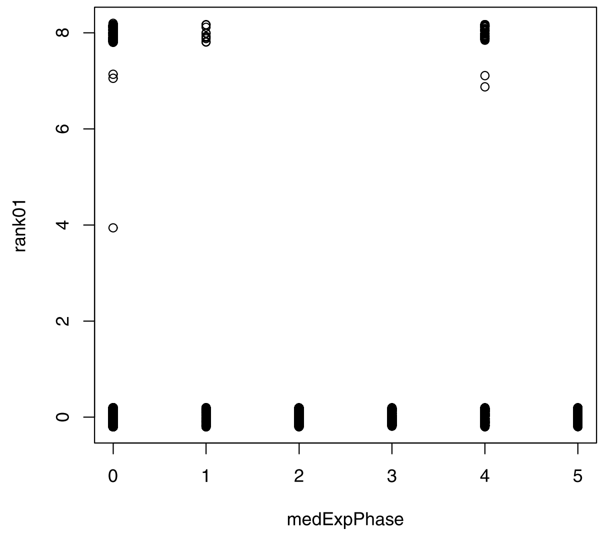


Fig. 10.5 Linear plots of cointegration rank values on 200 consecutive time intervals for four groups of LFP channels. *Top*: Time intervals 401–600. *Bottom*: Time intervals 601–800

of the movement beginning upon the release of the central pad and ending with the target being hit. Finally, the evidence of short-term synchronization at stage 5 may be connected to maintaining the grasp for a certain time until the reward is given.



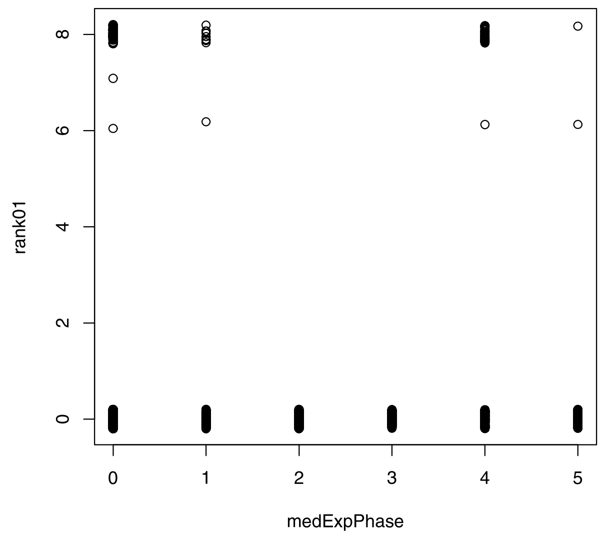


Fig. 10.6 Scatter plots of cointegration rank vs. experimental stage for four groups of LFP channels. A random jitter is added to the cointegration rank values to display coinciding points. *Top*: LFP 1–8. *Bottom*: LFP 9–16

10.4 Conclusion

Experimental data recorded during reach and grasp experiments in monkeys were analyzed using a novel technique of generalized phase synchronization. Our investigation into the dynamics of neural activation in the motor and premotor cortical areas engaged in movement planning and execution discovered the presence of short term cointegration in the instantaneous phases of neural channels located in a common row of the implanted microwire arrays. Since short-term synchronization of neuronal discharges is widely believed in neuroscience research community to be responsible for the dynamical integration of activity of distributed groups of neurons

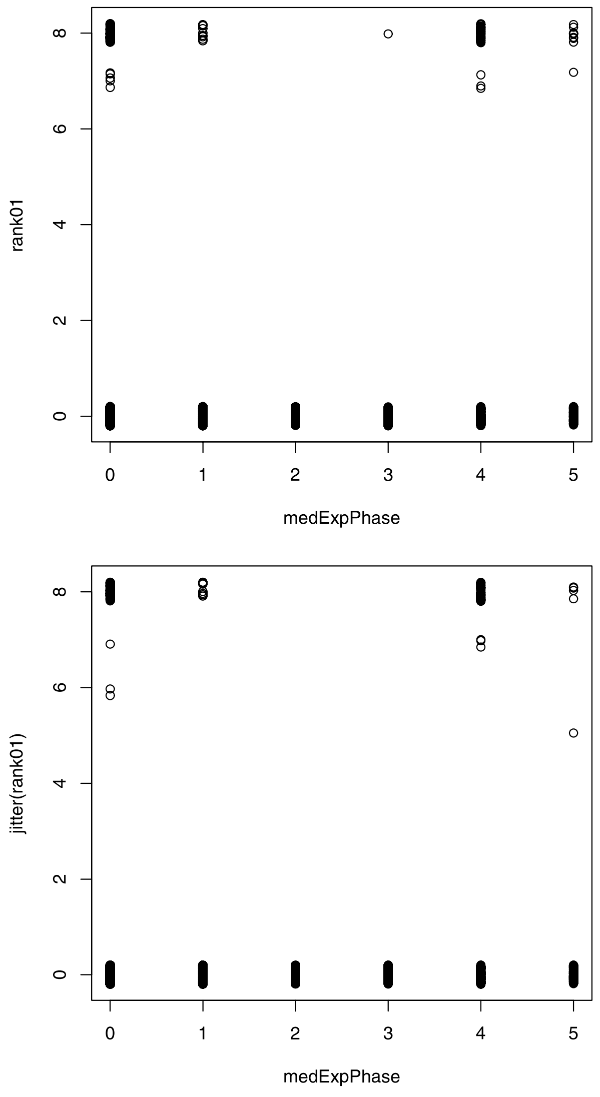


Fig. 10.7 Scatter plots of cointegration rank vs. experimental stage for four groups of LFP channels. A random jitter is added to the cointegration rank values to display coinciding points. *Top*: LFP 17–24. *Bottom*: LFP 25–32

that is necessary for execution of cognitive and motor tasks, the strong presence of cointegration in instantaneous phases may be attributed to transient fusion of neural signals.

References

1. Afraimovich, V.S., Verichev, N.N., Rabinovich, M.I.: General synchronization. Izv. Vysš. Učeb. Zaved. Radiofiz. **29**, 795–803 (1986)
2. Bronshtein, I.N., Semendyayev, K.A.: Handbook of Mathematics, 3rd edn. Springer, New York (1997)

3. Fries, P., Reynolds, J.H., Rorie, A.E., Desimone, R.: Modulation of oscillatory neuronal synchronization by selective visual attention. Science **291**, 1560–1563 (2001)
4. Fujisaka, H., Yamada, T.: Stability theory of synchronized motion in coupled-oscillator systems. Prog. Theor. Phys. **69**, 32–47 (1983)
5. Gabor, D.: Theory of communication. Proc. IEEE Lond. **93**, 429 (1946)
6. Gray, C.M., Koenig, P., Engel, A.K., Singer, W.: Oscillatory responses in cat visual cortex exhibit inter-columnar synchronization which reflects global stimulus properties. Nature **23**, 334–337 (1989)
7. Johansen, S., Juselius, K.: Maximum likelihood estimation and inference on cointegration – with applications to the demand for money. Oxford Bull. Econ. Stat. **52**, 169–210 (1990)
8. Kammerdiner, A.: Statistical methods in data mining of brain dynamics. PhD dissertation, University of Florida, Gainesville, FL (2008)
9. Kitajo, K., Doesburg, S.M., Yamanaka, K., Nozaki, D., Ward, L.M., Yamamoto, Y.: Noise-induced large-scale phase synchronization of human-brain activity associated with behavioural stochastic resonance. Europhys. Lett. (2007). doi:10.1209/0295-5075/80/40009
10. Lachaux, J.P., Rodriguez, E., Martinerie, J., Varela, F.J.: Measuring phase synchrony in brain signals. Hum. Brain Mapp. **8**, 194 (1999)
11. Le Van Quyen, M., Foucher, J., Lachaux, J.-P., Rodriguez, E., Lutz, A., Martinerie, J., Varela, F.J.: Comparison of Hilbert transform and wavelet methods for the analysis of neuronal synchrony. J. Neurosci. Methods **111**(2), 83–98 (2001)
12. Lütkepohl, H.: Introduction to Multiple Time Series Analysis. Springer, Berlin (1991)
13. Melloni, L., Molina, C., Pena, M., Torres, D., Singer, W., Rodriguez, E.: Synchronization of neural activity across cortical areas correlates with conscious perception. J. Neurosci. **27**, 2858–2865 (2007)
14. Miltner, W.H.R., Braun, C., Arnold, M., Witte, H., Taub, E.: Coherence of gamma-band EEG activity as a basis for associative learning. Nature **397**, 434–436 (1999)
15. Pikovsky, A., Zaks, M., Rosenblum, M., Osipov, G., Kurths, J.: Phase synchronization of chaotic oscillations in terms of periodic orbits. Chaos **7**, 680–687 (1997)
16. Quian Quiroga, R., Kraskov, A., Kreuz, T., Grassberger, P.: Performance of different synchronization measures in real data: A case study on electroencephalographic signals. Phys. Rev. E **65**, 041903 (2002)
17. Rodriguez, E., George, N., Lachaux, J.-P., Martinerie, J., Renault, B., Varela, F.J.: Perception's shadow: long-distance synchronization of human brain activity. Nature **397**(6718), 430–433 (1999)
18. Roelfsema, P.R., Engel, A.K., Konig, P., Singer, W.: Visuo-motor integration is associated with zero time-lag synchronization among cortical areas. Nature **385**, 157–161 (1997)
19. Singer, W.: Neuronal synchrony: A versatile code of the definition of relations? Neuron **24**, 49–65 (1999)
20. Theiler, J., Eubank, S., Longtin, A., Galdrikian, B., Farmer, D.: Testing for nonlinearity in time series: The method of surrogate data. Physica D **58**(1–4), 77–94 (1992)
21. Tononi, G., Srinivasan, R., Russell, D.P., Edelman, G.M.: Investigating neural correlates of conscious perception by frequency-tagged neuromagnetic responses. Proc. Natl. Acad. Sci. **95**, 3198–3203 (1998)
22. Uhlhaas, P., Singer, W.: Neural synchrony in brain disorders: Relevance for cognitive dysfunctions and pathophysiology. Neuron **52**, 155–168 (2006)
23. van der Pol, B., van der Mark, J.: The heartbeat considered as a relaxation oscillation, and an electrical model of the heart. Philos. Mag. **6**, 763 (1928)
24. Varela, F.J., Lachaux, J.-P., Rodriguez, E., Martinerie, J.: The brainweb: phase synchronization and large-scale integration. Nat. Rev. Neurosci. **2**(4), 229–39 (2001)
25. von Stein, A., Sarnthein, J.: Different frequencies for different scales of cortical integration: from local gamma to long range alpha/theta synchronization. Int. J. Psychophysiol. **38**, 301–313 (2000)

Chapter 11
An Info-Centric Trajectory Planner for Unmanned Ground Vehicles

Michael A. Hurni, Pooya Sekhavat, and I. Michael Ross

Summary We present a pseudospectral (PS) optimal control framework for autonomous trajectory planning and control of an Unmanned Ground Vehicle (UGV) with real-time information updates. The algorithm is introduced and implemented on a collection of motion planning scenarios with varying levels of information. The UGV mission is to traverse from an initial start point and reach the target point in minimum time, with maximum robustness, while avoiding both static and dynamic obstacles. This is achieved by computing the control solution that solves the initial planning problem by minimizing a cost function while satisfying dynamical and environmental constraints based on the initial global knowledge of the area. To overcome the problem of incomplete global knowledge and a dynamic environment, the UGV uses its sensors to map the locally detected changes in the environment and continuously updates its global map. At each information update, the optimal control is recomputed and implemented. Simulation results illustrate the performance of the planner under varying levels of information.

11.1 Introduction

Autonomous trajectory planning of unmanned vehicles has been one of the main goals in robotics for several years. In recent years, this problem has become particularly important as a result of rapid growth in its applications to both military and civilian missions. Various control methods have been proposed and examined for autonomous guidance and control of unmanned vehicles [4, 13].

There are two approaches to optimal trajectory planning for a dynamic system: The decoupled approach and the direct approach. The decoupled approach involves

M.A. Hurni (✉) · P. Sekhavat · I.M. Ross
Naval Postgraduate School, Monterey, CA 93943, USA
e-mail: mahurni@nps.edu

P. Sekhavat
e-mail: psekhava@nps.edu

I.M. Ross
e-mail: imross@nps.edu

first searching for a path (using a path planner) and then finding a time-optimal control solution on the path subject to the actuator limits. The direct approach searches for the optimal control trajectory directly within the system's state space [4].

Optimal control trajectory planning using numerical optimization, as described in [4], is a direct approach to the complete motion-planning problem, which determines the path to the target by searching for the optimal control trajectory within the vehicle's state space. The result is the complete state space and control solution from start to goal. The basic concept of how optimal path planning works follows from [4, 9]. The planner is given the kinodynamic equations of the vehicle, the obstacles' approximate location and geometry (to be coded into smooth path constraint functions), and the mission's boundary conditions and cost function. The kinodynamic equations can also be viewed as constraints (similar to the obstacles), defining the relationship between the vehicle state and the control input. The actual obstacles' geometry need not be smooth, but the constraints used to mathematically define the obstacles must be made up of one or more smooth functions. The optimal control technique finds a solution to the state equations that takes the vehicle from the initial state at time zero to the final state at the final time, while avoiding obstacles, obeying vehicle state and control limits, and minimizing the cost function. The cost function can be any function of state variables, control variables and time, as long as it is sufficiently smooth (i.e., continuous and differentiable).

Recent advances in optimal control and numerical optimization tools have made the method a viable approach for many applications. It has been demonstrated that using the initial (open-loop) trajectory generated off-line as the bias (guess) can produce run times of sufficient speed. Subsequently, each real-time solution can be used as the bias for the next run, thus maintaining solution speed and autonomy. This approach involves using the optimal control algorithm in a feedback form (closed loop), thus self-generating the needed bias. References [1–3, 6–8, 11, 12] solve various path planning problems using optimal control with numerical optimization in a feedback control algorithm. In [1, 3, 6–8, 11], the use of optimal control in a feedback form using a bias that is obtained autonomously is demonstrated on a Reusable Launch Vehicle, a tricycle, an Unmanned Ground Vehicle, an Unmanned Air Vehicle, and a spacecraft. In reference to the computational speed (cost), it has been shown in [1–3, 6–8, 11, 12] that the use of a bias to help steer the solution trajectory speeds up the run times significantly from seconds to fractions of a second. So the bias does not just aid in achieving feasible solutions, but also speeds up the solution process considerably. Also, advancements in sparse linear algebra, development of new algorithms, and improved computer processor speeds have made solving optimization problems relatively easy and fast [3]. Recent applications of real-time optimal control [1–3, 6–8, 11, 12] have proven to be very promising in facilitating feedback solutions to complex nonlinear systems [3].

This paper presents the implementation of a pseudospectral (PS) optimal control framework for autonomous trajectory planning and control of a UGV with real-time information updates on numerous motion planning scenarios with varying levels of available information. Simulation results illustrate the effects of varying levels of information and how more or less information would influence the performance of the planner.

11.2 Problem Formulation and Background

The UGV that is used in this paper is modeled as a four-wheeled car with front-wheel steering [6]. Only the front wheels are capable of turning and the back wheels must roll without slipping. L is the length of the car between the front and rear axles. r_t is the instantaneous turning radius. The state vector is composed of two position variables, x and y (m), an orientation variable, θ (rad), the car's velocity, v (m/sec) and the angle of the front wheels, ϕ (rad) with respect to the car's heading, (11.1). The x–y-position of the car is measured at the center point of the rear axle.

$$\underline{x} \in X := \left\{ \begin{array}{c} x : x_{\min} \le x(t) \le x_{\max} \\ y : y_{\min} \le y(t) \le y_{\max} \\ \theta : -2\pi \le \theta(t) \le 2\pi \\ v : -1 \le v(t) \le 1 \\ \phi : -1 \le \phi(t) \le 1 \end{array} \right\} \tag{11.1}$$

The control vector consists of the vehicle's acceleration, a (m/sec^2) and the rate of change of the front wheel angle, ω (rad/sec), (11.2).

$$\underline{u} \in U := \left\{ \begin{array}{c} a : -0.5 \le a(t) \le 0.5 \\ \omega : -0.33 \le \omega(t) \le 0.33 \end{array} \right\} \tag{11.2}$$

Using the fact that $L = r_t \tan(\phi)$ and $v = r_t \dot{\theta}$, assigning the length $L = 0.5$ m results in the following kinematic equations of motion for the UGV:

$$\dot{\underline{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ 2v\tan(\theta) \\ a \\ \omega \end{bmatrix} \tag{11.3}$$

The p-norm was used to algebraically model the shapes of the obstacles used in this work. Using the p-norm, one can easily model any square, rectangle, circle, or ellipse. These shapes are all that is needed in path planning since any obstacle can be modeled by fitting one of those shapes around it. Modeling an obstacle's exact shape and size is more complex and unnecessary. Equation (11.4) shows the general form of the equation used to model the obstacles.

$$h_i\big(x(t), y(t)\big) = \left| \left(\frac{x(t) - x_c}{a} \right)^p \right| + \left| \left(\frac{y(t) - y_c}{b} \right)^p \right| - \left| c^p \right| = 0 \tag{11.4}$$

Development of the cost function is discussed next. The objective is to minimize maneuver time while simultaneously maximizing robustness. Robustness is characterized by the vehicle being able to execute the mission in the presence of a certain amount of uncertainty and not hit any obstacles. This uncertainty can come from numerous sources: Approximations in the modeling of the vehicle, errors in sensor data, sensor accuracy, external unforeseen forces, and the error that is inherent in executing the previous control solution while calculating an updated solution (the lag between the two). The control interpolation error comes from the fact that low node solutions are necessary to effect lower run times, which are required for real time operation. A node is simply a discrete time point that describes the complete vehicle state and control vectors at that specific time. If high number of nodes are used, the trajectory can be made collision free, but at the expense of higher algorithm run times. The proposed solution to this problem is the addition of the robustness factor to a low-node problem definition. Since we desire to include robustness in the cost function, it is necessary to derive a Robustness Function, $r(t)$, whose integral over time will be minimum when robustness is maximized. To derive $r(t)$, we take (11.4) (which has a value of zero when on the obstacle and increases as you move away from the obstacle) and apply a double exponential operation on it [6]. This is done for each obstacle and the result is summed. The resulting $r(t)$ with n equal to the number of obstacles is (11.5). In the case of a single obstacle, $r(t) = 14.15$ at its center, $r(t) = 1.72$ on its edge, and $r(t)$ will continue to decrease exponentially to zero as the distance to the obstacle is increased. Equation (11.6) is the final cost function expressed in terms of the endpoint cost (final time) and the running cost (integral of $r(t)$) with appropriate weighting factors (ω_{t_f}, ω_r).

$$r(t) = \sum_{i=1}^{n} \left(e^{e^{-h_i(x(t),y(t))}} - 1 \right) \tag{11.5}$$

$$J\left[\underline{x}(.), \underline{u}(.), t_f\right] = \omega_{t_f} t_f + \omega_r \int_0^t r(t)\, dt \tag{11.6}$$

More details of UGV modeling/kinematics, obstacle modeling, and the trajectory planning framework can be found in [5]. For all scenarios in this work, the UGV mission is to traverse from an initial start point and reach the target point in minimum time, with maximum robustness, while avoiding both static and dynamic obstacles. This is achieved by deriving the control solution that carries out the initial planning problem while minimizing a cost and satisfying dynamical and environmental constraints based on the initial global knowledge of the area. To overcome the problem of incomplete global knowledge and a dynamic environment, the UGV uses its sensors to map the locally detected changes in the environment and continuously updates its global map; then recomputes and updates the optimal trajectory at each information update [5]. The numerical solver used to compute the optimal trajectories is the DIDO software package [10], which is a MATLAB based software package that utilizes pseudospectral methods to determine an extremal for a properly formulated optimal control problem.

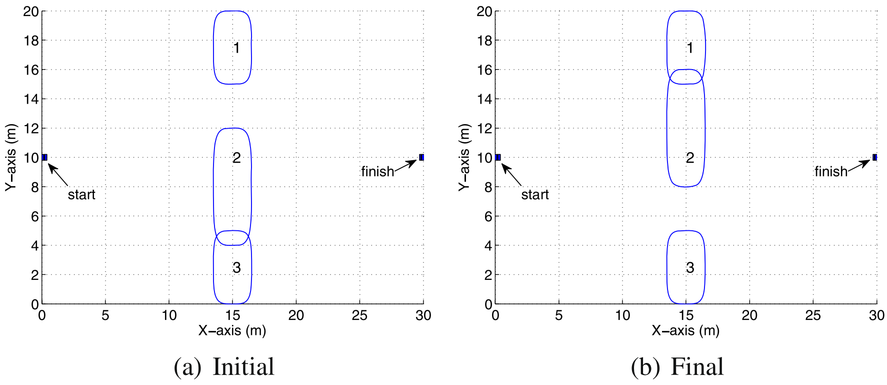


Fig. 11.1 Configuration for sliding door problem

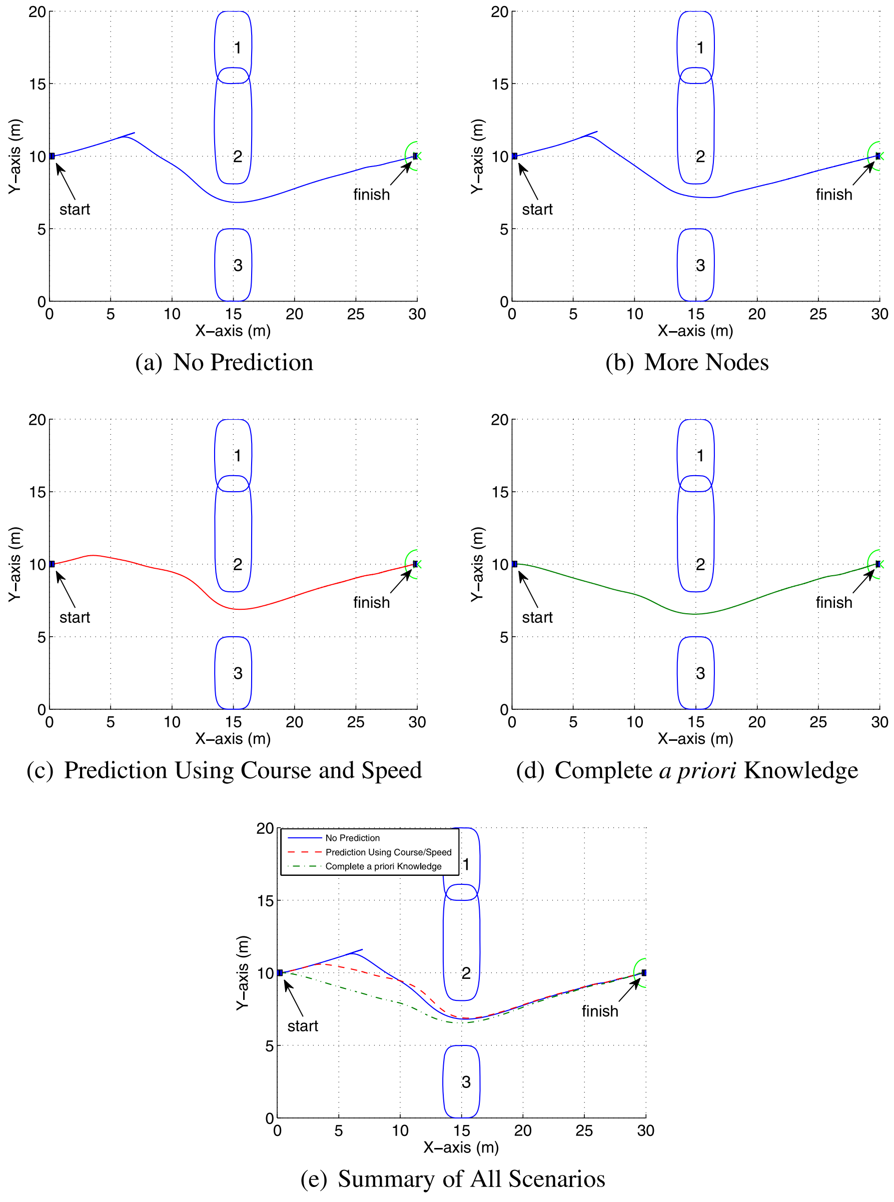
11.3 Obstacle Motion Studies

11.3.1 The Sliding Door

We start with the sliding door scenario of Figs. 11.1(a) and 11.1(b), which show the initial and final configurations of the environment obstacle, respectively. The start and finish positions of the vehicle are as shown. All obstacles and the vehicle are initially at rest. At 3 seconds into the maneuver, obstacle 2 moves north at a speed of 0.5 m/sec, gradually blocking the north passage and opening up the south passage.

The first simulation was completed using snapshots of the environment taken just prior to each iteration of the algorithm, with no prediction of obstacle position. Thus, during the time it takes to generate a new solution, the vehicle maneuvers based on the previous solution with no knowledge of obstacle motion until the next snapshot is taken and the vehicle "senses" the environment has changed. Each run produces a trajectory that solves the instantaneous static problem, even though the environment is dynamic. The resulting trajectory is shown in Fig. 11.2(a). The vehicle heads toward the north passage until it is closed off, at which point the vehicle stops, repositions (as required by the algorithm), generates a new bias, and then continues along the new path through the south passage. The trajectory in this case is not perfectly smooth, which can be attributed to the fact that a low node PS-solution results in higher propagation error. Figure 11.2b shows the smooth trajectory that could be achieved with higher node solutions. Less smooth trajectories are acceptable in trying to keep the number of nodes (and thus the run times) to a minimum.

The sliding door scenario was repeated with the addition of obstacle position prediction using course and speed. This can be achieved simply by comparing successive environment snapshots to estimate a moving obstacle's course and speed, and then using that course and speed to predict the obstacle's future position as part of the trajectory planning problem. Figure 11.2(c) shows the resulting trajectory. The vehicle initially heads toward the northern passage. Once obstacle 2 begins to move north, the algorithm determines the obstacle's future position using its course



(a) No Prediction

(b) More Nodes

(c) Prediction Using Course and Speed

(d) Complete *a priori* Knowledge

(e) Summary of All Scenarios

Fig. 11.2 Various scenarios for sliding door problem

and speed, resulting in the vehicle autonomously changing its course to steer clear of the obstacle. Obstacle 2 then stops moving, resulting in the vehicle making a new course correction further to the south in order to steer clear. This course correction is necessary because while the obstacle was moving, the prediction of its position was based on the assumption that it would continue on its current course and speed. The trajectory planner generates each new trajectory based on the current information on obstacle positions, courses and speeds. It does not attempt to predict course or speed changes. Any changes in the obstacle's course and speed (including complete stoppage) will require the planner to correct the vehicle's trajectory accordingly.

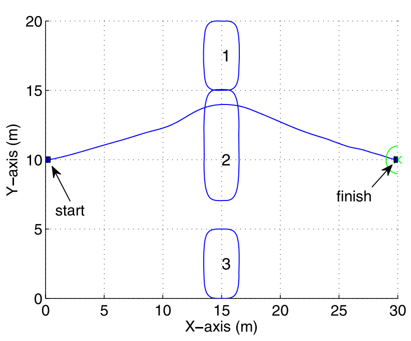
Finally, the sliding door scenario was repeated with complete a priori knowledge of obstacle 2 motion; i.e., all future course and speed changes of the obstacle are known in advance. Figure 11.2(d) shows the resulting trajectory. The vehicle, having complete knowledge of the future movement of obstacle 2, simply heads immediately down the optimal path through the south passage.

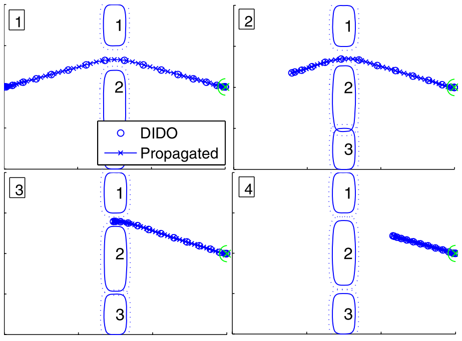
The three sliding door scenarios with various levels of information are plotted together on Fig. 11.2(e). The scenarios using no prediction and using course and speed for prediction start on the same trajectory, because obstacle 2 does not start moving until the three-second point into the simulations. Without obstacle motion, the two scenarios are identical. Once obstacle motion begins, using prediction results in a trajectory that is closer to the time-optimal solution based on complete a priori information. When the obstacle position is not predicted, the maneuver time is 41.3 seconds, which does not include the extra time it takes to calculate a new southerly bias when the north passage becomes blocked. When prediction is used, the maneuver time is 33.5 seconds. With complete a priori knowledge of the environment, the maneuver time is even shorter at 33 seconds.

11.3.2 The Cyclic Sliding Door

The sliding door scenario is modified so that the door continuously slides back and forth, thus alternately blocking the north and south passages. This type of problem provides the opportunity to examine the trajectory planning algorithm's performance at all three information levels (i.e., no prediction, prediction, and a priori knowledge). It allows us to simulate success or failure in algorithm performance by simply adjusting the cycling speed of obstacle 2 (i.e., raising and lowering its cycling frequency). It should be noted at the outset that given the maximum vehicle speed of 1 m/sec and the obstacle widths of 3 m (not including their expansion to account for vehicle size, see [5]), any obstacle 2 speed greater than 0.6 m/sec will result in failure, because the vehicle cannot traverse through the opening quickly enough. In other words, above a cycling speed of 0.6 m/sec there can be no solution to the problem due to the physical limit on the vehicle speed.

We start at the lowest information level (i.e., no prediction using course and speed) and the slowest cycling speed for obstacle 2 set at 0.1 m/sec. In this scenario, obstacle 2 moves so slowly that it never has a chance to reverse course and head south before the end of the vehicle's mission. In other words, it doesn't even complete a half cycle. The resulting trajectory is shown in Fig. 11.3(a). The position of obstacle 2 in Fig. 11.3(a) corresponds to its final position at the end of the scenario, and does not indicate its position at the time the vehicle passed through the north





(a) Overall Trajectory (b) Snapshots of Motion

Fig. 11.3 Cyclic sliding door (no prediction; 0.1 m/sec cycle speed)

passage. Figure 11.3(b) shows the evolution of the trajectory at 4 sample instances in time, and is composed of both the DIDO generated trajectory (circles corresponding to discrete nodes) and the propagated path. The propagated path is obtained by taking DIDO's discrete control solution, applying control trajectory interpolation, and then propagating the state trajectory using a Runge–Kutta algorithm. The fact that the propagated and DIDO solutions fall on top of each other is the proof of dynamic feasibility of the control solution. Obstacle 2's movement is slow enough that the vehicle need not change course to traverse the passage. The maneuver time for this scenario was 33.5 seconds. Future figures showing the full vehicle trajectory from start to finish will indicate obstacle positions at the final time only (as in Fig. 11.3(a)). Snapshots in time (as in Fig. 11.3(b)) will be given where appropriate.

The cycle speed of the sliding door was then raised to 0.2 m/sec, which prevented the vehicle from reaching the north passage in time to pass through. Since prediction is not being used, the vehicle does not know the north passage will be closed off until it actually happens. When the vehicle senses that the passage is blocked, it stops, repositions, and reformulates a new bias. Figure 11.4 shows the resulting collision (the dotted lines are the obstacle boundaries adjusted for the vehicle size). The obstacle is moving slow enough to infer a solution exists, but, without prediction, the vehicle cannot pass. The algorithm with this level of information also failed to safely guide the vehicle through the passage (north or south) for all other speeds up to the 0.6 m/sec limit. Suffice it to say that without more information the algorithm cannot safely guide the vehicle through this cyclic sliding door example.

The same scenarios presented above in Figs. 11.3(a) and 11.4 were simulated again using obstacle position predicted from course and speed data. Figure 11.5(a) shows the trajectory when the cycle speed is set to 0.1 m/sec. The maneuver time was 33.4 seconds. The obstacle speed is too slow to show any significant improvement of the trajectory over the case of not utilizing current course and speed data to calculate and include future obstacle positions while planning the vehicle's trajectory. The significance of again showing the evolution of the trajectory (Fig. 11.5(b)) is in the change of the trajectory in the second frame of the figure. We see the trajectory jump further away from the obstacle in frame 2, because the algorithm is

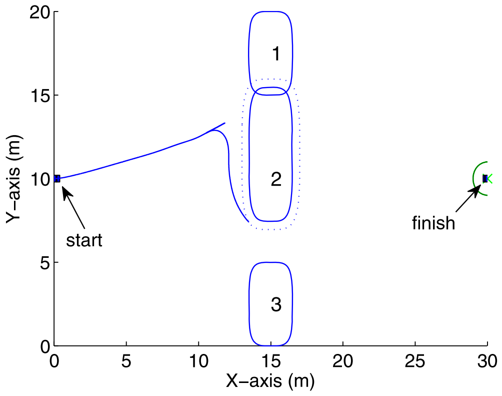
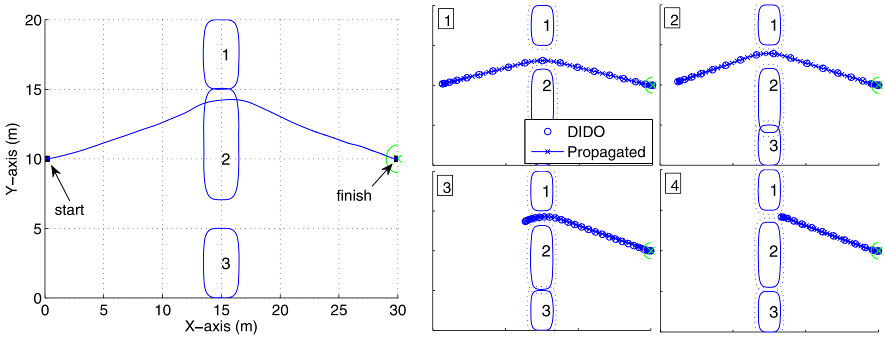


Fig. 11.4 Cyclic sliding door (no prediction; 0.2 m/sec cycle speed)

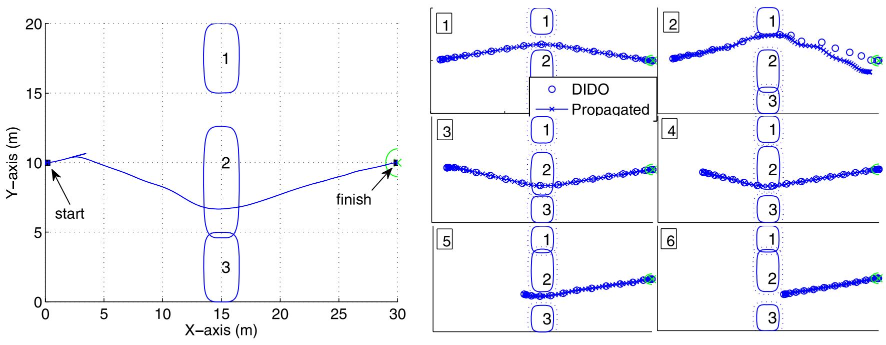


(a) Overall Trajectory (b) Snapshots of Motion

Fig. 11.5 Cyclic sliding door (with prediction; 0.1 m/sec cycle speed)

predicting the position of obstacle 2 as being further north at the time when the vehicle traverses the north passage.

The cycle speed was again raised to 0.2 m/sec to show that with the addition of prediction using course and speed, what was previously a trajectory planning failure became a success, as shown in Fig. 11.6(a). The vehicle initially heads to the north passage. Once obstacle 2 begins moving, the algorithm predicts an infeasibility, because the intercept time of the vehicle with either passage results in complete blockage of both passages (obstacle 2, when expanded to account for vehicle size, completely blocks the path). The result is the vehicle stops, repositions, and reformulates a new bias. In the time it takes the vehicle to stop and reposition, the intercept time changes, and a solution then exists to allow the new bias to be generated through the south passage. The total maneuver time in this case was 40 seconds. The reason for the large increase in maneuver time is the fact that the vehicle stops and repositions. It is obvious that rather than stop and reposition, the vehicle could simply change its speed slightly to affect a different intercept time that would result in an open south passage; however, it is counter to the minimum time formulation of the optimal control problem to slow down (use less control effort) in order to



(a) Overall Trajectory (b) Snapshots of Motion

Fig. 11.6 Cyclic sliding door (with prediction; 0.2 m/sec cycle speed)

achieve a feasible path; therefore, its output indicates an infeasible trajectory using the maximum control effort. Figure 11.6(b) shows the evolution of the vehicle trajectory. Notice the infeasibility in frame 2 of said figure.

The speed of obstacle 2 was further raised in 0.1 m/sec increments to determine where the algorithm would fail when course and speed is used to predict obstacle position. Figure 11.7(a) represents the trajectory corresponding to the fastest cycling speed (0.5 m/sec) that the algorithm could handle. The total maneuver time was 36.3 seconds, which was faster than the case in Fig. 11.6(a), because the vehicle did not have to stop and reposition. Figure 11.7(b) represents the evolution of the vehicle trajectory. Frame 1 shows the trajectory prior to the commencement of obstacle 2 movement. In frames 2 and 3, obstacle 2 is moving north, resulting in the prediction that the trajectory could wrap around the south side of the obstacle as it travels north. In frames 4 and 5, obstacle 2 has changed directions to head south, which results in a spontaneous replanning of the vehicle trajectory to wrap around the north side of the obstacle. In frame 6, obstacle 2 has changed directions again and is headed north, but by that point the vehicle has already safely navigated the north passage. Clearly, if the algorithm can predict obstacle position using course and speed, it is more effective than simply using still snapshots of the environment.

The next logical step here is to show the failure of the algorithm when the speed of obstacle 2 is raised to 0.6 m/sec. Figure 11.8 illustrates that trajectory, which changes direction (by prediction using course and speed) as the direction of obstacle motion changes, but the obstacle motion is too fast for this method to be successful, resulting in collision.

Finally, the cyclic sliding door scenario was repeated for different cycle speeds, but this time more information was known by the algorithm; specifically, the algorithm was given complete a priori knowledge of the motion of obstacle 2. Not only were the instantaneous course and speeds known, but all the future course and speed changes were also known in advance. Knowing the exact future position of obstacle 2 resulted in a trajectory that headed directly for the correct position that gave the vehicle safe passage. No replanning was necessary. Maneuver times at all

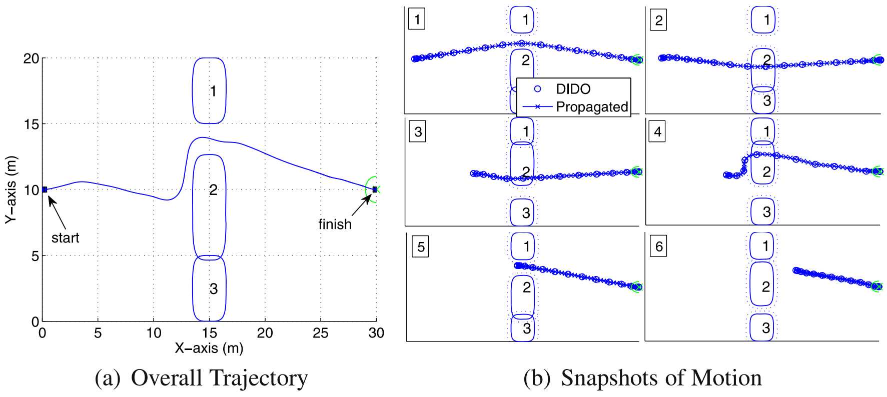


Fig. 11.7 Cyclic sliding door (with prediction; 0.5 m/sec cycle speed)

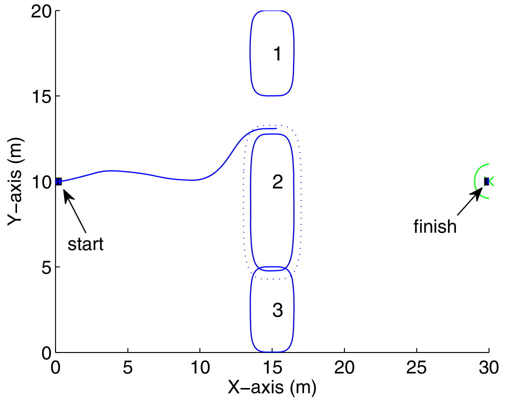


Fig. 11.8 Cyclic sliding door (with prediction; 0.6 m/sec cycle speed)

cycle speeds were the fastest possible (33.4 seconds). Figure 11.9 shows the trajectory for the cycle speed of 0.6 m/sec, above which there can be no solution to the problem due to the physical limit on vehicle speed. It follows that with advance knowledge of environmental changes, the algorithm will always generate feasible and safe trajectories within the physical limits of the vehicle.

Table 11.1 shows a summary of the results of executing the cyclic sliding door scenario at various cycle speeds up to the vehicle's limit using the three different levels of information known to the algorithm. As expected, being able to predict obstacle positions resulted in a higher success rate and better maneuver time than not using prediction; and having a priori knowledge of environmental conditions produced even better results. However, knowing the future positions of obstacles beyond predicting them from their current course and speed is highly unlikely. It's assumed in this work that if an obstacle's course and speed changes can be known in advance, then it is likely another vehicle being controlled by the same home base, which falls under multi-vehicle trajectory planning. For this reason, further motion

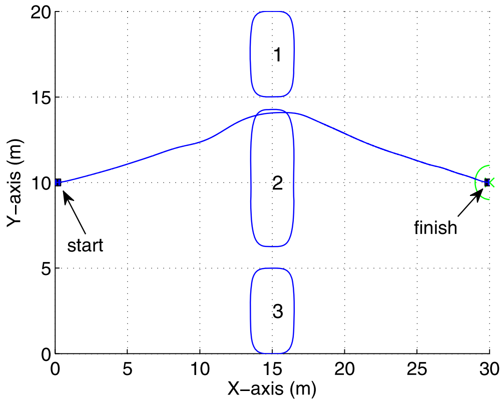


Fig. 11.9 Cyclic sliding door (a priori knowledge; 0.6 m/sec cycle speed)

Table 11.1 Summary of results for cyclic sliding door scenario

Cycle speed (m/sec)	Maneuver time (sec) No prediction	Maneuver time (sec) With prediction	Maneuver time (sec) a priori knowledge
0.1	33.5	33.4	33.4
0.2	collision	40	33.4
0.5	collision	36.3	33.4
0.6	collision	collision	33.4

studies will be limited to the comparisons between using prediction based on course and speed and not using prediction (i.e., still snapshots of the environment).

11.3.3 Obstacle Crossing (No Intercept)

This scenario can be described by any obstacle whose motion results in it crossing over the trajectory of the vehicle, but does not pose a danger of collision. Figure 11.10 shows an example of the expected effect of a crossing obstacle on vehicle trajectory. The obstacle starts out motionless, but after 5 seconds it moves from $(x, y = 20, 5)$ meters to $(x, y = 20, 25)$ meters at the speed of 1.5 m/sec across the vehicle's path as the vehicle travels from $(x, y = 0, 15)$ meters to $(x, y = 30, 15)$ meters. The result shown in Fig. 11.10 was generated using course and speed to predict the obstacle position. The overall maneuver time was 32.3 seconds.

If prediction is not used, the results are significantly worse. The scenario of Fig. 11.10 was repeated without prediction, and the resulting trajectory is shown in Fig. 11.11(a) with a maneuver time that is significantly higher at 49.9 seconds. Figure 11.11(b) shows the trajectory's evolution over time. The figure shows that due to the sensitivity of numerical optimization techniques (DIDO [10] in this case) to the bias, the trajectory is pushed over the crossing obstacle, even though no collision

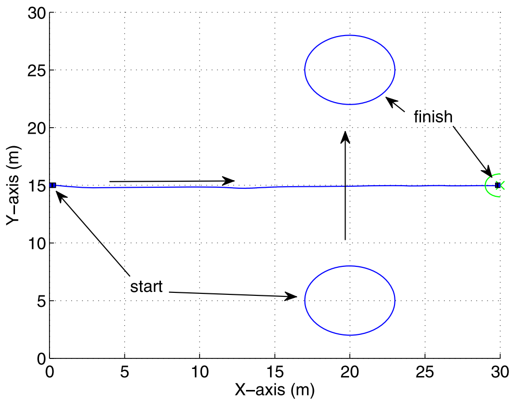


Fig. 11.10 Obstacle crossing vehicle path (with prediction)

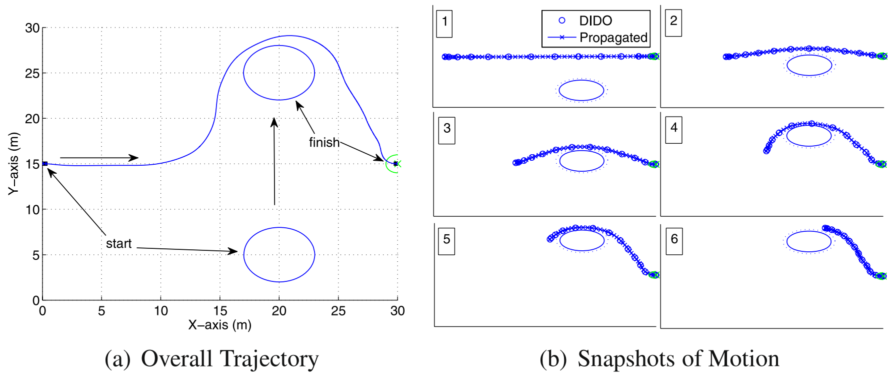
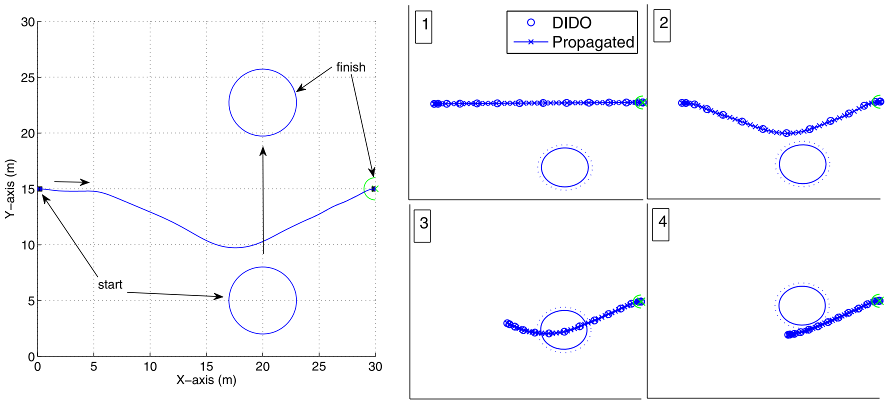


Fig. 11.11 Obstacle crossing vehicle path (no prediction)

course existed. The trajectory essentially gets shaped by any obstacles that cross its path. The same effect can be produced by laying a string on a table whose endpoints correspond to the desired start and goal positions, then sliding objects around the table across the string, resulting in the string wrapping itself around those objects. Adding prediction allows obstacles to skip over the string (trajectory) when encountered. Another situation that would allow for the obstacle to cross without adversely affecting the trajectory is when the obstacle travels fast enough to cause it to skip over the trajectory in successive environment snapshots.

11.3.4 Obstacle Intercept

The same obstacle crossing scenario was used, except the obstacle speed was slowed down to 0.6 m/sec, which put it on a collision course with the vehicle. Figure 11.12(a) shows the trajectory of the vehicle when using prediction, and the maneuver time was 34.5 seconds. The time evolution of the trajectory is shown in



(a) Overall Trajectory (b) Snapshots of Motion

Fig. 11.12 Obstacle intercept (with prediction)

Fig. 11.13 Obstacle intercept (no prediction)

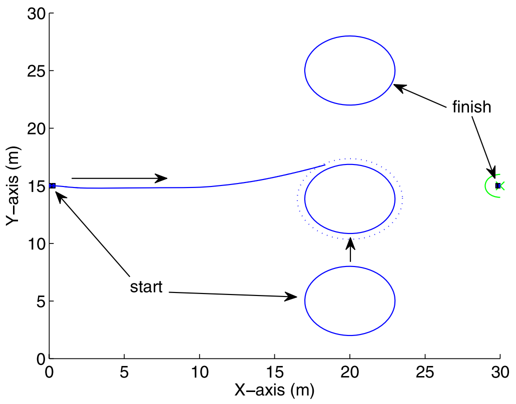


Fig. 11.12(b). For the first 5 seconds, before obstacle motion begins, the vehicle heads straight at the goal. Once the vehicle detects obstacle motion, it estimates the obstacle's course and speed, and autonomously makes a course correction (assuming the obstacle's course and speed stays the same) to avoid collision.

This obstacle intercept scenario was repeated without the use of course and speed to predict obstacle position. The resulting trajectory collided with the obstacle, as shown in Fig. 11.13.

11.3.5 Obstacle Intercept Window

The intercept window is defined, in this work, as a finite period in an obstacle motion in which it is on a collision course with the vehicle, but the obstacle's course and speed is not constant and, therefore, the threat of collision only exists during a

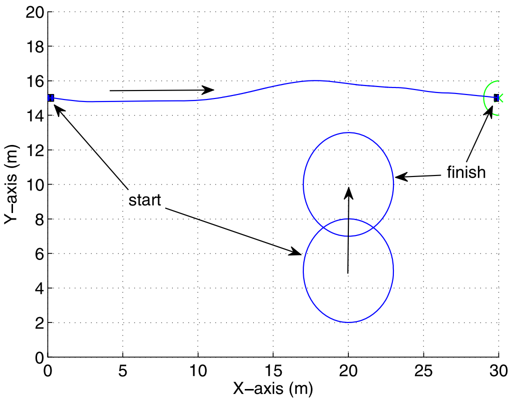
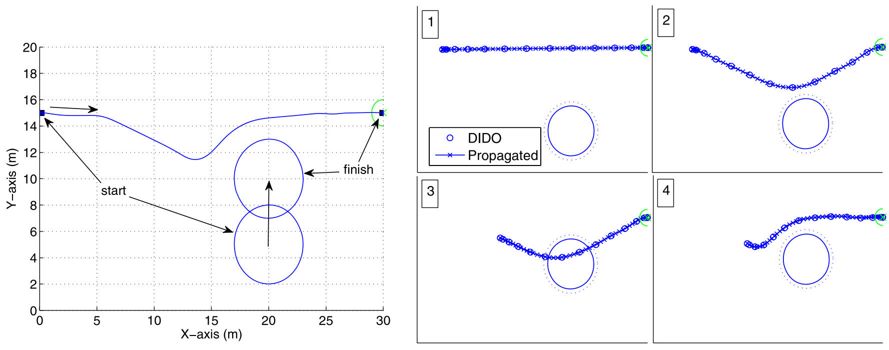


Fig. 11.14 Obstacle intercept window (no prediction)

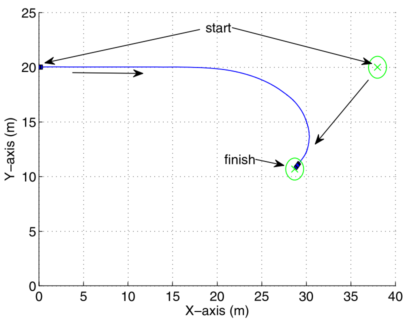


(a) Overall Trajectory (b) Snapshots of Motion

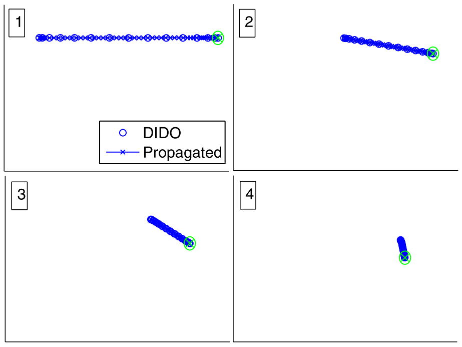
Fig. 11.15 Obstacle intercept window (with prediction)

finite period of time. For example, two cars approaching an intersection have an intercept window when after being on a collision course for a while one car turns onto another road or slows down, thus, eliminating the chance of intercept. The obstacle intercept problem was repeated, but the obstacle's motion was modified to stop at $(x, y) = (20, 10)$ meters, eliminating any possibility of collision. In this scenario the obstacle never obstructs the vehicle's path to the goal. Figure 11.14 shows the vehicle trajectory when prediction is not used. The small alteration in the vehicle's trajectory is due to the effect of the robustness term in the cost function [5]. The overall maneuver time was 32.3 seconds.

The scenario was repeated using prediction, and the resulting trajectory is shown in Fig. 11.15(a). The vehicle was baited off its original trajectory due to the fact that it incorrectly predicted the obstacle's collision course and autonomously replanned to avoid the collision. The maneuver time for this scenario was 33.8 seconds, which is longer than when prediction was not utilized (32.3 seconds). The evolution of the trajectory is shown in Fig. 11.15(b). Frame 1 shows the trajectory before obstacle motion commenced. Frames 2 and 3 show the predicted trajectory once the algo-



(a) Overall Trajectory



(b) Snapshots of Motion

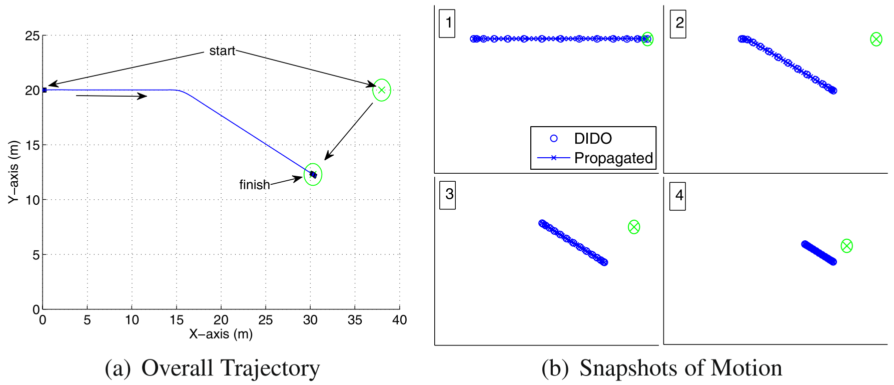
Fig. 11.16 Vehicle faster than target (no prediction)

rithm determined the obstacle's course and speed. Frame 4 shows the new trajectory after the obstacle stopped. Essentially, the obstacle's motion faked out the vehicle into executing an extra maneuver, resulting in it taking a longer time to reach the goal. This intercept window scenario is the one instance when not using prediction actually results in a less maneuver time than when using prediction. However, given the fact that prediction still produced a valid trajectory (albeit a little longer) and the fact that in all other cases prediction is the better way to go, this issue is an acceptable disadvantage.

11.4 Target Motion Studies

11.4.1 Target Rendezvous: Vehicle Faster than Target

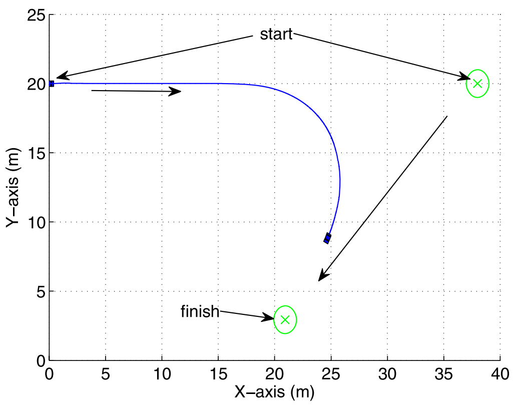
In this scenario, the vehicle must rendezvous with a moving target. Figure 11.16(a) shows the vehicle trajectory without predicting the target position, and Fig. 11.16(b) shows the evolution of that trajectory. Initially, the target is motionless, and after 15 seconds it commences its movement at 0.6 m/sec. Since the algorithm does not use the target's course and speed to predict a rendezvous point, each trajectory planning iteration simply aims the vehicle at the current target position. This results in the vehicle falling in behind the target and catching up to it. The maneuver time was 37.2 seconds. Using the target's course and speed to predict a rendezvous point resulted in the trajectory of Fig. 11.17(a) and a faster maneuver time of 33.3 seconds. Figure 11.17(b) shows the evolution of that trajectory. Notice that the trajectory follows an intercept course based on the prediction of the target future positions.



(a) Overall Trajectory (b) Snapshots of Motion

Fig. 11.17 Vehicle faster than target (with prediction)

Fig. 11.18 Vehicle slower than target (no prediction)



11.4.2 Target Rendezvous: Vehicle Slower than Target

The same target rendezvous scenario was simulated, but this time the target speed was faster than the vehicle's maximum speed limit at 1.2 m/sec. The resulting trajectory for the case of no prediction is shown in Fig. 11.18. The vehicle falls in behind the target and can never catch up. Unless the target slows down, the vehicle can never complete the mission. Figure 11.19 shows the optimal trajectory using target course and speed to predict its future position. In this case, the vehicle was able to rendezvous with the target by calculating an intercept maneuver. It follows that if we want to maximize the vehicle's chances of being able to catch a moving target, then prediction is the way to achieve the best results.

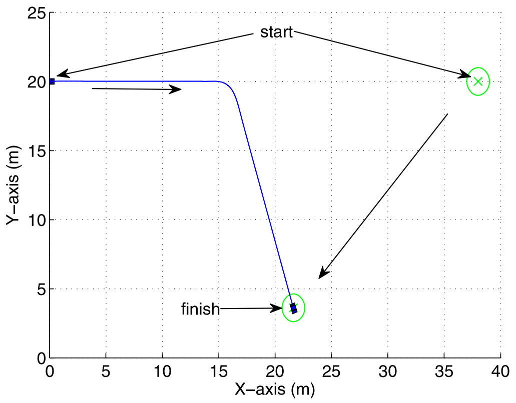


Fig. 11.19 Vehicle slower than target (with prediction)

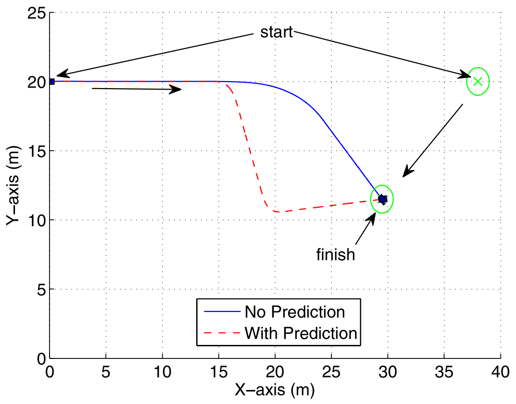


Fig. 11.20 Variable target motion

11.4.3 Target Rendezvous: Variable Target Motion

In this section, we study the effect of changes in the target motion during the vehicle's maneuver. The scenario is the same as in Sect. 11.4.1, except instead of the target moving indefinitely, it stops after 10 seconds of motion. Figure 11.20 shows both cases with and without prediction. The case of no prediction produced a less tortuous maneuver as well as a faster one, posting a 33.9 second maneuver time as opposed to the 36.8 second maneuver time when prediction was used. This scenario shows that in a small number of cases where the obstacle's course and speed varies frequently and significantly, not using prediction can actually produce better results than when using prediction. However, as stated previously, the price paid is acceptable when weighing the benefits of using prediction.

11.5 Conclusion

An information-oriented pseudospectral optimal control framework provides a natural approach for autonomous trajectory planning. Simulation results illustrate the effects of varying levels of information and how more or less information influences the performance of the planner. The simulations and discussions in this paper explain how the trajectory planning framework developed in [5] is an information centric planner that, with some exceptions, produces better results with more information. Predicting obstacle and target positions using course and speed data produced faster (with a few exceptions) and more reliable trajectories than using still snapshots of the environment. Having a priori knowledge of environmental conditions will always produce the best results, however, providing such a level of information is not always practical. In summary, when conducting trajectory planning, it is desirable to include as much information in the optimal control problem formulation as possible.

References

1. Bollino, K.P., Lewis, L.R.: Optimal path planning and control of tactical unmanned aerial vehicles in urban environments. In: Proceedings of the AUVSI's Unmanned Systems North America 2007 Conference, Washington, DC (2007)
2. Bollino, K.P., Ross, I.M., Doman, D.: Optimal nonlinear feedback guidance for reentry vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO, AIAA-2006-6074 (2006)
3. Bollino, K.P., Lewis, L.R., Sekhavat, P., Ross, I.M.: Pseudospectral optimal control: a clear road for autonomous intelligent path planning. In: AIAA Infotech@Aerospace 2007 Conference and Exhibit, Rohnert Park, CA, AIAA-2007-2831 (2007)
4. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion; Theory, Algorithms, and Implementation. MIT Press, Cambridge (2005)
5. Hurni, M.A., Sekhavat, P., Ross, I.M.: Autonomous trajectory planning using real-time information updates. In: Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Honolulu, Hawaii, AIAA-2008-6305 (2008)
6. Lewis, L.R.: Rapid motion planning and autonomous obstacle avoidance for unmanned vehicles. Master's Thesis, Naval Postgraduate School, Monterey, CA (2006)
7. Lewis, L.R., Ross, I.M.: A pseudospectral method for real-time motion planning and obstacle avoidance. In: AVT-SCI Joint Symposium on Platform Innovations and System Integration for Unmanned Air, Land and Sea Vehicles, Florence, Italy (2007)
8. Lewis, L.R., Ross, I.M., Gong, Q.: Pseudospectral motion planning techniques for autonomous obstacle avoidance. In: Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, pp. 5997–6002 (2007)
9. Ross, I.M.: Lecture Notes in Control and Optimization. Naval Postgraduate School, Monterey (2007)
10. Ross, I.M.: A beginner's guide to DIDO: a MATLAB application package for solving optimal control problems. Elissar Technical Report TR-711, http://www.elissar.biz (2007)

11. Ross, I.M., Sekhavat, P., Fleming, A., Gong, Q.: Optimal feedback control: foundations, examples, and experimental results for a new approach. J. Guid. Control Dyn. **31**(2), 307–321 (2008)
12. Sekhavat, P., Fleming, A., Ross, I.M.: Time-optimal nonlinear feedback control for the NPSAT1 spacecraft. In: Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Monterey, CA, pp. 843–850 (2005)
13. Siegwart, R., Nourbakhsh, I.R.: Introduction to Autonomous Mobile Robots. MIT Press, Cambridge (2004)

Chapter 12
Orbital Evasive Target Tracking and Sensor Management

**Huimin Chen, Genshe Chen, Dan Shen,
Erik P. Blasch, and Khanh Pham**

Summary In this chapter, we consider the sensor management problem for tracking space targets where the targets may apply evasive maneuvering strategy to avoid being tracked by the space borne observers. We first study the case of single target tracking by a single observer and formulate the pursuit–evasion game with complete information. Then we extend the tracking problem to a set of collaborative observers and each observer has to decide when to sense which target in order to achieve the desired estimation error covariance. A popularly used criterion for sensor management is to maximize the total information gain in the observer-to-target assignment. We compare the information based approach to the game theoretic criterion where the observers are assigned according to the best response of the terminal result in the pursuit–evasion game. Finally, we use realistic satellite orbits to simulate the space resource management for situation awareness. We adopted NASA's General Mission Analysis Tool (GMAT) for space target tracking with multiple space borne observers. The results indicate that the game theoretic approach is more effective than the information based approach in handling intelligent target maneuvers.

12.1 Introduction

Over recent decades, the space environment has become more complex with a significant increase in space debris among densely populated satellites. Efficient and

H. Chen
Department of Electrical Engineering, University of New Orleans, New Orleans, LA, USA

G. Chen (✉) · D. Shen
DCM Research Resources, LLC, Germantown, MD, USA
e-mail: gchen@dcmresearchresources.com

E.P. Blasch
AFRL/RYAA, WPAFB, OH, USA

K. Pham
AFRL/RVSV, Kirtland AFB, NM, USA

reliable space operations rely heavily on the space situation awareness where search and tracking space targets and identifying their intent are crucial in creating a consistent global picture of the space. Orbit determination with measurements provided by a constellation of satellites has been studied extensively [6, 7, 14, 16]. The tracking and data relay satellite system uses satellites in geostationary orbits (GEO) to track the satellites in low-Earth orbit (LEO). The global positioning system (GPS) uses a constellation of satellites with pseudo-range measurements, i.e., range measurements with clock differentials to determine the location of a user. Unlike ground targets whose motion may contain frequent maneuvers, a satellite usually follows its orbit so that long-term prediction of its orbital trajectory is possible once the orbital elements are known [6]. However, a space target can also make an orbital change owing to its desired mission or intentionally hiding from the space borne observers. Existing maneuvering target tracking literature mainly focuses on modeling target maneuver motion at random onset time (see, e.g., [1, 13]). In space surveillance, the constellation of satellite observers is usually known to the adversary and very unlikely to change frequently due to energy constraint. In this case, evasive maneuvering motion can be intelligently designed to take advantage of the sensing geometry, e.g., transferring to an orbit with maximum duration of the Earth blockage to an observer with known orbital trajectory. Accordingly, a sensor management method has to optimally utilize the sensing resources to acquire and track space targets with sparse measurements, i.e., with a typically large sampling interval, in order to maintain a large number of tracks simultaneously.

Sensor management is concerned with the sensor-to-target assignment and a schedule of sensing actions for each sensor in the near future given the currently available information on the space targets. Sensor assignment and scheduling usually aim to optimize a certain criterion under energy, data processing and communication constraints. One popularly used criterion is the total information gain for all the targets being tracked [11]. However, this criterion does not prioritize the targets with respect to their types or identities. Alternatively, covariance control optimizes the sensing resources to achieve the desired estimation error covariance for each target [10]. It has the flexibility to design the desired tracking accuracy according to the importance of each target. We want to compare both informative based criterion and the covariance control method in a game theoretic setting where the target can intelligently choose its maneuvering motion and onset time based on the knowledge of observers' constellation.

The rest of the chapter is organized as follows. Section 12.2 presents the space target motion and sensor measurement model. Section 12.3 provides a game theoretic formulation of target maneuvering motion. Section 12.4 discusses the information based performance metric for sensor management and covariance control method with possibly evasive target maneuvering motion. Section 12.5 compares the performance of the proposed tracking and sensor management scheme with the existing methods. Conclusions and future work are presented in Sect. 12.6.

12.2 Fundamentals of Space Target Orbits

12.2.1 Time and Coordinate Systems

Several time systems are popularly used in the orbit determination problems. Satellite laser ranging measurements are usually time-tagged in coordinated universal time (UTC) while global positioning system (GPS) measurements are time tagged in GPS system time (GPS-ST). Although both UTC and GPS-ST are based on atomic time standards, UTC is loosely tied to the rotation of the Earth through the application of "leap seconds" while GPS-ST is continuous with the relation GPS-ST = UTC + n where n is the number of leap seconds since January 6, 1980. The orbital equation describing near-Earth satellite motion is typically tagged with terrestrial dynamical time (TDT). It is an abstract, uniform time scale implicitly defined by the motion equation and can be converted to UTC or GPS-ST for any given reference date.

The Earth centered inertial (ECI) coordinate system used to link GPS-ST with UTC is a geocentric system defined by the mean equator and vernal equinox at Julian epoch 2000.0. Its XY-plane coincides with the equatorial plane of the Earth and the X-axis points toward the vernal equinox direction. The Z-axis points toward the north pole and the Y-axis completes the right hand coordinate systems.

The Earth centered Earth fixed (ECEF) coordinate system has the same XY-plane and the Z-axis as in the inertial coordinate system. However, its X-axis rotates with the Earth and points to the prime meridian and the Y-axis completes the right hand coordinate systems.

The local Cartesian system commonly referred to as east-north-up (ENU) coordinate system has its origin at some point on the Earth surface or above (typically at the location of an observer). Its Z-axis is normal to the Earth's reference ellipsoid defined by the geodetic latitude. The X-axis points toward the east while the Y-axis points toward the north. The conversion among these three coordinate systems is provided in Appendix 1.

12.2.2 Orbital Equation and Orbital Parameter Estimation

Without any perturbing force, the position $\mathbf{r}$ of a space target relative to the center of the Earth in ECI coordinate system should satisfy

$$\ddot{\mathbf{r}} = -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} \tag{12.1}$$

where μ is the Earth's gravitational parameter. The target velocity is $\mathbf{v} \triangleq \dot{\mathbf{r}}$ and the radial velocity is $v_r \triangleq \frac{\mathbf{v}\cdot\mathbf{r}}{r}$ where $r \triangleq \|\mathbf{r}\|$ is the distance from the target to the center of the Earth. In order to determine the position and velocity of a satellite at any time instance, six parameters are needed, typically, the three position components and three velocity components at a certain time instance. Alternatively, the orbital trajectory can be conveniently described by the six components of the Keplerian elements. The description of Keplerian elements and their relationship to the kinematic state of the target can be found in Appendix 2.

In reality, a number of forces act on the satellite in addition to the Earth's gravity. To distinguish them from the central force created by the satellite target, these forces are often referred to as perturbing forces. In a continuous time state space model, perturbing forces are often lumped into the noise term of the system dynamics. Denote by $\mathbf{x}(t)$ the continuous time target state given by

$$\mathbf{x}(t) \triangleq \begin{bmatrix} \mathbf{r}(t) \\ \dot{\mathbf{r}}(t) \end{bmatrix} = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ v_x(t) \\ v_y(t) \\ v_z(t) \end{bmatrix} \tag{12.2}$$

For convenience, we omit the argument t and write the nonlinear state equation as follows.

$$\dot{\mathbf{x}} = f(\mathbf{x}) + \mathbf{w} \tag{12.3}$$

where

$$f(\mathbf{x}) = \begin{bmatrix} v_x \\ v_y \\ v_z \\ -(\mu/r^3)x \\ -(\mu/r^3)y \\ -(\mu/r^3)z \end{bmatrix} \tag{12.4}$$

and

$$\mathbf{w} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ w_x \\ w_y \\ w_z \end{bmatrix} \tag{12.5}$$

is the acceleration resulting from perturbing forces. As opposed to treating the perturbing acceleration as noise, spacecraft general propagation (SGP) model maintains general perturbation element sets and finds analytical solution to the satellite motion equation with time varying Keplerian elements [12]. For precise orbit determination, numerical integration of (12.3) is often a viable solution where both the epoch state and the force model have to be periodically updated when a new measurement is available [17].

12.3 Modeling Maneuvering Target Motion in Space Target Tracking

12.3.1 Sensor Measurement Model

We consider the case that a space satellite in low-Earth orbit (LEO) observes a target in geostationary orbit (GEO). A radar onboard the space satellite can provide the following type of measurements: range, azimuth, elevation, and range rate. The range between the ith observer located at (x_i, y_i, z_i) and the space target located at (x, y, z) is given by

$$d_r(i) = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} \tag{12.6}$$

The azimuth is

$$d_a(i) = \tan^{-1}\left(\frac{y - y_i}{x - x_i}\right) \tag{12.7}$$

The elevation is

$$d_e(i) = \tan^{-1}\left(\frac{z - z_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}}\right) \tag{12.8}$$

The range rate is

$$d_{\dot{r}}(i) = \frac{(x - x_i)(\dot{x} - \dot{x}_i) + (y - y_i)(\dot{y} - \dot{y}_i) + (z - z_i)(\dot{z} - \dot{z}_i)}{d_r} \tag{12.9}$$

Measurements from the ith observer will be unavailable when the line-of-sight path between the observer and the target is blocked by the Earth. Thus, the constellation of multiple observers is important to maintain consistent coverage of the target of interest.

The condition of Earth blockage is examined as follows. If there exist $\alpha \in [0, 1]$ such that $D_\alpha(i) < R_E$, where

$$D_\alpha(i) = \sqrt{\left[(1-\alpha)x_i + \alpha x\right]^2 + \left[(1-\alpha)y_i + \alpha y\right]^2 + \left[(1-\alpha)z_i + \alpha z\right]^2} \tag{12.10}$$

then the measurement from the ith observer to the target will be unavailable. The minimum of $D_\alpha(i)$ is achieved at $\alpha = \alpha^*$ given by

$$\alpha^* = -\frac{x_i(x - x_i) + y_i(y - y_i) + z_i(z - z_i)}{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} \tag{12.11}$$

Thus, we first examine whether $\alpha^* \in [0, 1]$ and then check the Earth blockage condition $D_{\alpha^*}(i) < R_E$.

12.3.2 Game Theoretic Formulation for Target Maneuvering Onset Time

We consider the case that a single observer tracks a single space target. Initially, the observer knows the target's state and the target also knows the observer's state. Assume that the target can only apply a T second burn that produces a specific thrust $\mathbf{w}$ with a maximum acceleration of a m/s^2. The goal of the target is to determine the maneuvering onset time and the direction of the thrust so that the resulting orbit will have the maximum duration of the Earth blockage to the observer. The goal of the observer is to maintain the target track with the highest estimation accuracy. To achieve this, the observer has to determine the sensor revisit time and cuing region as well as notify other observers having better geometry when Earth blockage occurs. Without loss of generality, we assume that the target can transfer its orbit to the same plane as the observer. In this case, when the target is at the opposite side of the Earth with respect to the observer and rotating in the same direction as the observer, the duration of the Earth blockage will be the maximum compared with other orbits with the same orbital elements except the inclinations. Note that in the pursuit–evasion game confined to a two dimensional plane, the minimax solution requires that the target applies the same thrust angle as the observer's (see Appendix 4 for details). Thus, an intelligent target will choose its maneuvering onset time as soon as its predicted observer's orbital trajectory has the Earth blockage. The corresponding maneuvering thrust will follow the minimax solution to the pursuit–evasion game.

When a target is tracked by multiple space borne observers, an observer can predict the target's maneuvering motion based on its estimated target state and the corresponding response of the pursuit–evasion game from the target where the terminal condition will lead to the Earth blockage to the observer. Thus, there is a need for the sensor manager to select the appropriate set of sensors that can persistently monitor all the targets especially when they maneuver.

12.3.3 Nonlinear Filter Design for Space Target Tracking

When a space target has been detected, the filter will predict the target state at any time instance in the future based on the available sensor measurements. Denote by $\hat{\mathbf{x}}_k^-$ the state prediction from time t_{k-1} to time t_k based on the state estimate $\hat{\mathbf{x}}_{k-1}^+$ at time t_{k-1} with all measurements up to t_{k-1}. The prediction is made by numerically integrating the state equation given by

$$\dot{\hat{\mathbf{x}}}(t) = f\left(\hat{\mathbf{x}}(t)\right) \tag{12.12}$$

without process noise. The mean square error (MSE) of the state prediction is obtained by numerically integrating the following matrix equation

$$\dot{P}(t) = F\left(\hat{\mathbf{x}}_k^-\right)P(t) + P(t)F\left(\hat{\mathbf{x}}_k^-\right)^T + Q(t) \tag{12.13}$$

where $F(\hat{\mathbf{x}}_k^-)$ is the Jacobian matrix given by

$$F(\mathbf{x}) = \begin{bmatrix} 0_{3\times3} & I_3 \\ F_0(\mathbf{x}) & 0_{3\times3} \end{bmatrix} \tag{12.14}$$

$$F_0(\mathbf{x}) = \mu \begin{bmatrix} \frac{3x^2}{r^5} - \frac{1}{r^3} & \frac{3xy}{r^5} & \frac{3xz}{r^5} \\ \frac{3xy}{r^5} & \frac{3y^2}{r^5} - \frac{1}{r^3} & \frac{3yz}{r^5} \\ \frac{3xz}{r^5} & \frac{3yz}{r^5} & \frac{3z^2}{r^5} - \frac{1}{r^3} \end{bmatrix} \tag{12.15}$$

$$r = \sqrt{x^2 + y^2 + z^2} \tag{12.16}$$

and evaluated at $\mathbf{x} = \hat{\mathbf{x}}_k^-$. The measurement $\mathbf{z}_k$ obtained at time t_k is given by

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{12.17}$$

where

$$\mathbf{v}_k \sim \mathcal{N}(0, R_k) \tag{12.18}$$

is the measurement noise, which is assumed independent of each other and independent to the initial state as well as process noise.

The recursive linear minimum mean square error (LMMSE) filter applies the following update equation [2]

$$\hat{\mathbf{x}}_{k|k} \overset{\Delta}{=} E^*\big[\mathbf{x}_k|\mathbf{Z}^k\big] = \hat{\mathbf{x}}_{k|k-1} + K_k\tilde{\mathbf{z}}_{k|k-1} \tag{12.19}$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k' \tag{12.20}$$

where

$$\begin{aligned}
\hat{\mathbf{x}}_{k|k-1} &= E^*\big[\mathbf{x}_k|\mathbf{Z}^{k-1}\big] \\
\hat{\mathbf{z}}_{k|k-1} &= E^*\big[\mathbf{z}_k|\mathbf{Z}^{k-1}\big] \\
\tilde{\mathbf{x}}_{k|k-1} &= \mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} \\
\tilde{\mathbf{z}}_{k|k-1} &= \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1} \\
P_{k|k-1} &= E\big[\tilde{\mathbf{x}}_{k|k-1}\tilde{\mathbf{x}}_{k|k-1}'\big] \\
S_k &= E\big[\tilde{\mathbf{z}}_{k|k-1}\tilde{\mathbf{z}}_{k|k-1}'\big] \\
K_k &= C_{\tilde{\mathbf{x}}_k\tilde{\mathbf{z}}_k} S_k^{-1} \\
C_{\tilde{\mathbf{x}}_k\tilde{\mathbf{z}}_k} &= E\big[\tilde{\mathbf{x}}_{k|k-1}\tilde{\mathbf{z}}_{k|k-1}'\big]
\end{aligned}$$

Note that $E^*[\cdot]$ becomes the conditional mean of the state for linear Gaussian dynamics and the above filtering equations become the celebrated Kalman filter [2]. For nonlinear dynamic system, (12.19) is optimal in the mean square error sense when the state estimate is constrained to be an affine function of the measurement.

Given the state estimate $\hat{\mathbf{x}}_{k-1|k-1}$ and its error covariance $P_{k-1|k-1}$ at time t_{k-1}, if the state prediction $\hat{\mathbf{x}}_{k|k-1}$, the corresponding error covariance $P_{k|k-1}$, the measurement prediction $\hat{\mathbf{z}}_{k|k-1}$, the corresponding error covariance S_k, and the crosscovariance $E[\tilde{\mathbf{x}}_{k|k-1}\tilde{\mathbf{z}}'_{k|k-1}]$ in (12.19) and (12.20) can be expressed as a function only through $\hat{\mathbf{x}}_{k-1|k-1}$ and $P_{k-1|k-1}$, then the above formula is truly recursive. However, for general nonlinear system dynamics (12.3) and measurement equation (12.17) , we have

$$\hat{\mathbf{x}}_{k|k-1} = E^*\Big[\int_{t_{k-1}}^{t_k} f\big(\mathbf{x}(t), \mathbf{w}(t)\big)\,dt + \mathbf{x}_{k-1}|\mathbf{Z}^{k-1}\Big] \tag{12.21}$$

$$\hat{\mathbf{z}}_{k|k-1} = E^*\big[h(\mathbf{x}_k, \mathbf{v}_k)|\mathbf{Z}^{k-1}\big] \tag{12.22}$$

Both $\hat{\mathbf{x}}_{k|k-1}$ and $\hat{\mathbf{z}}_{k|k-1}$ will depend on the measurement history $\mathbf{Z}^{k-1}$ and the corresponding moments in the LMMSE formula. In order to have a truly recursive filter, the required terms at time t_k can be obtained *approximately* through $\hat{\mathbf{x}}_{k-1|k-1}$ and $P_{k-1|k-1}$, i.e.,

$$\{\hat{\mathbf{x}}_{k|k-1}, P_{k|k-1}\} \approx \mathrm{Pred}\big[f(\cdot), \hat{\mathbf{x}}_{k-1|k-1}, P_{k-1|k-1}\big]$$
$$\{\hat{\mathbf{z}}_{k|k-1}, S_k, C_{\tilde{\mathbf{x}}_k\tilde{\mathbf{z}}_k}\} \approx \mathrm{Pred}\big[h(\cdot), \hat{\mathbf{x}}_{k|k-1}, P_{k|k-1}\big]$$

where $\mathrm{Pred}[f(\cdot), \hat{\mathbf{x}}_{k-1|k-1}, P_{k-1|k-1}]$ denotes that $\{\hat{\mathbf{x}}_{k-1|k-1}, P_{k-1|k-1}\}$ propagates through the nonlinear function $f(\cdot)$ to approximate $E^*[f(\cdot)|\mathbf{Z}^{k-1}]$ and the corresponding error covariance $P_{k|k-1}$.

Similarly, $\mathrm{Pred}[h(\cdot), \hat{\mathbf{x}}_{k|k-1}, P_{k|k-1}]$ predicts the measurement and the corresponding error covariance only through the approximated state prediction. This poses difficulties for the implementation of the recursive LMMSE filter due to insufficient information. The prediction of a random variable going through a nonlinear function, most often, can not be completely determined using only the first and second moments. Two remedies are often used: One is to approximate the system to the best extent such that the prediction based on the approximated system can be carried out only through $\{\hat{\mathbf{x}}_{k-1|k-1}, P_{k-1|k-1}\}$ [20]. Another is by approximating the density function with a set of particles and propagating those particles in the recursive Bayesian filtering framework, i.e., using a particle filter [8].

12.3.4 Posterior Cramer–Rao Lower Bound of the State Estimation Error

Denote by $J(t)$ the Fisher information matrix. Then the posterior Cramer–Rao lower bound (PCRLB) is given by [18]

$$B(t) = J(t)^{-1} \tag{12.23}$$

which quantifies the ideal mean square error of any filtering algorithm, i.e.,

$$E\left[\left(\hat{\mathbf{x}}(t_k)-\mathbf{x}(t_k)\right)\left(\hat{\mathbf{x}}(t_k)-\mathbf{x}(t_k)\right)^T|\mathbf{Z}^k\right]\geq B(t_k) \tag{12.24}$$

Assuming an additive white Gaussian process noise model, the Fisher information matrix satisfies the following differential equation

$$\dot{J}(t)=-J(t)F(\mathbf{x})-F(\mathbf{x})^TJ(t)-J(t)Q(t)J(t) \tag{12.25}$$

for $t_{k-1}\leq t\leq t_k$ where F is the Jacobian matrix given by

$$F(\mathbf{x})=\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \tag{12.26}$$

When a measurement is obtained at time t_k with additive Gaussian noise $\mathcal{N}(0, R_k)$, the new Fisher information matrix is

$$J(t_k^+)=J(t_k^-)+E_{\mathbf{x}}\left[H(\mathbf{x})^TR_k^{-1}H(\mathbf{x})\right] \tag{12.27}$$

where H is the Jacobian matrix given by

$$H(\mathbf{x})=\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} \tag{12.28}$$

See Appendix 3 for the numerical procedure to evaluate the Jacobian matrix for a non-perturbed orbital trajectory propagation. The initial condition for the recursion is $J(t_0)$ and the PCRLB can be obtained with respect to the true distribution of the state $\mathbf{x}(t)$. In practice, the sensor manager will use the estimated target state to compute the PCRLB for any time instance of interest and decide whether a new measurement has to be made to improve the estimation accuracy.

12.4 Sensor Management for Situation Awareness

12.4.1 Information Theoretic Measure for Sensor Assignment

In sensor management, each observer has to decide when to measure which target so that the performance gain in terms of a certain metric can be maximized. For a Kalman filter or its extension for the nonlinear dynamic state or measurement equations, namely, the recursive LMMSE filter, the error covariance of the state estimate has the following recursive form [2].

$$P_{k+1|k+1}^{-1}=P_{k+1|k}^{-1}+H(\mathbf{x}_{k+1})^TR_{k+1}^{-1}H(\mathbf{x}_{k+1}) \tag{12.29}$$

Thus the information gain from the sensor measurement at time t_{k+1} in terms of the inverse of the state estimation error covariance is $H(\mathbf{x}_{k+1})^TR_{k+1}^{-1}H(\mathbf{x}_{k+1})$ where R_{k+1} is the measurement error covariance. Consider M observers each of which can measure at most one target at any sampling time. When there are N space targets being tracked by M observers, sensor assignment is concerned with the sensor to target correspondence so that the total information gain can be maximized. Denote by χ_{ij} the assignment of observer i to target j at any particular time t_{k+1}. The sensor assignment problem is

$$\min_{\chi_{ij}} c_{ij}\chi_{ij} \tag{12.30}$$

$$\text{subject to} \quad \sum_{i=1}^{M} \chi_{ij} \le 1, \quad j = 1, \ldots, N; \tag{12.31}$$

$$\sum_{j=1}^{N} \chi_{ij} \le 1, \quad i = 1, \ldots, M; \tag{12.32}$$

and $\chi_{ij} \in \{0, 1\}$. The cost c_{ij} is

$$c_{ij} = \mathrm{Tr}\big\{H\big(\mathbf{x}_j\big((t_{k+1})\big)\big)^T R_i^{-1}(t_{k+1}) H_i\big(\mathbf{x}_j\big((t_{k+1})\big)\big)\big\} \tag{12.33}$$

if there is no Earth blockage between observer i to target j. In the above formulation, all targets are assumed to have the same importance so the observes are scheduled to make the most informative measurements. This may lead to a greedy solution where those targets close to the observers will be tracked more accurately than those away from the observers. It may not achieve the desired tracking accuracy for each target.

12.4.2 Covariance Control for Sensor Scheduling

Covariance control method does not optimize a performance metric directly in the sensor-to-target assignment, instead, it requires the filter designer to specify a desired state estimation error covariance so that the selection of sensors and the corresponding sensing times will meet the specified requirements after the tracker update using the sensor measurements as scheduled. Denote by $P_d(t_{k+1})$ the desired error covariance of the state estimation at time t_{k+1}. Then the need of a sensor measurement at t_{k+1} for this target, according to the covariance control method, is

$$n(t_{k+1}) = -\min\big(\mathrm{eig}\big\{P_d(t_{k+1}) - P_{k+1|k}\big\}\big) \tag{12.34}$$

where the negative sign has the following implication: A positive value of the eigenvalue difference implies that the desired covariance requirement is not met. The goal of covariance control is to minimize the total sensing cost so that all the desired covariance requirements are met. If we use the same notation c_{ij} as the cost for the observer i to sense the target j at t_{k+1}, then the covariance control tries to solve the following optimization problem.

$$\min_{\chi_{ij}} c_{ij}\chi_{ij} \tag{12.35}$$

$$\text{subject to} \quad n_{ij}(t_{k+1}) = -\min\big(\text{eig}\big\{P_{dij}(t_{k+1}) - P_{k+1|k+1}(\chi_{ij})\big\}\big) \le 0,$$
$$i = 1, \ldots, M, \; j = 1, \ldots, N \tag{12.36}$$

and $\chi_{ij} \in \{0, 1\}$. In this setting, more than one observer can sense the same target at the same time. The optimization problem is combinatorial in nature and one has to evaluate 2^{MN} possible sensor-to-target combinations in general, which is computationally prohibitive. Alternatively, a suboptimal need-based greedy algorithm has been proposed [10]. It also considers the case where certain constraints can not be met, i.e., the desired covariance is unachievable even with all the sensing resources.

12.4.3 Game Theoretic Covariance Prediction for Sensor Management

In the formulation of the sensor management problem, we need the filter to provide the information on the state estimation error covariance which is based on the orbital trajectory propagation assuming that the target does not maneuver. If the target maneuvers, then the filter calculated error covariance assuming the non-maneuver motion model will be too optimistic. There are two possible approaches to account for the target maneuver motion. One is to detect target maneuver and estimate its onset time as quickly as possible [15]. Then the filter will be adjusted with larger process noise covariance to account for the target maneuvering motion. Alternatively, one can design a few typical target maneuvering motion models and run a multiple model estimator with both non-maneuver and maneuver motion models [2]. The multiple model filter will provide the model conditioned state estimation error covariances as well as the unconditional error covariance for sensor management purposes. Note that the multiple model estimator does not make a hard decision on which target motion model is in effect at any particular time, but evaluates the probability of each model. The corresponding unconditional covariance immediately after target maneuver onset time can still be very optimistic, which is needed to support the evidence that a maneuvering motion model is more likely than a non-maneuvering one. As a consequence, the scheduled sensing action in response to the target maneuver based on the unconditional covariance from a multiple model estimator can be too late for evasive target motion.

We propose to use generalized Page's test for detecting target maneuver [15] and apply the *model conditioned* error covariance from each filter in the sensor management. Denoted by $S_m(t_{k+1})$ the set of targets being classified as in the maneuvering mode and $S_{-m}(t_{k+1})$ the set of targets in the nonmaneuvering model, respectively. We apply covariance control for sensor-to-target allocation only to those targets in $S_m(t_{k+1})$ and use the remaining sensing resources to those targets in $S_{-m}(t_{k+1})$ by maximizing the information gain. The optimization problem becomes

$$\min_{\chi_{ij}} c_{ij}\chi_{ij} \tag{12.37}$$

$$\text{subject to} \quad n_{ij}(t_{k+1}) = -\min\big(\text{eig}\big\{P_{dij}(t_{k+1}) - P_{k+1|k+1}(\chi_{ij})\big\}\big) \le 0,$$

$$i = 1, \ldots, M, \; j \in S_m(t_{k+1}) \tag{12.38}$$

$$\sum_{i=1}^{M} \chi_{ij} \le 1, \quad j \in S_{-m}(t_{k+1}) \tag{12.39}$$

$$\sum_{j \in S_{-m}(t_{k+1})} \chi_{ij} \le 1, \quad i = 1, \ldots, M \tag{12.40}$$

and $\chi_{ij} \in \{0, 1\}$. The cost c_{ij} is

$$c_{ij} = \begin{cases} \text{Tr}\big\{H\big(\mathbf{x}_j\big((t_{k+1})\big)\big)^T R_i^{-1}(t_{k+1}) H_i\big(\mathbf{x}_j\big((t_{k+1})\big)\big)\big\} & j \in S_{-m}(t_{k+1}) \\ 0 & j \in S_m(t_{k+1}) \end{cases} \tag{12.41}$$

Given that target $j \in S_{-m}(t_k)$, we assume that observer i will declare $j \in S_m(t_{k+1})$ if the predicted target location has Earth blockage to the observer i at t_{k+1}. If the target is declared as in the maneuvering mode, then the predicted error covariance will be based on the worst case scenario of the pursuit–evasion game between the observer and the target (see Appendix 4 for details).

12.5 Simulation Study

12.5.1 Scenario Description

We consider a small scale space target tracking scenario where four satellite observers collaboratively track two satellite targets. The nominal orbital trajectories are generated from realistic satellite targets selected in the SpaceTrack database. The four observer satellites are (1) ARIANE 44L, (2) OPS 856, (3) VANGUARD 1, and (4) ECHO 1. They are in low-Earth orbits. The two target satellites are: (1) ECHOSTAR 10, and (2) COSMOS 2350. They are in geostationary orbits. The simulation was based on the software package that utilizes the general mission analysis tool (GMAT).[1] Figure 12.1 shows the orbital trajectories of the observers and targets. The associated tracking errors were obtained based on the recursive linear minimum mean square error filter when sensors are assigned to targets according to the nonmaneuvering motion. The error increases in some time segments are due to the Earth blockage where no measurements are available from any observer.

[1]General Mission Analysis Tool, released by National Aeronautics and Space Administration (NASA), available at http://gmat.gsfc.nasa.gov/.

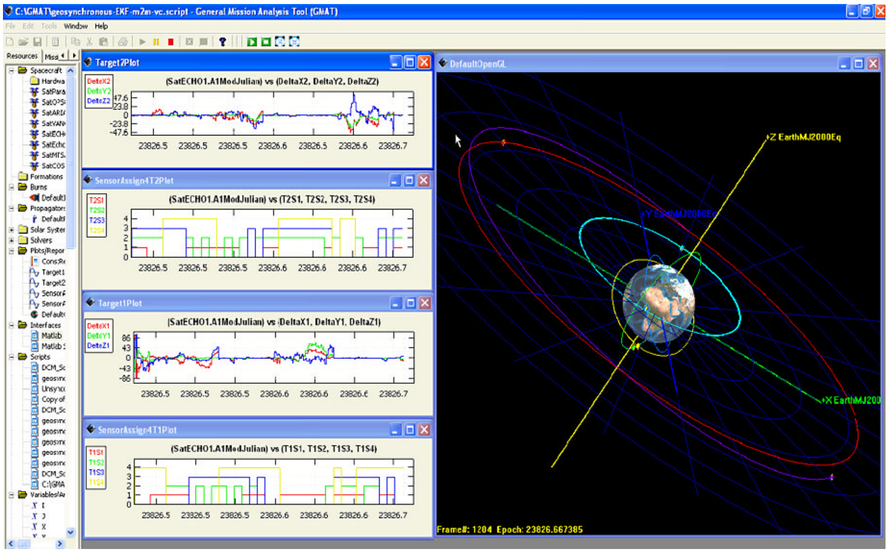

C:\GMAT\geosynchronous-EKF-m2m-vc.script - General Mission Analysis Tool (GMAT)
Target2Plot
(SatECHO1.A1ModJulian) vs (DeltaX2, DeltaY2, DeltaZ2)
SensorAssign4T2Plot
(SatECHO1.A1ModJulian) vs (T2S1, T2S2, T2S3, T2S4)
Target1Plot
(SatECHO1.A1ModJulian) vs (DeltaX1, DeltaY1, DeltaZ1)
SensorAssign4T1Plot
(SatECHO1.A1ModJulian) vs (T1S1, T1S2, T1S3, T1S4)
DefaultOpenGL
+Z EarthMJ2000Eq
+Y EarthMJ2000Eq
+X EarthMJ200
Frame#: 1204 Epoch: 23826.667385

Fig. 12.1 The space target tracking scenario where four observers collaboratively track two targets

12.5.2 Performance Comparison

We now consider the case in which the target performs an unknown thrust maneuver that changes the eccentricity of its orbit. In particular, both targets are initially in the GEO orbit and at time $t = 1000$ s, target 1 performs a 1 s burn that produces a specific thrust, i.e., an acceleration $w = [0\ 0.3\ 0]^T$ km/s^2, while, at time $t = 1500$ s, target 2 performs a 1 s burn that produces a specific thrust $w = [0\ 0.5\ 0]^T$ km/s^2. The eccentricity change of target 1 after the burn is around $e \approx 0.35$ while the eccentricity change of target 2 after the second burn is $e \approx 0.59$. Another type of target maneuver is the inclination change produced by a specific thrust. Two inclination changes are generated with $i \approx 0.16$ for target 1 and $i \approx 0.09$ for target 2. Each observer has the minimal sampling interval of 50 s and we assume that all observers are synchronized and the sensor manager can make centralized coordination based on the centralized estimator for each target. We consider two cases: (i) the observer has range and angle measurements with standard deviations 0.1 km and 10 mrad, respectively, and (ii) the observer has range, angle and range rate measurements with standard deviations 0.1 km, 10 mrad, 2 m/s, respectively.

We applied the generalized Page's test (GPT) without and with range rate measurement for maneuver detection while the filter update of state estimate does not use the range rate measurement [15]. The reason is that the nonlinear filter designed assuming non-maneuver target motion is sensitive to the model mismatch in the range rate when target maneuvers. The thresholds of the generalized Page's test for both case (i) and case (ii) were chosen to have the false alarm probability $P_{FA} = 1\%$. The average delays of target maneuver onset detection for both cases are measured in terms of the average number of observations from the maneuver onset time to

Table 12.1 Comparison of tracking accuracy for various orbital changes of the targets

Cases	$e \to 0.35$	$e \to 0.59$	$i \to 0.09$	$i \to 0.16$
(i) Average delay per observations	7.3	6.4	12.1	9.6
(i) PTR, peak position error (km)	33.7	54.8	14.5	18.9
(i) IMM, peak position error (km)	48.1	66.5	12.4	22.3
(i) PTR, peak velocity error (km/s)	0.38	0.40	0.22	0.25
(i) IMM, peak velocity error (km/s)	0.41	0.44	0.21	0.26
(ii) Average delay per observations	1.3	1.0	2.9	2.4
(ii) PTR, peak position error (km)	6.9	7.4	3.1	3.8
(ii) IMM, peak position error (km)	8.3	11.2	3.2	4.1
(ii) PTR, peak velocity error (km/s)	0.28	0.31	0.16	0.19
(ii) IMM, peak velocity error (km/s)	0.30	0.34	0.16	0.21

the declaration of the target maneuver. Once target maneuver is declared, then another filter assuming the white noise acceleration with process noise spectrum of 0.6 km/s^2 was used along both tangential and normal directions of the estimated target motion. Alternatively, an interacting multiple model (IMM) estimator [2] with nonmaneuver and maneuver motion models using the same parameter settings as the model switching filters embedded in the target maneuvering detector was used to compare the tracking accuracy. Table 12.1 compares the peak errors in position and velocity for each target maneuvering motion using model switching filter and the IMM estimator. The average detection delays for both cases are also shown in Table 12.1 for GPT algorithm. We can see that the range rate measurement, if available, can improve the average detection delay of target maneuver significantly and thus reducing the peak estimation errors in both target position and velocity. The model switching filter using GPT has better tracking accuracy than the IMM estimator in most cases even for the peak errors. It should be clear that the filter based on nonmaneuver motion model outperforms the IMM estimator during the segment that the target does not have an orbital change. Interestingly, even though the inclination change takes longer time to detect compared with the eccentricity change for both targets, the resulting peak estimation errors in position and velocity are relatively smaller for both the model switching filter and the IMM estimator. The extensive comparison among other nonlinear filtering methods for space target tracking can be found in [4].

Next, we assume that both target 1 and target 2 will choose their maneuver onset times intelligently based on their geometries to the observers. Both targets can have a maximum acceleration of 0.05 km/s^2 with a maximum of 10 s burn. We assume that each observer can measure target range, angle, and range rate with the same accuracy as in the case (ii) of the tracking scenario considered previously. We implemented the following configurations of the sensor management schemes to determine which observer measures which target at a certain time instance. (i) Information based method: Sensors are allocated with a uniform sampling interval of 50 s to maximize the information gain. (ii) Covariance control based method: Sen-

Table 12.2 Performance comparison for various sensor management methods

Configuration	Peak position error (km)	Average position error (km)	Peak velocity error (km/s)	Average velocity error (km/s)	Average sampling interval (s)
(i) Target 1	89.7	24.4	2.1	0.16	50
(i) Target 2	76.3	14.2	1.6	0.14	50
(ii) Target 1	16.7	2.4	0.36	0.10	13.5
(ii) Target 2	15.2	1.8	0.29	0.09	16.8
(iii) Target 1	12.2	1.8	0.28	0.08	37.6
(iii) Target 2	10.9	1.3	0.27	0.08	39.2

sors are scheduled with the sampling interval such that the desired position error is within 10 km for each target. The state prediction error covariance for nonmaneuver or maneuver motion model is computed based on the posterior Cramer–Rao lower bound. (iii) Game theoretic method: Sensors are allocated by maximizing the information gain for nonmaneuvering targets and covariance control will be applied to maneuvering targets. The maneuvering onset time is predicted based on the pursuit evasion game for each observer-to-target pairing. In all three configurations, model switching filter is used for tracking each target. We compare the peak and average errors in position and velocity for both targets as well as the average sampling interval from all observers for configurations (i)–(iii). The results are listed in Table 12.2. We can see that the information based method (configuration (i)) yields the largest peak errors among the three schemes. This is due to the fact that both targets apply evasive maneuver so that the peak errors will be much larger compared with the results in Table 12.1 based on the same tracker design and sampling interval. In order to achieve the desired position error, covariance control method (configuration (ii)) achieves much smaller peak and average errors compared with configuration (i) at the price of making the sampling interval much shorter. Note that the peak position errors are larger than 10 km for both targets owing to the detection delay of maneuvering onset time. The proposed game theoretic method (configuration (iii)) has the smallest peak and average errors because of the prediction of maneuvering onset time by modeling target evasive motion from the pursuit evasion game. Note that it also has longer average sampling interval than that of configuration (ii) due to a quicker transient when each target stops its burn, indicating possible energy saving for the overall system. The sensing resources saved with configuration (iii) can also be applied to search other potential targets in some designated cells. One possible approach to perform joint search and tracking of space targets was discussed in [5].

12.6 Summary and Conclusions

In space target tracking by satellite observers, it is crucial to assign the appropriate sensor set to each target and minimizes the Earth blockage period. With complete knowledge of the space borne observers, a target may engage its evasive maneuvering motion immediately after the Earth blockage occurs and change its orbit to maximize the duration of the Earth blockage. We presented a game theoretic model for the determination of maneuvering onset time and consequently, the covariance control is applied to the maneuvering targets in the sensor-to-target assignment. For nonmaneuvering targets, we try to maximize the total information gain by selecting sensors that will provide the most informative measurements on the target's state. We simulated a multi-observer multi-target tracking scenario where four LEO observers collaboratively track two GEO targets. We found that the multiple model estimator assuming random maneuvering onset time yields much larger estimation error compared with the model switching tracker based on maneuvering detection from the solution to the pursuit–evasion game. In addition, sensor assignment based on maximum information gain can lead to large tracking error for evasive targets while using the same desired error covariance for all targets can only alleviate the issue at the price of more frequent revisit time for each target. Fortunately, the sensor assignment based on covariance control for maneuvering targets and maximum information gain for nonmaneuvering targets achieves a reasonable tradeoff between the tracking accuracy and the consumption of sensing resources.

There are many avenues to extend the existing work in order to achieve space situational awareness. First, target intent can be inferred based on its orbital history. It is of great interest to separate the nonevasive and evasive orbital maneuvering motions and allocate the sensing resources accordingly. Second, our model of the pursuit–evasion game relies on the complete knowledge of the observer's and target's state which may not be known to both players in real life. This poses challenges in the development of the game theoretic model with incomplete information which is computationally tractable for the sensor management to allocate sensing resources ahead of time. Finally, the current nonlinear filter does not consider the clutter and closely spaced targets where imperfect data association has to be handled by the filtering algorithm. It should be noted that the posterior Cramer–Rao lower bound for single target tracking with random clutter and imperfect detection [19] can be readily applied to the covariance control. However, it is still an open research problem to design efficient nonlinear filtering method that can achieve the theoretical bound of the estimation error covariance.

Acknowledgement This work was supported in part by the US Air Force under contracts FA8650-09-M-1552 and FA9453-09-C-0175, US Army Research Office under contract W911NF-08-1-0409, Louisiana Board of Regents NSF(2009)-PFUND-162, and the Office of Research and Sponsored Programs, the University of New Orleans.

Appendix 1: Conversion of the Coordinate Systems

The following conversion schemes among different coordinate systems are based on [3]. Given the position $\mathbf{r} = [\xi \;\; \eta \;\; \zeta]'$ in the ECEF frame, the latitude φ, longitude λ and altitude h (which are the three components of $\mathbf{r}_{geo}$) are determined by

```
STEP 1  $\varphi = 0$
        repeat
          $\varphi_{\text{old}} = \varphi$
          $D_\varphi = R_e\left[1 - \epsilon_e \sin^2 \varphi_{\text{old}}\right]^{-\frac{1}{2}}$
          $\varphi = \text{atan}\left(\frac{\zeta + D_\varphi \epsilon_e^2 \sin \varphi_{\text{old}}}{\sqrt{\xi^2 + \eta^2}}\right)$
        until $|\varphi - \varphi_{\text{old}}| < TOL$
STEP 2  $\lambda = \text{atan}(\frac{\eta}{\xi})$
        $h = \frac{\zeta}{\sin \varphi} - D_\varphi\left(1 - \epsilon_e^2\right)$
```

where $R_e = 6378.137$ km and $\epsilon_e = 0.0818191$ are the equatorial radius and eccentricity of the Earth, respectively. *TOL* is the error tolerance (e.g., 10^{-10}) and the convergence occurs normally within 10 iterations.

The origin of the local Cartesian frame $\mathbf{O}$ is given by

$$\mathbf{O} = \begin{bmatrix} 0 & D_\varphi \epsilon_e^2 \sin \varphi \cos \varphi & D_\varphi\left(\epsilon_e^2 \sin^2 \varphi - 1\right) \end{bmatrix}' \tag{12.42}$$

and the rotation matrix is given by

$$A = \begin{bmatrix} -\sin \lambda & \cos \lambda & 0 \\ -\sin \varphi \cos \lambda & -\sin \varphi \sin \lambda & \cos \varphi \\ \cos \varphi \cos \lambda & \cos \varphi \sin \lambda & \sin \varphi \end{bmatrix} \tag{12.43}$$

The position $\mathbf{r}$ in the local Cartesian frame is given by

$$\mathbf{r}_{\text{loc}} = A\mathbf{r} + \mathbf{O} \tag{12.44}$$

Appendix 2: Keplerian Elements

The *specific angular momentum* lies normal to the orbital plane given by $\mathbf{h} = \mathbf{r} \times \mathbf{v}$ with magnitude $h \triangleq \|\mathbf{h}\|$. *Inclination* is the angle between the equatorial plane and the orbital plane given by $i \triangleq \cos^{-1}(\frac{h_z}{h})$ where h_z is the z-component of $\mathbf{h}$. *Eccentricity* of the orbit is given by

$$\mathbf{e} \triangleq \frac{1}{\mu}\left[\left(v^2 - \frac{\mu}{r}\right)\mathbf{r} - r v_r \mathbf{v}\right] \tag{12.45}$$

with magnitude $e \triangleq \|\mathbf{e}\|$. The longitude of the ascending node is given by

$$\Omega \triangleq \begin{cases} \cos^{-1}(\frac{n_x}{n}) & n_y \geq 0 \\ 2\pi - \cos^{-1}(\frac{n_x}{n}) & n_y < 0 \end{cases} \tag{12.46}$$

where $\mathbf{n}$ is the vector pointing towards the ascending node with magnitude $n \overset{\Delta}{=} \|\mathbf{n}\|$. The *argument of perigee* is angle between the node line and the eccentricity vector given by

$$\omega \overset{\Delta}{=} \begin{cases} \cos^{-1}(\frac{\mathbf{ne}}{ne}) & e_z > 0 \\ 2\pi - \cos^{-1}(\frac{\mathbf{ne}}{ne}) & e_z < 0 \end{cases} \tag{12.47}$$

with the convention that

$$\omega = \cos^{-1}\left(\frac{e_x}{e}\right) \tag{12.48}$$

for an equatorial orbit. The *true anomaly* ν is the angle between the eccentricity vector and the target's position vector given by

$$\nu \overset{\Delta}{=} \begin{cases} \cos^{-1}(\frac{\mathbf{er}}{er}) & v_r > 0 \\ 2\pi - \cos^{-1}(\frac{\mathbf{er}}{er}) & v_r < 0 \end{cases} \tag{12.49}$$

with the convention that $\nu = \cos^{-1}(\frac{r_x}{r})$ for a circular orbit. The *eccentric anomaly* is the angle between apogee and the current position of the target given by

$$E = \cos^{-1}\left(\frac{1 - r/a}{e}\right) \tag{12.50}$$

where a is the orbit's semi-major axis given by $a = \frac{1}{\frac{2}{r} - \frac{v^2}{\mu}}$. The *mean anomaly* is $M = E - e \sin E$. The orbital period is given by $T = 2\pi\sqrt{\frac{a^3}{\mu}}$.

The six Keplerian elements are $\{a, i, \Omega, \omega, e, M\}$. The orbit of a space target can be fully determined by the parameter set $\{i, \Omega, \omega, T, e, M\}$ with initial condition given by the target position at any particular time [17]. The angles $\{i, \Omega, \omega\}$ transform the inertial frame to the orbital frame while T and e specify the size and shape of the ellipsoidal orbit. The time dependent parameter $\nu(t)$ represents the position of the target along its orbit in the polar coordinate system.

Appendix 3: Algorithm for Orbital State Propagation

Presented below is an algorithm that propagates the state of an object in an orbital trajectory around the Earth following [3]. Both the trajectory propagation and the corresponding Jacobian matrix of the nonlinear orbital equation are given. Let $x(t)' = [\mathbf{r}(t)' \ \dot{\mathbf{r}}(t)']$ be the unknown state to be computed at time t, given the state $x_0' = \mathbf{x}(t_0)' = [\mathbf{r}_0' \dot{\mathbf{r}}_0']$ at the time t_0. The gravitational parameter $\mu = 3.986012 \times 10^5$ km^3/sec^2 and the convergence check parameter $TOL = 10^{-10}$ are used.

```
STEP 1   $r_0 := \|\mathbf{r}_0\|; \quad v_0 := \|\dot{\mathbf{r}}_0\|; \quad q_0 := \frac{1}{\mu}\mathbf{r}_0'\dot{\mathbf{r}}_0$
         $a_0 := \frac{2}{r_0} - \frac{v_0^2}{\mu}; \quad p_0 := \frac{1 - a_0 r_0}{\sqrt{\mu}}$
STEP 2   $\alpha := \frac{a_0(t - t_0)}{\sqrt{\mu}}; \quad \beta := a_0\alpha^2$
STEP 3   $c := \frac{1 - \cos(\sqrt{\beta})}{\beta}; \quad s := \frac{\sqrt{\beta} - \sin(\sqrt{\beta})}{\beta\sqrt{\beta}}$
STEP 4   $\tau := p_0\alpha^3 s + q_0\alpha^2 c + \frac{r_0}{\sqrt{\mu}}\alpha$
         $\frac{\mathrm{d}\tau}{\mathrm{d}\alpha} := p_0\alpha^2 c + q_0\alpha(1 - s\beta) + \frac{r_0}{\sqrt{\mu}}$
         $\alpha := \alpha + \left[\frac{\mathrm{d}\tau}{\mathrm{d}\alpha}\right]^{-1}\left[(t - t_0) - \tau\right]$
STEP 5   if$\left(\left[(t - t_0) - \tau\right] > TOL\right)$
         gotoSTEP 3
STEP 6   $f := 1 - \frac{\alpha^2 c}{r_0}; \quad g := (t - t_0) - \frac{\alpha^3 s}{\sqrt{\mu}}$
         $\mathbf{r}(t) := f\mathbf{r}_0 + g\dot{\mathbf{r}}_0; \quad r := \left\|\mathbf{r}(t)\right\|$
STEP 7   $\dot{f} := \left(\frac{\sqrt{\mu}}{r_0}\right)(s\beta - 1)\left(\frac{\alpha}{r}\right); \quad \dot{g} := 1 - \frac{\alpha^2 c}{r}; \quad \dot{\mathbf{r}}(t) := \dot{f}\mathbf{r}_0 + \dot{g}\dot{\mathbf{r}}_0$
```

The above steps yield the required state $x(t)' = [\mathbf{r}(t)'\dot{\mathbf{r}}(t)']$ at time t. In order to predict the covariance of the position $\mathbf{r}(t)$ by propagating the covariance of $\mathbf{r}(t_0)$ from t_0 to t, we need to compute the 6×3 matrix $\nabla_{x_0}\mathbf{r}(t)$. The computation of this matrix involves the following additional steps.

```
STEP 8   $\nabla_{x_0} r_0 := \begin{bmatrix} \mathbf{r}_0 \\ 0 \end{bmatrix}\left(\frac{1}{r_0}\right); \quad \nabla_{x_0} v_0 := \begin{bmatrix} 0 \\ \dot{\mathbf{r}}_0 \end{bmatrix}\left(\frac{1}{v_0}\right)$
         $\nabla_{x_0} q_0 := \begin{bmatrix} \dot{\mathbf{r}}_0 \\ \mathbf{r}_0 \end{bmatrix}\left(\frac{1}{\mu}\right)$
         $\nabla_{x_0} a_0 := (\nabla_{x_0} r_0)\left(\frac{-2}{r_0^2}\right) + (\nabla_{x_0} v_0)\left(\frac{-2v_0}{\mu}\right)$
         $\nabla_{x_0} p_0 := (\nabla_{x_0} r_0)\left(\frac{-a_0}{\sqrt{\mu}}\right) + (\nabla_{x_0} a_0)\left(\frac{-r_0}{\sqrt{\mu}}\right)$
```

```
STEP 9    $\frac{\mathrm{d}s}{\mathrm{d}\beta} := \frac{c-3s}{2\beta}; \quad \frac{\mathrm{d}c}{\mathrm{d}\beta} := \frac{1-s\beta-2c}{2\beta}$
STEP 10   $b_1 := (\nabla_{x_0} q_0)\left(-\alpha^2 c\right) + (\nabla_{x_0} p_0)\left(-\alpha^3 s\right) + (\nabla_{x_0} r_0)\left(\frac{-\alpha}{\sqrt{\mu}}\right)$
          $b_2 := (\nabla_{x_0} a_0)\left(-\alpha^2\right)$
          $A := \begin{bmatrix} 3p_0\alpha^2 s + 2q_0\alpha c + \frac{r_0}{\sqrt{\mu}} & p_0\alpha^3 \frac{\mathrm{d}s}{\mathrm{d}\beta} + q_0\alpha^2 \frac{\mathrm{d}c}{\mathrm{d}\beta} \\ 2a_0\alpha & -1 \end{bmatrix}$
          $\left[(\nabla_{x_0}\alpha)(\nabla_{x_0}\beta)\right] := [b_1 b_2]A^{-1}$
STEP 11   $\nabla_{x_0} f := \left[(\nabla_{x_0} r_0)\left(\frac{\alpha c}{r_0}\right) - (\nabla_{x_0}\alpha)(2c) - (\nabla_{x_0}\beta)\left(\alpha\frac{\mathrm{d}c}{\mathrm{d}\beta}\right)\right]\left[\frac{\alpha}{r_0}\right]$
          $\nabla_{x_0} g := \left[(\nabla_{x_0}\alpha)(3s) - (\nabla_{x_0}\beta)\left(\alpha\frac{\mathrm{d}s}{\mathrm{d}\beta}\right)\right]\left[\frac{-\alpha^2}{\sqrt{\mu}}\right]$
STEP 12   $\nabla_{x_0}\mathbf{r}(t) = \begin{bmatrix} fI_3 \\ gI_3 \end{bmatrix} + (\nabla_{x_0} f)\mathbf{r}_0' + (\nabla_{x_0} g)\dot{\mathbf{r}}_0'.$
```

Appendix 4: Pursuit Evasion Game in a 2D Plane

Consider the space target orbiting the Earth with the polar coordinate system fixed on the Earth's center. The motion of the target is given by

$$\ddot{r} - r\dot{\theta}^2 = -\frac{\mu}{r^2} + \frac{F\sin\alpha}{m} \tag{12.51}$$

$$r\ddot{\theta} + 2\dot{r}\dot{\theta} = \frac{F\cos\alpha}{m} \tag{12.52}$$

where α is the angle the thrust vector and the local horizontal as shown in Fig. 12.2. Denote by v_θ and v_r the tangential and radial velocities of the target, respectively. The dynamic equation of the space target can be written as

$$\dot{v}_r - \frac{v_\theta^2}{r} = -\frac{\mu}{r^2} + \frac{F\sin\alpha}{m} \tag{12.53}$$

$$\dot{v}_\theta + \frac{v_r v_\theta}{r} = \frac{F\cos\alpha}{m} \tag{12.54}$$

It is desirable to normalize the parameters with respect to a reference circular orbit with radius r_0 and velocity v_0, so that significant figures will not be lost due to linearizing the target motion equation. Define the following normalized state variables.

$$x_1 = \frac{r}{r_0} \tag{12.55}$$

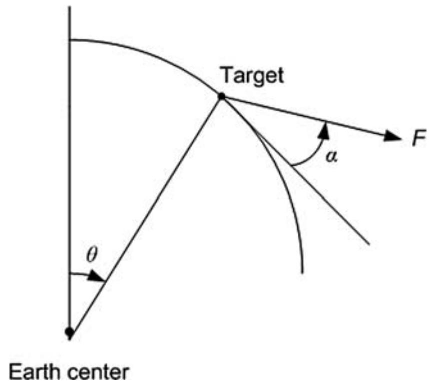


Fig. 12.2 Simplified 2D geometry of a space target

$$x_2 = \frac{v_r}{v_0} \tag{12.56}$$

$$x_3 = \frac{v_\theta}{v_0} \tag{12.57}$$

$$x_4 = \theta \tag{12.58}$$

Let $\tau = \frac{v_0}{r_0} t$. The state equation with respect to τ can be written as

$$\dot{x}_1 = x_2 \tag{12.59}$$

$$\dot{x}_2 = \frac{x_3^2}{x_1} - \frac{1}{x_1^2} + F_0 \sin\alpha \tag{12.60}$$

where $F_0 = \frac{r_0 F}{v_0^2 m}$ is a constant depending the thrust F and target's mass m.

$$\dot{x}_3 = -\frac{x_2 x_3}{x_1} + F_0 \cos\alpha \tag{12.61}$$

$$\dot{x}_4 = \frac{x_3}{x_1} \tag{12.62}$$

In the standard pursuit evasion game, the objective is to find the minimax solution, if exists, to the objective function

$$J = \phi\big(\mathbf{x}(t_f)\big) \tag{12.63}$$

with the state dynamics given by

$$\dot{\mathbf{x}} = f(\mathbf{x}, u, v, t) \tag{12.64}$$

and the initial condition $\mathbf{x}(t_0) = \mathbf{x}_0$ as well as the terminal constraint $\psi(\mathbf{x}(t_f)) = 0$. Here u and v represent the controls associated with the pursuer and the evader, respectively. The goal is to determine $\{u^*, v^*\}$ such that

$$J\big(u^*, v\big) \le J\big(u^*, v^*\big) \le J\big(u, v^*\big) \tag{12.65}$$

The necessary condition for the minimax solution to exist is that the costate $\boldsymbol{\lambda}$ satisfies the following transversality condition.

$$\boldsymbol{\lambda} = \phi_{\mathbf{x}}(t_f) + \nu\psi_{\mathbf{x}}(t_f) \tag{12.66}$$

$$H(t_f) = \phi_t(t_f) + \nu\psi_t(t_f) \tag{12.67}$$

where ν is a Lagrange multiplier and H is the Hamiltonian associated with $\boldsymbol{\lambda}$ that has to be optimized

$$H^* = \max_v \min_u H(\mathbf{x}, \boldsymbol{\lambda}, u, v, t) \tag{12.68}$$

It has been shown in [9] that at the optimal solution, the thrust angles of the pursuer and the evader are the same.

References

1. Bar-Shalom, Y., Blair, W.D.: Multitarget-Multisensor Tracking: Applications and Advances, vol. III. Artech House, Norwood (2000)
2. Bar-Shalom, Y., Li, X.R., Kirubarajan, T.: Estimation with Applications to Tracking and Navigation: Algorithms and Software for Information Extraction. Wiley, New York (2001)
3. Bate, R., et al.: Fundamentals of Astrodynamics. Dover, New York (1971)
4. Chen, H., Chen, G., Blasch, E., Pham, K.: Comparison of several space target tracking filters. In: Proc. SPIE Defense, Security Sensing, vol. 7730, Orlando, FL, USA (2009)
5. Chen, G., Chen, H., Pham, K., Blasch, E., Cruz, J.B.: Awareness-based game theoretic space resource management. In: Proc. SPIE Defense, Security Sensing, vol. 7730, Orlando, FL, USA (2009)
6. Duong, N., Winn, C.B.: Orbit determination by range-only data. J. Spacecr. Rockets **10**, 132–136 (1973)
7. Fowler, J.L., Lee, J.S.: Extended Kalman filter in a dynamic spherical coordinate system for space based satellite tracking. In: Proc. AIAA 23rd Aerospace Sciences Meeting. AIAA-85-0289, Reno, NV (1985)
8. Gordon, N., Salmond, D., Smith, A.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEE Proc. F **140**, 107–113 (1993)
9. Isaacs, R.: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization. Wiley, New York (1965)
10. Kalandros, M., Pao, L.Y.: Covariance control for multisensor systems. IEEE Trans. Aerosp. Electron. Syst. **38**, 1138–1157 (2002)
11. Kreucher, C.M., Hero, A.O., Kastella, K.D., Morelande, M.R.: An information based approach to sensor management in large dynamic networks. IEEE Proc. **95**, 978–999 (2007)
12. Lane, M.H., Hoots, F.R.: General perturbations theories derived from the 1965 lane drag theory. Project Space Track Report No. 2, Aerospace Defense Command, Peterson AFB, CO (1979)
13. Li, X.R., Jilkov, V.P.: Survey of maneuvering target tracking. Part V, multiple-model methods. IEEE Trans. Aerosp. Electron. Syst. **41**, 1255–1321 (2005)
14. Pisacane, V.L., Mcconahy, R.J., Pryor, L.L., Whisnant, J.M., Black, H.D.: Orbit determination from passive range observations. IEEE Trans. Aerosp. Electron. Syst. **10**, 487–491 (1974)
15. Ru, J., Chen, H., Li, X.R., Chen, G.: A range rate based detection technique for tracking a maneuvering target. In: Proc. of SPIE Conf. Signal and Data Processing of Small Targets. San Diego, CA, USA (2005)

16. Teixeira, B.O.S., Santillo, M.A., Erwin, R.S., Bernstein, D.S.: Spacecraft tracking using sampled-data Kalman filters—an illustrative application of extended and unscented estimators. IEEE Control Syst. Mag. **28**(4), 78–94 (2008)
17. Vallado, D.A.: Fundamentals of Astrodynamics and Applications, 2nd edn. Microcosm Press, El Segundo (2001)
18. Van Trees, H.L.: Detection, Estimation, and Modulation Theory, Part I. Wiley, New York (1968)
19. Van Trees, H.L., Bell, K.L.: Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking. Wiley-Interscience, New York (2007)
20. Zhao, Z., Li, X.R., Jilkov, V.P.: Best linear unbiased filtering with nonlinear measurements for target tracking. IEEE Trans. Aerosp. Electron. Syst. **40**, 1324–1336 (2004)

Chapter 13
Decentralized Cooperative Control of Autonomous Surface Vehicles

Pedro DeLima, Dimitri Zarzhitsky, and Daniel Pack

Summary Many pressing issues associated with control of cooperative intelligent systems involve challenges that arise from the difficulty of coordinating multiple task objectives in highly dynamic, unstructured environments. This chapter presents a multi-objective cooperative control methodology for a team of autonomous surface vehicles deployed with the purpose of protecting a waterway against hostile intruders. The methodology captures the intent of a human commander by breaking down high-level mission objectives into specific task assignments for a fleet of autonomous boats with a suite of on-board sensors, limited processing units, and short-range communication capabilities. The fundamental technologies supporting our control method have already been field-tested on a team of autonomous aerial vehicles, and the aim of this work is to extend the previously developed theories to multiple problem domains using heterogeneous vehicle platforms.

13.1 Introduction

Over the course of the past several years, the research team from the Sensor-based Intelligent Robotics Laboratory at the U.S. Air Force Academy has been working on a comprehensive and principled approach to the design of decentralized mobile sensor networks [5]. A significant amount of this ongoing effort is presently dedicated to the development of a scalable control algorithm that can coordinate activities of mixed robot-human teams on a variety of battlefield scenarios. The emphasis of the current work is on the management of the cooperative behaviors of autonomous surface vehicles (ASVs) through a decentralized decision-making mechanism based on

P. DeLima (✉) · D. Zarzhitsky · D. Pack
Department of Electrical and Computer Engineering, U.S. Air Force Academy, USAFA, CO 80840, USA
e-mail: pedro.lima@usafa.edu

D. Zarzhitsky
e-mail: dimitri.zarzhitsky@usafa.edu

D. Pack
e-mail: daniel.pack@usafa.edu

a periodic analysis of current mission objectives and the overall mission progress. The on-board vehicle controller is based on a multi-tiered architecture, and accommodates multiple short- and long-term objectives using a consistent mathematical formulation. The decentralized approach is necessary for any desired scalable system with minimal communication requirements among cooperative vehicles [10]. This chapter provides a more in-depth description of this novel control framework.

13.2 Motivation

To better focus our discussion on task-level vehicle control, we assume an existence of a low-level drive mechanism responsible for actuating control surfaces of an ASV, so that the vehicle can navigate toward a specified waypoint with the help of reactive obstacle avoidance technique when necessary [4]. We also assume the existence of a general path-planning module capable of charting optimal routes (on a coarse scale) from one location to another, e.g., [1]. To help with the evaluation of our methodology for designing cooperative control algorithms, we plan to deploy several ASVs tasked with a sequence of multi-objective missions [9], such as transit from one port to another, search for threats within a specific area, protect stationary and mobile assets, as well as investigate, pursuit, and engage hostile targets.

In order to realize our solution in an efficient manner, we should rely as much as possible on the intelligence and autonomy of our robotic platform, and our control algorithm must address two important challenges. First, a decision logic that allows a cooperative vehicle to determine its current operating mode and associated sub-tasks for each operating mode must be formulated. This decision logic must incorporate inputs from a participating human observer, in addition to the information and requests sent by other autonomous vehicles. The vehicle operating *modes* are general goals, such as surveillance, transit, pursuit, and engagement. Within the context of each mode, a vehicle can perform different sub-tasks depending on the environmental circumstances, human intervention, and interaction with other vehicles. Second, given a specific mode of operation and a particular task selected within it, a vehicle must compute, in real time, a sequence of waypoints that will guide it in a manner appropriate to each situation. Although in the envisioned decentralized control approach each vehicle must determine its own path independently, the ultimate collective, cooperative behaviors rely significantly on the coordination among vehicles in order to improve the group's combined performance. Likewise, knowledge of the on-board payload and essential characteristics of cooperating vehicles, as well as timely communication is essential for the creation of such trajectories.

13.3 Decentralized Hierarchical Supervisor

In this section, we present a novel decentralized hierarchical supervisor for the ASV task allocation technique during missions with multiple objectives. The supervisor

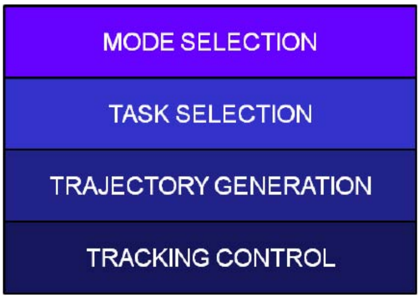


Fig. 13.1 The four layers of the decentralized hierarchical supervisor

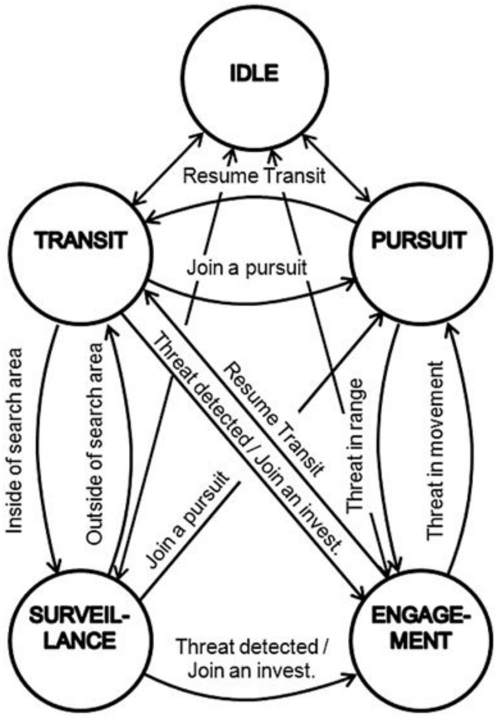


Fig. 13.2 Decision logic for mode selection

module is composed of four layers (shown in Fig. 13.1). The top layer contains the decision logic for making the mode selection, which is shown in Fig. 13.2. The *mode selection* decision logic is responsible for evaluating the conditions necessary for making a transition between the five principal modes of operation: *idle*, *surveillance, transit, engagement,* and *pursuit*. Upon entering a new mode, vehicles examine the current modes in which their neighbors operate in order to select potential agents for cooperation. Once those candidates are identified, a sequence of negotiations involving requests and current constraints results in making a decision whether or not vehicles should join the cooperative effort, and may in turn lead to changes in the tasks performed by members of the new group. For example, during a surveillance mission, vehicles that cooperate with other craft operating in the surveillance mode can do so in a passive or active fashion. Passive cooperation is possible with vehicles operating in other modes, as long as they have at least one operational sensor covering a portion of the designated mission area. Active cooperation requires that supporting vehicles alter their routes in order to realize the benefits from expanded coverage of their sensors. Since different search areas can be delegated to different vehicles, an interesting form of active collaborative surveillance occurs when there exists a partial overlap between the search areas of the two vehicles, and they must adjust their search routes accordingly in order to cooperatively maximize collective coverage.

The second layer of the supervisor focuses on *allocating tasks* within the operational regime specified by the mode selection layer described above. Tasks within a given mode share the same general goal, and specify the role taken by each individual craft to achieve that goal. For instance, within the surveillance mode, asset protection and persistent intelligence, surveillance, and reconnaissance (ISR) are two tasks that both share the common goal of searching an area for potential threats. However, in order to achieve the ISR objectives, the ships can operate within a well-defined, static area; on the other hand, protection of mobile assets requires that the search area be altered in real-time in a quasi-unpredictable manner (i.e., escorting a moving asset). Within the transit mode, one can find tasks such as dedicated transit, in which the ASV focuses exclusively on minimizing its travel time to the destination, but note that even this mode permits the ASVs to pursue other secondary goals, such as increasing the threat detection probability, by sharing the results of area coverage obtained with their on-board sensors.

We developed specific task allocation algorithms for each mode. In some cases, such as the surveillance mode, the task allocation is determined by the characteristics of the mission and the various constraints provided by a human operator. In other cases, such as the transit mode, the tasks determine different roles that can be carried out simultaneously by multiple ASVs to achieve the common goal. In this case, the allocation of individual duties is solely a result of the active negotiation between the cooperating agents.

The third layer of the decentralized hierarchical supervisor is the *trajectory generator*, responsible for the generation of the navigation paths for the ASV, with the implicit goal of maximizing the vessel's performance on the current task. During the persistent ISR task, the goal of the trajectory generator is to plot a route that maximizes the cooperative coverage of an area, while during a transit task, the trajectory generator may concern itself only with computing a sequence of waypoints that guide the vehicle from its current position to its desired location along the shortest, feasible path, i.e., without violating the traversable constrains (such as terrain) of the given mission area.

The fourth and final layer in the supervisor architecture is the *tracking controller*. Fundamentally different from the other three layers, this component is strongly platform dependent, because its goal is to execute all the commands that are specific to the vehicle's actuators in order to advance it toward the trajectory assigned by the third layer.

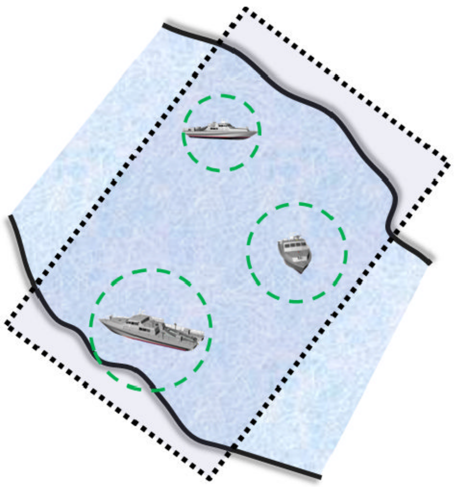


Fig. 13.3 Three vehicles cooperate performing the persistent ISR task, while operating in the surveillance mode

This chapter provides a detailed analysis of two of the four modes: surveillance and transit. Discussion of the surveillance mode focuses on a cooperative persistent ISR task, while the description of the transit mode emphasizes the two tasks involving autonomous negotiations between the ASVs to assign leader and follower roles, each with a different set of objectives. The following sections explain the objectives and challenges of each operating mode, and provide an overview of the algorithms implemented to support the trajectory generation layer. A trajectory generation device, such as a standard autopilot, is assumed to be available for the implementation of the described technique.

13.3.1 Persistent ISR Task

In this section, we consider the surveillance mode, and in particular the persistent ISR task, in which ASVs monitor a specified region of space for potential threats. An example of the implementation of such a task is shown in Fig. 13.3, where three craft position themselves in a way that increases the likelihood of detection of potential threats moving across the traversable area of the designated search area (shown in the illustration as the region delimited by the dotted rectangle). As detailed in [6], the likelihood of detecting a mobile threat in such scenarios is directly proportional to the average frequency of the surveillance coverage, and is inversely proportional to the variability of the coverage rate frequency across the search area. In other words, an effective ISR strategy must cover the entire area of interest in a homogeneous manner as frequently as possible.

The search task for surface vehicles described here was initially proposed and developed (see [2]) in the context of ISR missions for unmanned aerial vehicles (UAVs). Our decentralized search algorithm expresses individual and collective

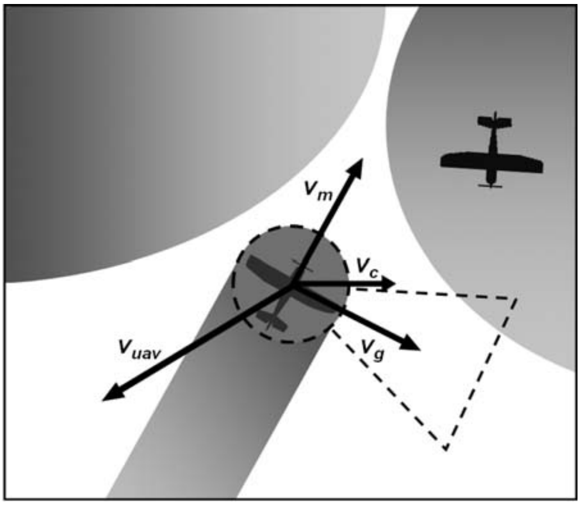


Fig. 13.4 Example of the four search vectors acting on a single UAV. The search boundary is on the *left* and a cooperating UAV is to the *right*. *Shaded areas* represent regions that were already scanned by the UAVs

goals in four weighted directional vectors which, when summed, provide the desired future heading **h** of a vehicle,

$$\mathbf{h} = w_g \mathbf{v_g} + w_{veh} \mathbf{v_{veh}} + w_c \mathbf{v_c} + w_m \mathbf{v_m} \tag{13.1}$$

where $\mathbf{v_g}$ is the goal vector, $\mathbf{v_{veh}}$ is the vehicle spread vector, $\mathbf{v_c}$ is the vector that repels the vehicles from the search area boundaries, $\mathbf{v_m}$ is the momentum vector, and the w's are their corresponding weights (see Fig. 13.4). New desired headings are calculated at fixed discrete moments, triggered by the arrival of new sensor data [2].

In practice, additional factors need to be considered to adapt this search method for aquatic surface vehicles. Because the vessels operate in the same horizontal plane, collision avoidance procedures must be considered, and since navigable areas are limited by shore lines and coastal rights of way, the ASV controller has to ensure that multiple external constraints are satisfied [1]. To restrict the search to a traversable area (assumed to be known a priori), the search algorithm originally developed for UAVs was modified to account for the change in the spatial coverage. To prevent nontraversable areas, such as land masses and bridge structures, from misdirecting the goal vector $\mathbf{v_g}$, the map of traversable areas is treated as an overlay onto the coverage map to assign a zero probability to the detection of targets in the nontraversable areas (independent of the spatial coverage history). This overlay is applied only during the calculation of the goal vector $\mathbf{v_g}$, while the overall target detection probability map is transferred to the next iteration, preserving the complete sensor coverage information stored within it.

Having determined the desired heading **h** via a consideration of the traversable area, the next step is to anticipate whether the ASV will attempt to cross nontraversable sections of the space while maintaining its assigned heading. Consider the case shown in Fig. 13.5 where the heading **h**, derived from the four intentionality vectors of (13.1), indicates the direction with the greatest impact on the likelihood of target detection. If we were to assess the feasibility of the heading **h**, we would

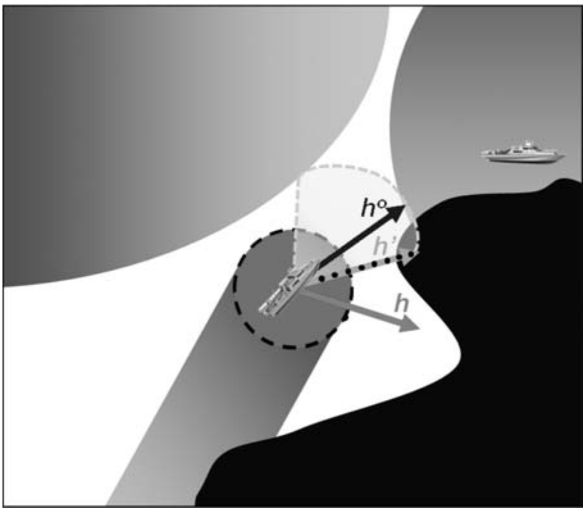


Fig. 13.5 Illustration of the determination process for computing a feasible heading $\mathbf{h}'$ and the valid heading $\mathbf{h^o}$. *Shaded areas* represent regions that were already scanned by the UAVs, while the *black area* on the *right side* of the image represents a nontraversable zone

conclude that it was a safe maneuver for the ASV to perform. However, restrictions on vehicle dynamics under the specific trajectory tracking controllers will limit the turn rate by some maximum value. This maximum turn rate, assumed to be known as part of the physical characteristics of the vehicles, can be translated into a "feasibility" cone, such as the one shown in Fig. 13.5, centered around the craft's current heading, with an extension equal to the distance that can be reached by the vessel under its current speed within the discrete time evaluation period of (13.1), and the aperture based on the maximum turning angle achievable by the craft within the same time interval. If the intended heading $\mathbf{h}$ falls within the angle range constrained by the feasibility cone, the heading $\mathbf{h}$ is assigned as the feasible heading $\mathbf{h}'$ (i.e., $\mathbf{h}' = \mathbf{h}$). On the other hand, if the intended heading $\mathbf{h}$, as shown in Fig. 13.5, is located outside of the feasibility cone, the boundary of the feasibility cone with the smallest included angle with respect to $\mathbf{h}$ is selected as $\mathbf{h}'$.

Before being relayed to the low-level tracking controller, the feasibility of $\mathbf{h}'$ must be confirmed. This is accomplished by evaluating the direction indicated by the vector $\mathbf{h}'$ at fixed spatial intervals against the traversability map. If any of the evaluated points indicate a nontraversable section of the space, we iteratively compute other heading candidates within the feasibility cone, starting with those that have the smallest angle change from the original $\mathbf{h}'$ vector. Once a heading vector is found such that it results in a traversable path, it is labeled as the valid heading $\mathbf{h^o}$, and is given to the tracking controller for execution.

However, note that in certain circumstances, such as the one pictured in Fig. 13.6, no valid heading $\mathbf{h^o}$ may exist within the feasibility range determined by the craft's current speed. In such circumstances, progressive reductions in the craft's speed are considered. Note that with each reduction in the speed, the feasibility cone decreases its extension, since the amount of ground covered until the next iteration is also reduced. Also, through a similar process, the aperture of the cone increases as the dynamic characteristics of the vehicle allow for greater turn rates at slower speeds.

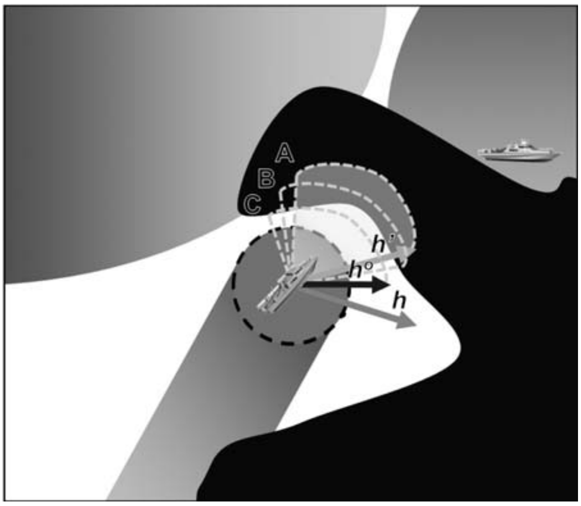


Fig. 13.6 Determination of the feasible heading $\mathbf{h}'$ and the valid heading $\mathbf{h^o}$ in situations that require a speed reduction. Feasibility cones A, B, and C are related to progressively slower speeds. *Shaded areas* represent regions that were already scanned by the UAVs, while the *black area* on the *right side* of the image represents a nontraversable zone

As in the example depicted in Fig. 13.6, the process of progressive speed reduction continues until a valid heading $\mathbf{h^o}$ is found.

13.3.2 Transit

The primary objective of the transit task is to transport an ASV from its current location to a goal destination. In particular, for the purposes of this chapter (in which only the surveillance and transit modes are considered), the transitions between the two modes, shown in Fig. 13.2, suggest that all transit tasks terminate when the designated search area is reached. Thus, if at a particular moment in time, a single craft is on its way to a specific search area performing the transit task, then the minimization of travel time is of the highest priority to the ASV controller. Therefore, the challenge for the single craft operating in the transit mode is to determine a path to its final destination; a path that does not violate the restrictions embedded in the traversability map. In such cases, the same technique as was described in Sect. 13.3.1 can be applied, with the intended heading $\mathbf{h}$ determined as a unit vector in the direction of the destination search area center.

On the other hand, if multiple vehicles share a common destination, then a negotiation among the craft is necessary. One of the ships declares itself as the group leader, while the rest of the fleet will take up the follower role. The selection of the leader can be based on proximity to the destination, but other characteristics, such as speed of vehicles, can also act as decision factors in heterogeneous teams of ASVs. The leader craft selects a trajectory generation algorithm identical to the one employed in the single craft case. The boats that follow, however, in addition to reaching the specified destination, must also assist in increasing the joint sensor coverage range during the transit by assuming one of the positions indicated by

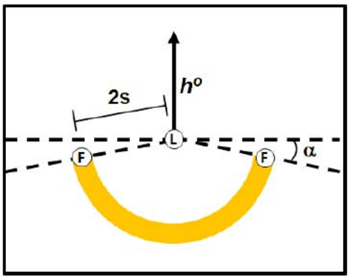


Fig. 13.7 Resource assignment for the cooperative transit task. The *highlighted semi-arch* represents the area of relative position candidates where a follower craft (F) can be positioned with respect to the leader craft (L) in order to maximize the joint coverage of omni-directional sensors with range s. The displacement angle α is measured from an orthogonal line to the leader's current valid heading $\mathbf{h^o}$

the highlighted area in Fig. 13.7. The leader ASV does not consider the followers' actions in this implementation in order to avoid the risk of creating potentially unstable feedback loops, as well as to ensure that the leader arrives at the destination as quickly as possible. Note that the actual location of the semi-arch of potentially suitable locations is determined based on the position of the leader and its currently reported valid heading $\mathbf{h^o}$. Similarly, the distance s is defined as the range of the on-board omni-directional sensor, while α is a small variable angle related to the speed adjustment function, discussed at length in Sect. 13.3.2.2.

The discussion that follows presents the task allocation algorithm for the transit mode, as well as the trajectory generation algorithm for the ASVs operating as transit followers.

13.3.2.1 Transit Mode Task Allocation

In some circumstances, mission priorities may not allow a transit vehicle to deviate from its course; still, it may be able to passively assist others. For instance, the path of a vehicle dedicated solely to transit may cross a surveillance area assigned to another vehicle. Without deviating from its path, the transit vessel can scan the area with its sensors for the same types of threats that the surveillance boat is targeting, thus momentarily increasing the total sensor coverage of the fleet and eliminating the need to search the transit corridor [8]. On the other hand, when the constraints on the transit task assigned to the ASV are flexible, it may be possible to temporarily deviate from the transit path in order to meet secondary mission goals [2]. When multiple ASVs simultaneously engage in the transit task along adjacent routes, advantageous cooperation may be realized with minimal extra effort.

Consider a simplified scenario in which two crafts, operating in transit mode and sharing the same destination, find themselves out on the open water having the same orientation with respect to a common reference frame. For this straightforward configuration, it is almost obvious that it would be more efficient to have the vessel

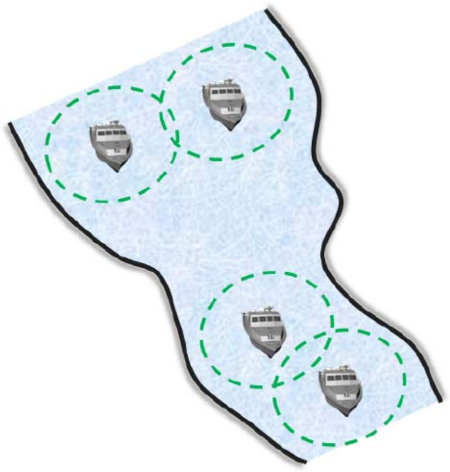


Fig. 13.8 Formation control for sensor coverage optimization during transit

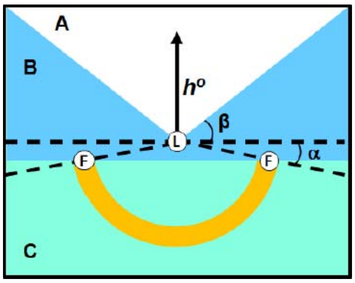


Fig. 13.9 Relative spatial regions for mode transition

closest to the final destination select itself as the group leader, and leave the other to position itself in formation behind the leader. However, if the poses of the two ships do not match at the beginning of this task, the ASV nearest to the goal may need to move further away to execute the desired maneuver, with each vessel obeying the constraints of its physical dynamics and propulsion limitations. Furthermore, in more realistic scenarios, such as the one visualized in Fig. 13.8 where the geometry of the waterway impedes the boats' movements, predicting which specific craft can achieve the shortest arrival time at the final destination in a distributed fashion (in which information is only available locally and subject to change by other uncontrolled actors in the environment) is extremely difficult, and is subject to change several times before the goal location is reached.

Consider the diagram shown in Fig. 13.9 in which the angle β is added to the original configuration of Fig. 13.7, defining three separate regions (A, B, and C) relative to the position and valid heading $\mathbf{h^o}$ of the fleet leader. The task allocation algorithm listed in Fig. 13.10 makes use of such regions to establish the negotiation protocol that is employed by each and every member of the group operating in the transit mode and having the same destination; the purpose of this procedure is to select a single fleet leader in a distributed manner by iteratively converging to a consensus. Note that the information required to execute this real-time negotiation

```
if no leader is present
  become leader
else
  if there are multiple leaders that share the same destination
    if this is the closest vehicle to the destination
      become leader
    else
      become follower
  else
    if this vehicle is the current leader
      and a follower is located within the region $A$
        become follower
    else
      if this vehicle is currently a follower
        and it is located within the leader's region $A$
          become leader
```

Fig. 13.10 Task allocation algorithm for the transit mode

protocol consists of the current position, pose, mode, and task of the neighboring vessels.

Note that the evaluation of the other possible scenarios, such as the presence of no leaders or of multiple leaders, is necessary for the successful resolution of the transitory states that can be generated in the practical implementation of the negotiation algorithm due to communication delays, for example. Furthermore, instead of a direct threshold for the mode transitions in the final **else** statement, the use of the relative region A creates a hysteresis space which encompasses regions B and a part of C, and prevents the emergence of oscillatory transitions between the ASV roles due to the use of different trajectory generating strategies for each task.

13.3.2.2 Follower Trajectory Generating Algorithm

Having previously described the trajectory generating algorithm of the leader craft as a simplification of the search procedure described in Sect. 13.3.1, here, we introduce a trajectory generation algorithm for the follower craft. As previously mentioned, as a follower, an ASV must achieve both the primary goal of arriving at the destination, and the secondary goal of maximizing the overall sensor coverage of the entire group. While the process of evaluating the desired heading $\mathbf{h}$ remains the same (see Sect. 13.3.1), the algorithm for generating the $\mathbf{h}$ is modified to better fit the priorities of a follower unit. For these boats, the desired heading $\mathbf{h}$ is a normalization of the sum of the two vectors: the tracking vector $\mathbf{t}$, and the positioning vector $\mathbf{p}$, as illustrated in Fig. 13.11. The purpose of the tracking vector $\mathbf{t}$ is to ensure that the followers maintain their relative positions with respect to the current leader, hence $\mathbf{t}$ is defined along the same direction as the valid heading $\mathbf{h^o}$ of the leader.

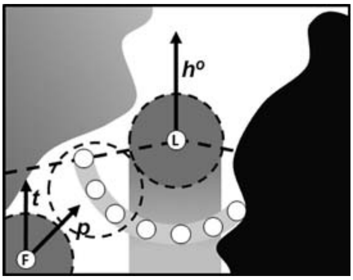
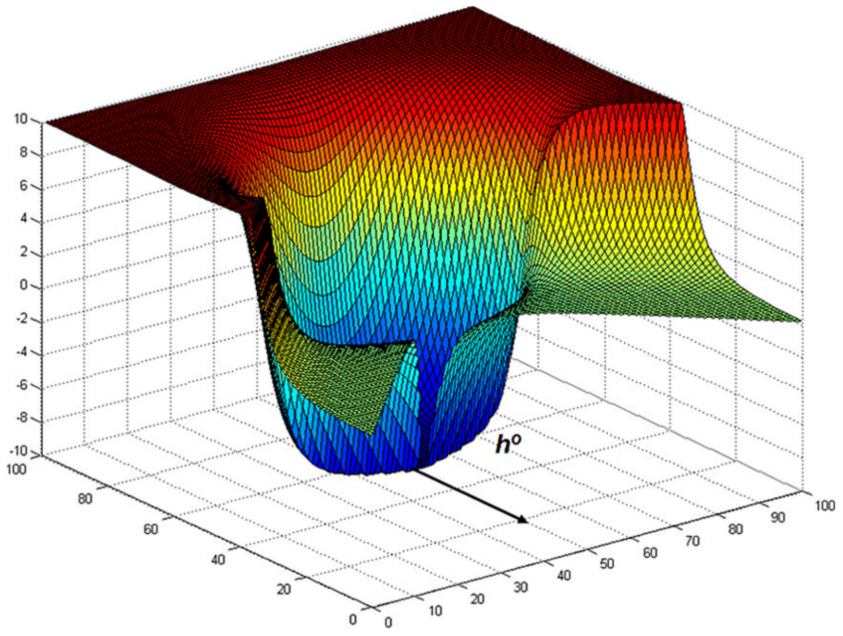


Fig. 13.11 Example scenario for the generation of the tracking vector **t**, and the positioning vector **p** for the determination of the desired heading **h** of a follower craft. *Shaded areas* represent regions that were already scanned by the ASVs, while the *black area* on the *right side* of the image represents a nontraversable zone

The positioning vector **p** guides the craft to a point around the leader craft within the semi-arch of a radius equal to twice the sensor range of the vehicles, so that the joint area coverage of the group is maximized. The direction of **p** is defined in terms of the current position of the follower, and the position along the previously computed semi-arch around the leading vessel that is both feasible, and results in the greatest gain for the combined sensor coverage of the fleet, taking into consideration the traversability map and the mission's coverage status map [7]. This position is selected from a set of finite, equally spaced candidates along the formation semi-arch, as in the example configuration depicted in Fig. 13.11. While the direction vector **t** is a unit vector, the norm of vector **p** is equal to the distance between the current position of the follower craft and the selected position along the formation semi-arch. Constructed in this manner, the resulting desired heading **h** is dominated by **p** at greater distances, but as the desired position is approached, the influence of **t** increases gradually, allowing the vehicles to maintain formation. Once a desired heading **h** is established, it evaluates $\mathbf{h}'$ before generating the craft's final navigation vector $\mathbf{h^o}$. Recall that the obstacle avoidance routine, which constantly operates at a lower control level, will prevent any collision-prone $\mathbf{h^o}$ from being undertaken by a vehicle.

We should note that in order to generate the necessary formations in all of the potential collaborative transit scenarios, control of the heading alone is insufficient. In order to "catch up" with a distant leader, or to yield when moving ahead of the desired relative position, it is also necessary to implement some form of speed control. For this purpose, we developed a new speed control function shown in Fig. 13.12, where fixed speed-differential values are assigned based on the relative positions of the followers with respect to the transit leader. The speed control map is tuned to fit the vehicle's dynamic constraints, particularly with respect to the maximum achievable and minimum effective speeds. In all cases, however, it is important that the final curves traced by the craft's control algorithm through the state space be smooth and at least piece-wise continuous. To fulfill both of these requirements, the



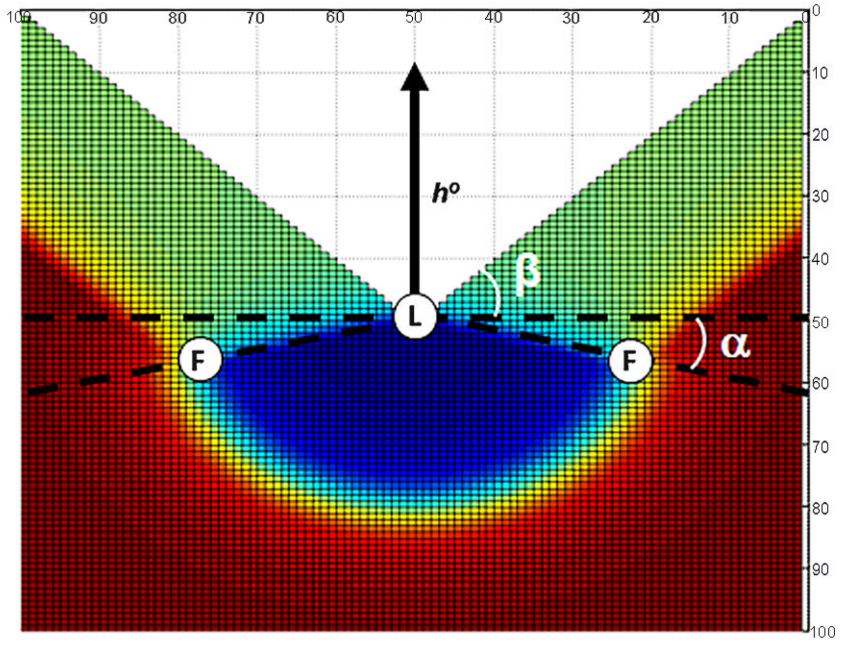


Fig. 13.12 Sample speed adjustment maps shown in 3D and 2D perspectives. *White circles* indicate task leader location (L) and optimal follower locations (F); the map orientation vector $\mathbf{h^o}$ indicates the leader's current valid heading. *Bright shaded areas* represent a neutral relative speed and serve as follower attractors, while the *darker area* in the circular section in the *center* is related to slower speeds, and the *darker area outside* corresponds to faster speeds

generated speed map was constructed from sigmoidal primitives of the form:

$$\Delta S = A\frac{x}{\sqrt{1+x^2}}$$

where ΔS is the speed differential over the vehicle's standard cruise speed, A is the amplitude of the maximum allowed speed differential ($A = 10\,\text{m/s}$ in the example shown in Fig. 13.12), and x is the distance to a relative point in the speed map with respect to the leader's location (a valid heading is chosen to generate the desired shape).

From the speed map's three dimensional representation (Fig. 13.12, left), observe that in the formation semi-arch region, the speed differential is zero, and since the heading of the follower at these locations matches that of the leader (i.e., **p** has a zero norm), these areas represent followers' convergence points. The speed-differential map is not defined in region A because as soon as a follower moves into this position, the task assignment algorithm (see Sect. 13.3.2.1) will select it as the new leader, and the ASV will not alter its speed from the standard cruise value.

13.4 Simulation Results

To demonstrate the benefit of cooperation that can be realized with the proposed search and transit task algorithms, we simulated missions using both cooperating and noncooperating ASVs. The simulated mission conducted under both configurations consisted of searching two separate locations within a designated mission area. These two search regions were located in different portions of the mission area, separated by a "river" (i.e., a narrow traversable channel such that no single constant heading can be used to navigate from one search area to another).

The mission began with the two ASVs at different traversable locations selected at random within the south-east (SE) search area. After 20% of the total mission time had elapsed, both ASVs were instructed to perform the search of the north-west area, followed by another search command after 65% of the mission time, indicating once more the need to search the original SE search area. Figure 13.13 shows an example trace of the cooperative ASV simulation in which the search begins at the bottom-right corner, then switches to the second search area at the top-left corner of the mission area.

In order to generate results for the same mission operating without cooperation among ASVs, we eliminated the ship-to-ship communication in the second set of simulations. During the search task, a lack of communication results in each unit not being aware of the area covered by the sensors of the other, therefore causing each craft to compute its movements based solely on the information contained in the individual on-board dynamic coverage map. During the transit task, since each vehicle lacks the knowledge of the common destination it shares with its neighbor, both assume the role of leaders, and proceed to minimize the arrival time without attempting to increase the joint sensor coverage. An example of a run without the cooperation between the craft can be seen in Fig. 13.14.

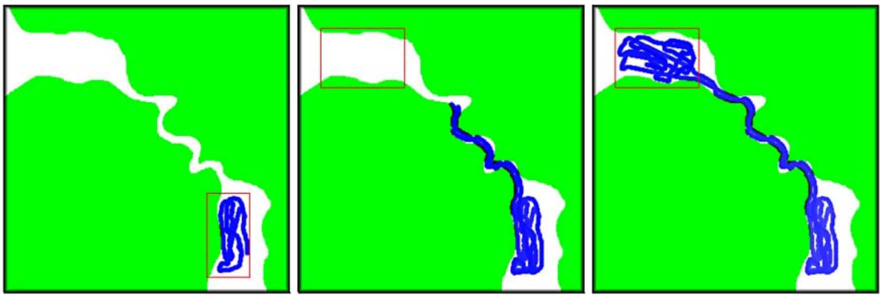


Fig. 13.13 Visualization of the multi-objective mission involving the collaborative search of a designated area in the SE corner of the map (*left*) by two autonomous ASVs, followed by a search-in–transit while navigating a river channel in formation (*middle*), and resumption of the search within the NW zone (*right*)

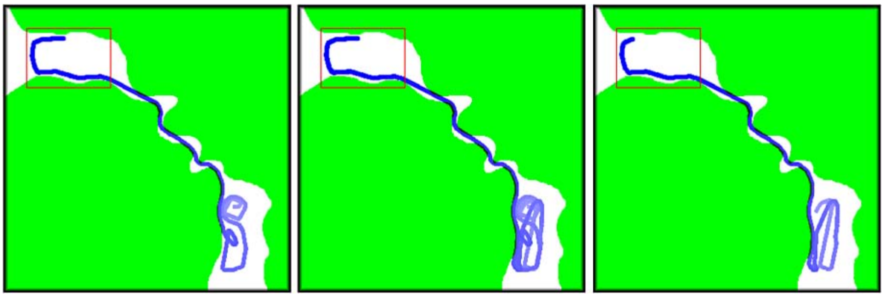


Fig. 13.14 Visualization of the multi-objective mission without communication between ASVs. The map on the *left* shows the self-perceived coverage of the first ASV. The map on the *right* displays the same content from the perspective of the second ASV. The map in the *center* shows the combined effect on sensor coverage from both ASVs, generated after the end of the mission by a global observer

To generate statistically accurate results, a total of 20 empirical experiments were performed for each mission scenario. The comparison between the average coverage of the entire search region (i.e., the two search areas and the river channel connecting them) over time in the two scenarios is plotted in Fig. 13.15. Note that a marked improvement in the search coverage due to the cooperation between the two boats can be seen during the initial moments of the simulation, when the mission goal is to search the SE area. The difference in slopes of the two curves increases as the cooperative behavior exhibited by the craft that assumes the follower role during transit provides additional coverage in the region between the two search areas. A pronounced increase in the sensor coverage performance can also be seen during the search performed in the NW search area, where the absence of cooperation between the ASVs becomes particularly detrimental. This result is due in part to the fact that the lack of communication caused the craft to use nearby entry points into the search area after emerging from the river channel, which subsequently leads to highly similar navigation decisions (i.e., the two craft end up covering the same

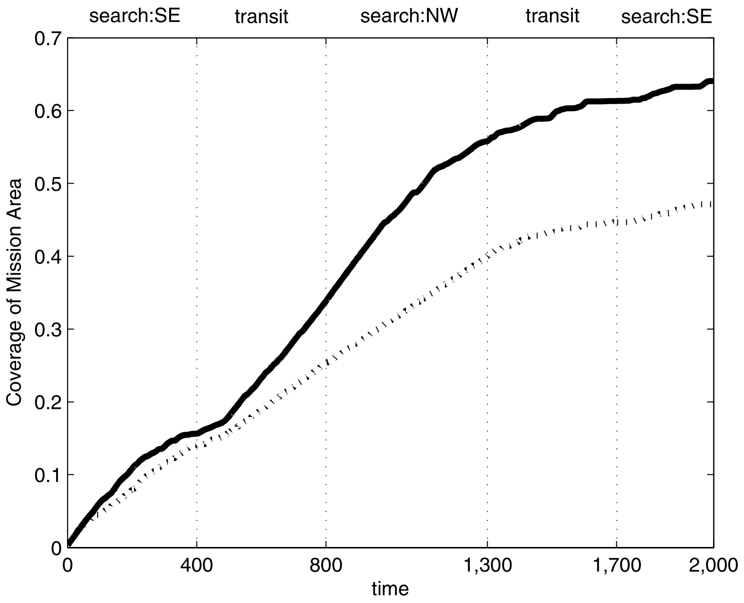


Fig. 13.15 Comparison of the coverage of the total mission area achieved during each of the search and transit intervals of the mission for the cooperative (*solid*) and independent (*dashed*) scenarios

regions). On average, within the parameters of the constructed mission, the sensor coverage of the SE area increased by 17.53% when cooperation was allowed, while the much larger gain of 48.89% was realized in the NW area. During the transit from the SE area to the NW area, the execution of the proposed cooperative transit algorithm resulted in a 47.06% increase in the coverage of the region connecting the two search areas. Using the Wilcoxon rank sum test [3], we found that the task performance improvements achieved by the cooperative controller are statistically significant from the noncooperative version ($p = 0.01$).

13.5 Conclusion and Future Work

In this chapter, we presented a novel decentralized hierarchical supervisor architecture for unmanned surface vehicles composed of four layers for mode selection, task allocation, trajectory generation, and tracking control. For vehicles deployed on surveillance missions involving multiple disjoint areas, we have developed the modes of surveillance and transit, and described the autonomous algorithms that compose each of the corresponding supervisor layers. Simulation results from testing the application of the proposed architecture revealed statistically significant benefits of the cooperative control algorithm over conventional, independent approaches in terms of increasing the performance of concurrent search and transit objectives with only a modest increase in the ASVs' on-board hardware and communication requirements.

Advances outlined here comprise a part of a much larger research effort concerning the technological gap between current state-of-the-art manned and unmanned systems. The long-term goal of this work is to construct a comprehensive framework for analysis and development of such collaborative technologies, and the overall feasibility of our approach for solving real-world problems is evident from the numerous successful outcomes we achieved with multiple autonomous platforms on a set of nontrivial, complex, challenging tasks [10].

References

1. Benjamin, M., Curcio, J., Leonard, J., Newman, P.: A method for protocol-based collision avoidance between autonomous marine surface craft. J. Field Robotics **23**(5) (2006)
2. DeLima, P., Pack, D.: Toward developing an optimal cooperative search algorithm for multiple unmanned aerial vehicles. In: Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS-08), pp. 506–512, May 2008
3. Gibbons, J.D.: Nonparametric Statistical Inference. Marcel Dekker, New York (1985)
4. Matos, A., Cruz, N.: Coordinated operation of autonomous underwater and surface vehicles. In: Proceedings of the Oceans 2007, pp. 1–6, October 2007
5. Pack, D., York, G.: Developing a control architecture for multiple unmanned aerial vehicles to search and localize RF time-varying mobile targets: part I. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 3965–3970, April 2005
6. Pack, D., DeLima, P., Toussaint, G., York, G.: Cooperative control of UAVs for localization of intermittently emitting mobile targets. IEEE Trans. Syst. Man Cybern. Part B, Cybern. **39**(4), 959–970 (2009)
7. Plett, G., DeLima, P., Pack, D.: Target localization using multiple UAVs with sensor fusion via Sigma-Point Kalman filtering. In: Proceedings of the 2007 AIAA (2007)
8. Plett, G., Zarzhitsky, D., Pack, D.: Out-of-order Sigma-Point Kalman filtering for target localization using cooperating unmanned aerial vehicles. In: Hirsch, M.J., Pardalos, P.M., Murphey, R., Grundel, D. (eds.) Advances in Cooperative Control and Optimization. Lecture Notes in Control and Information Sciences, vol. 369, pp. 22–44 (2007)
9. Willcox, S., Goldberg, D., Vaganay, J., Curcio, J.: Multi-vehicle cooperative navigation and autonomy with the bluefin CADRE system. In: Proceedings of the International Federation of Automatic Control Conference (IFAC-06), September 2006
10. Zarzhitsky, D., Schlegel, M., Decker, A., Pack, D.: An event-driven software architecture for multiple unmanned aerial vehicles to cooperatively locate mobile targets. In: Hirsch, M.J., Commander, C., Pardalos, P.M., Murphey, R. (eds.) Optimization and Cooperative Control Strategies. Lecture Notes in Control and Information Sciences, vol. 381, pp. 299–318 (2009)

Chapter 14
A Connectivity Reduction Strategy for Multi-agent Systems

Xiaojun Geng and David Jeffcoat

Summary This paper considers the connectivity reduction of multi-agent systems which are represented with directed graphs. A simple distributed algorithm is presented for each agent to independently remove some of its incoming links based on only the local information of its neighbors. The algorithm results in an information graph with sparser connections. The goal is to reduce computational effort associated with communication while still maintaining overall system performance. The main contribution of this paper is a distributed algorithm that can, under certain conditions, find and remove redundant edges in a directed graph using only local information.

14.1 Introduction

Groups of multiple agents have been studied with the aid of algebraic graph theory, for example, in [2, 4, 7, 8, 10, 11], and [5]. Directed or undirected, weighted or unweighted graphs are used to characterize a network of multiple agents, in which agents are represented by vertices of a graph and information interactions by arcs/edges. Once a graph is constructed for the networked agents, decentralized control laws are applied to drive the behavior of each agent using only information available to that agent. This information comes from its own sensing devices or from other agents through communications.

Generally, more local information available to each individual agent results in better performance for the overall group. The second smallest eigenvalue of the Laplacian matrix, also called the algebraic connectivity of the graph, has been used

X. Geng (✉)
California State University, 18111 Nordhoff St., Northridge, CA 91330, USA
e-mail: xjgeng@csun.edu

D. Jeffcoat
Air Force Research Laboratory, Eglin AFB, FL 32542, USA
e-mail: david.jeffcoat@eglin.af.mil

M.J. Hirsch et al. (eds.), *Dynamics of Information Systems,*
Springer Optimization and Its Applications 40, DOI 10.1007/978-1-4419-5689-7_14,
© Springer Science+Business Media, LLC 2010

as a measure of network connectivity [6, 12]. For undirected graphs (corresponding to bidirectional communications or sensing), Fielder [6] shows that the second smallest eigenvalue of the Laplacian grows monotonically with the number of edges.

The network connectivity may change dynamically due to agent motion or communication failure. For this reason, preservation of network connectivity has been a goal of multi-agent coordination. Along this line, a control law is designed in [1] by transforming the connectivity requirement into motion constraints on each agent. Special potential functions are used in [7] to design control laws to guarantee connectedness. The above approaches preserve all the initial connections under the assumption that the initial graph is connected.

Instead of preserving connectivity, this paper aims to simplify network connections by removing some redundant edges. A motivation of this idea comes from situations where significant dispersion behaviors of agents may be expected. A meshed network with intricate information exchanges offers fault tolerant capability to the system, but at the price of heavy computational and communications burden. In addition, preserving all initial connections may impose too much restriction on the motion of agents resulting in performance degradation, especially for coordination tasks requiring significant dispersion.

This paper uses directed graphs (digraphs) to represent the information interaction between agents, in which information may flow in one direction only. Unidirectional communication exists when some agents have only transmitter or receivers, or when transmission power varies between agents. In addition, different weights can be assigned to different communication links to represent the level of trust on the information received through a particular link.

In this paper, a simple distributed scheme is developed for each agent to select its information sources based on only local communication topology. The result of this algorithm is a relatively sparser graph, which will be referred to as "reduced digraph" in the paper. It is expected that coordination of agents based on reduced digraphs will be more efficient in some applications.

The paper is organized as follows. In Sect. 14.2, we introduce the system model of multiple agents, some definitions and results from graph theory, and a measure of edge robustness. Section 14.3 defines triangle closures, presents the principles used to select redundant edges in triangle closures, and describes the distributed algorithm for each agent to select its information sources using only local triangle topologies. The algorithm produces a reduced system digraph. Some discussion about the graph reduction and an illustration of the algorithm are in Sect. 14.4, followed by concluding remarks in Sect. 14.5.

14.2 Background

14.2.1 Model

Consider a group of n heterogeneous vehicles in 2-dimensional Euclidean space. Each agent is assigned a label $1, 2, \dots, n$. Let $\mathbb{I} = \{1, 2, \dots, n\}$ be the index set (adopted as IDs in the paper) of the agents. Each vehicle i is associated with a position vector $x_i(k) \in \mathbb{R}^2$ at time k, where $i \in \mathbb{I}$. We assume each vehicle i is equipped with a transceiver for which the transmission range is δ_i. Note that we allow the transmission range for each agent to be different to model the situation where agents may carry different equipment or different power capacity. With the above setup, if agent i is located within a closed disk of radius δ_j centered at $x_j(k)$, then agent i can receive information from agent j at time k.

A weighted simple directed graph $G(k) = (V, E(k), W(k))$ is used to model the communication topology among agents at time k, where V is the set of n vertices corresponding to n agents, $E(k) \subseteq V(k) \times V(k)$ is the edge set consisting of ordered pairs of vertices, and $W(k)$ is the set of weights assigned to each communication link in $E(k)$. For any edge $(i, j) \in E(k)$, there corresponds a directed edge from j to i in the graph, where i is called the terminal node/vertex and j the initial node/vertex, and we say that i is *reachable* from j, or j can reach i. It also implies that i is within the transmission range of j, i.e.,

$$\left\| x_i(k) - x_j(k) \right\| = \delta_j$$

For edge $(i, j) \in E(k)$, $w_{ij}(k) \in W(k)$ is its weighting factor, perhaps reflecting the reliability or significance of the information flow from j to i. In this paper, we define the in-degree of vertex i as the number of edges whose terminal nodes are i, and the out-degree of vertex i as the number of edges whose initial nodes are i.

The communication topology represented by $G(k)$ may change from time to time due to vehicle movements. At time k, all the agents from which i can receive messages constitute the neighbor set of i, denoted as

$$N_i(k) = \left\{ j \in V(k) | (i, j) \in E(k), \ \& \, i \neq j \right\}$$

In the following sections, we sometimes drop k from the notation of $G(k)$ since the algorithm does not depend on past or future graph topologies.

A directed walk in a digraph G is a sequence of directed edges (i_1, i_2), (i_2, i_3), $\ldots, (i_{p-1}, i_p)$ in E where $i_1, \ldots, ip \in \mathbb{I}$, and it is a directed path if further $i_1 \neq i_2 \cdots \neq i_{p-1}$. If so, we say i_1 can be reached by i_p. A digraph is *strongly connected* if there is a directed path between any pair of distinct vertices. A *subgraph* $\hat{G} = (\hat{V}, \hat{E})$ of a directed graph $G = (V, E)$ is a directed graph that satisfies $\hat{V} \subseteq V$ and $\hat{E} \subset E$. A directed tree is a digraph in which every vertex is a terminal node of one and only one edge, except one node which is called the root of the tree. A *spanning tree* of a digraph G is a subgraph of G and a tree that connects all nodes of G.

14.2.2 Edge Robustness

In this paper, we allow each agent to drop messages from some of its neighbors according to the reliability measure of these communication links. The motivation is that an agent would rather ignore certain information if the associated link is not reliable, assuming that the same piece of information may be obtained elsewhere. Communication flows, represented as directed edges, are selected in this paper in terms of their robustness. The concept of robust connectivity is taken from [4], where edge robustness is defined for undirected graphs in which bidirectional communications are assumed.

Denote the transmission range of agent A_i as δ_i, and let the distance between A_i and A_j be $d_{ij}(k) := \|x_i(k) - x_j(k)\|$, where $i, j = 1, \ldots, n$. We define the edge robustness as follows.

Definition 1 Given a directed graph $G(k) = (V, E(k), W(k))$, for edge $(i, j) \in E(k)$, the robustness at time k is given by

$$r_{ij}(k) = \frac{\delta_j - d_{ij}(k)}{\delta_j} \times w_{ij}(k)$$

From the above definition, it follows that the robustness value satisfies $0 = r_{ij} = 1$ for edge $(i, j) \in E(k)$, assuming the weight satisfies $0 = w_{ij}(k) = 1$. This robustness value won't be negative due to the fact that $d_{ij}(k)$ is not greater than δ_j if i can hear j. The robustness is a measure of the flexibility agent i has in its relative motion with j. The higher the robustness of the link, the more maneuver range the agent has, and the more reliable the link is. Note that the robustness of the edges (i, j) and (j, i) may be different, even when $w_{ij} = w_{ji}$. Applying this idea to paths with two edges, we can define the path robustness as described below.

Definition 2 Consider a directed graph $G(k) = (V, E(k), W(k))$, with $(i, j) \in E(k)$ and $(j, h) \in E(k)$. The path from h to i through j, denoted by (i, j, h), has the following robustness:

$$r_{ijh}(k) = \min(r_{ij}(k), r_{jh}(k))$$

The above definition can be extended to directed paths of any number of edges. When transmission ranges for all edges are the same, i.e., $\delta_1 = \cdots = \delta_n$, the communication topology becomes an undirected graph, in which we have $r_{ij} = r_{ji}$ for any undirected edge (i, j).

14.3 A Distributed Scheme of Graph Reduction

In Sect. 14.2, edge robustness is quantified for directed communication topologies. Using edge robustness as a measure, a distributed scheme will be presented here to drop some of the information links in the communication graph. As a result of executing this local scheme on each agent, a reduced graph of the group topology is formed.

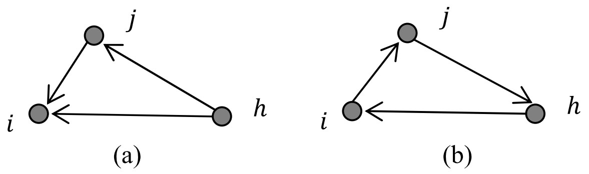


Fig. 14.1 (**a**) A triangle closure for vertex i, and (**b**) a directed triangle cycle

14.3.1 Redundant Edges and Triangle Closures

First, we define redundant edges in directed graphs. Given a directed graph $G(k) = \{V, E(k)\}$, we say that an edge $(i, j) \in E(k)$ is *redundant* if the vertex i is still reachable from j after the edge (i, j) is removed. Finding and removing all the redundant edges in a directed graph is called a *transitive reduction* problem [1] or *minimum equivalent graph problem* [3]. This problem is proven to be NP-hard; approximation algorithms in polynomial time are studied [9]. However, to the best of our knowledge, no distributed algorithm using only local information from neighbors has been reported in the literature.

In this paper, we study a distributed algorithm of deleting redundant edges for directed graphs. Our objective is to allow each agent to select its information sources according to simple rules based only on the local information of its neighbors. Inspired by Spanos and Murray [4], we consider triangle topologies defined below.

Given a directed graph $G = (V, E)$, we say that three vertices $i, j, h \in V$ form a *triangle closure* if the edges (i, j), (j, h), and (i, h) are contained in E. An example is given in Fig. 14.1(a), where the in-degree of the vertex i is two and it has two neighbors j and h. There are two paths from h to i; one is the edge (i, h) and the other (i, j, h) consisting of two edges. For the case in Fig. 14.1(a), we say that the node i possesses a triangle closure, denoted by $\Delta(i, j, h)$ where (i, j, h) is an ordered triple corresponding to the longest path in the topology. Clearly, for this triangle closure, the reachability relation remains between i and its neighbors if the edge (i, h) is deleted.

For comparison, a directed triangle cycle for vertices i, j, and h is shown in Fig. 14.1(b). A *directed cycle* is a directed path in which the initial vertex of the path is also the terminal vertex of the path. Note that each vertex in this triangle cycle has only one neighbor, therefore no redundant edge exists in this topology. The following statement about directed cycles is true for strongly-connected graphs.

Lemma 1 *The minimum equivalent graph* (*no redundant edges*) *of a strongly-connected digraph is either a directed cycle of length* n, *or a group of joined directed cycles*, *where* n *is the number of vertices of the graph.*

Proof (Main idea) Since the graph is strongly connected, there is a directed walk from any agent i back to agent i going through all the other agents in the network. This walk is a directed cycle or a group of joined cycles, and furthermore, it doesn't contain redundant edges. □

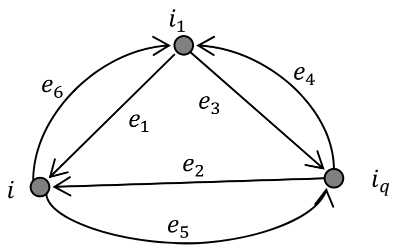


Fig. 14.2 Each possible edge among these three vertices i, i_1, and i_q is assigned a Boolean variable $e_1, \ldots, e_6$, which is 1 if there is a link and 0 otherwise

14.3.2 Local Triangle Topologies

In this paper, we build a distributed algorithm using local information to identify triangle closures and redundant edges. As a major part of this algorithm, each agent determines the redundancy of its two incident edges based on the local topology with a pair of its neighbors, for which the details are stated below. Note that in the following discussion, we assume the communication links are unweighted.

First of all, we assume the following information is included in the messages that an agent i broadcasts to its neighbors:

$$\{i, x_i(k), \delta_i(k)\} \tag{14.1}$$

i.e., the agent ID, the current position, and the present transmission range. In other words, each agent transmits only its own current information. Using received messages from its neighbors, agent i forms its neighbor set $N_i(k)$, and computes the distance between any pair of its neighbors and the distance from any neighbor to itself. For example, if agent i receives messages from i_1 and i_q, it knows that they are its neighbors, i.e., $\{i_1, i_q\} \in N_i(k)$, and knows their positions $x_{i_1}(k)$, $x_{i_q}(k)$ and their transmission ranges $\delta_{i_1}(k), \delta_{i_q}(k)$. Using this information along with its own position $x_i(k)$ and transmission range $\delta_i(k)$, agent i can obtain the local communication topology among itself and these two neighbors.

According to the local communication topology among the three vertices, agent i will make one of three possible decisions: no redundant edge; edge (i, i_1) is redundant; or (i, i_q) is redundant. The decision is made according to a set of rules given below.

To simplify the description, we assign a variable to each possible edge joining any pair of the three vertices, as illustrated in Fig. 14.2. Each variable, $e_1, \ldots, e_6$, takes a Boolean value: 1 if the corresponding edge exists and 0 otherwise. These values are essentially the off-diagonal entries of the adjacency matrix for the local communication graph of these three vertices. If i_1 and i_q are neighbors of i in Fig. 14.2, the agent i knows that $e_1 = e_2 = 1$ without calculation.

When the two neighbors of i are not connected, there is no triangle closure formed; therefore, no redundant edge can be identified, as illustrated in Fig. 14.3(a).

Suppose that the two neighbors of i are linked in one direction, for example, i_1 can reach i_q, as depicted in Fig. 14.3(b, c, and d). For the connection topology in Fig. 14.3(b), no matter whether i can reach i_1 ($e_6 = 1$) or not ($e_6 = 0$), only one triangle closure exists in this local triangle topology, and it is for agent i. Therefore, (i, i_1) is labeled redundant by agent i. However, when both e_3 and e_5 appear in the graph, as depicted in Fig. 14.3(c), there are two triangle closures in the network, one for i, and the other for i_q. In the triangle closure for agent i, (i, i_1) is redundant assuming (i_q, i_1) is available, while in agent i_q's triangle closure, (i_q, i_1) is redundant assuming (i, i_1) is present. It can be seen that only one edge of (i, i_1) and (i_q, i_1) can be removed to maintain the reachability of this local network; otherwise vertex i_1 becomes isolated. Note that nodes i and i_1 make their decisions independently. To ensure that they achieve an agreement, edge robustness is used as a criterion for judgment: the less robust edge should be removed. For edges (i, i_1) and (i_q, i_1), longer edge length results in less robustness. Thus, for agent i, if (i, i_1) is shorter than (i_q, i_1), no redundant edge is reported; otherwise, agent i declares (i, i_1) to be redundant. In a trivial case when the two edge lengths are the same, the edge incident to a larger agent ID will be considered redundant. In other words, if $i > i_q$, then (i, i_1) is redundant. For the case when both e_5 and e_6 exist, as depicted in Fig. 14.3(d), we will postpone the discussion.

Now, consider the case in which agent i's two neighbors i_1 and i_q can reach each other ($e_3 = e_4 = 1$). When i cannot reach i_1 and i_q, as shown in Fig. 14.3(e), agent i possesses two triangle closures. The robustness of the two options, i.e., the robustness of the two-edge paths (i, i_1, i_q) and (i, i_q, i_1), will be compared by agent i to make a decision on redundancy.

Now, let us look at the last three types of topologies for these three nodes, as illustrated in Fig. 14.3(d, f, and g). A common feature of these three triangle topologies is that they all possess triangle cycles; in other words, the local graph of these three nodes is strongly connected. To preserve the connectedness, each agent needs to help maintain the directed cycle. Therefore, the agent i can only declare (i, i_1) to be redundant for Fig. 14.3(d) and (f). Figure 14.3(g) contains two triangle cycles; only one of them needs to be preserved. To guarantee a consensus among these three nodes, robustness values of the two directed cycles (i, i_1, i_q, i) and (i, i_q, i_1, i) will be used as a measure. For agent i, either the edge (i, i_1) or (i, i_q) must be redundant, depending on which cycle to keep.

In the case where two cycles have the same robustness, agent IDs will be used to reach unanimous decision. One approach, which is used in the algorithm below, is to keep the triangle cycle in the direction so that the agent IDs follow the ascending order in a circular way. An alternative solution for breaking triangle closures while maintaining reachability is to remove the two longest (least robust) edges of the same length. As a result, four edges will remain for the local graph of Fig. 14.3(g).

14.3.3 Distributed Algorithm

The rules described in Sect. 14.3.2 are applied in our distributed algorithm to each agent and each pair of the neighbors of the agent. Each agent makes independent decisions on determining redundant incoming edges, based on only the local information of its neighbors. When an agent marks an edge as redundant, it will ignore

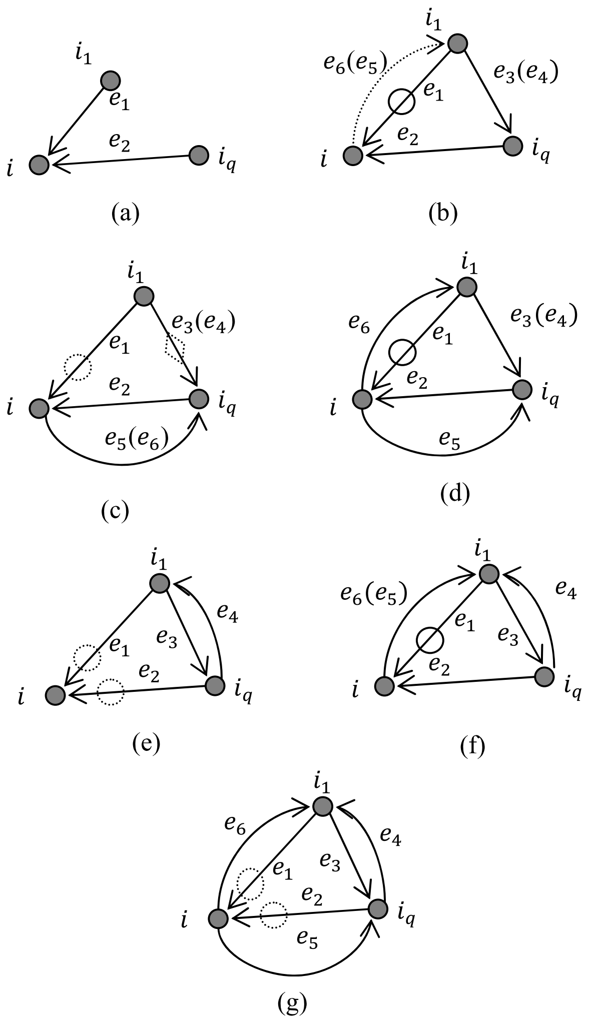


Fig. 14.3 Possible topologies for vertex i with its two neighbors i_1 and i_q. The *solid circles* in (**b**), (**d**), (**f**) indicate the edges that will be removed; the two *dashed circles* in (**e**) and (**g**) show that one of the edges will be removed depending on the robustness; and the *diamond* in (**c**) signifies the edge that will be used determine the robustness of the edge with a *circle* on. The parenthesis includes another possibility

the information coming through that link. Consequently, the edge corresponding to the link is dropped in the communication topology. Overall, the algorithm results in a group network with reduced connectivity.

Assume the neighbor set of agent i is $N_i = \{i_1, i_2, \ldots, i_p\}$ where $i_1, \ldots i_p \in \mathbb{I}$. Let the set of redundant edges of agent i be R_i with initial value $\emptyset$ (empty set). The algorithm works in two rough steps for any agent i. First, it sequentially scans through each pair of agent i's neighbors to examine the existence of triangle closures. When triangle closures are found, the algorithm determines redundant edges and adds them to R_i, based on the local communication topology between these three nodes. The algorithm given below applies at any vertex i in the network.

In the algorithm, for agent i, a set denoted as $\overline{N_i}$ is maintained to keep track of neighbors which wait for examination. As the algorithm is carried on, the neighbors that are incident to the identified redundant edges will be removed from $\overline{N}_i$ gradually, as well as the neighbors that have been examined. The algorithm halts when $\overline{N}_i$ contains only one neighbor. At the end of the algorithm, the set R_i keeps all the redundant edges that will be dropped by agent i. For simplicity, we define the following actions *A1˜A3* that are used in the algorithm:

Action 1 (*A1*): Repeat *Step 2* with $q = q + 1$;
Action 2 (*A2*): $R_i = R_i \cup \{(i, i_1)\}$ and go to *Step 3*;
Action 3 (*A3*): $R_i = R_i \cup \{(i, i_q)\}$ and repeat *Step 2* with $q = q + 1$, where *Step 2* and *Step 3* are given in the algorithm below.

Algorithm 4 (For agent i)
Step 1: Let $m = p$, $\overline{N}_i = N_i = \{i_1, i_2, \ldots, i_m\}$, and the set of the corresponding transmission range

$$\delta_i = \{\delta_{i_1}, \delta_{i_2}, \ldots, \delta_{i_m}\}$$

where $i_1, \ldots i_m \in \mathbb{I}$. Let $q = 2$, and $R_i = \emptyset$.
Step 2: If $q > m$, go to *Step 3*; otherwise, calculate the distance (denoted as d_3) between i_1 and i_q, and compute e_3 and e_4.
Cases:
(a) If $e_3 = e_4 = 0$, do *A1*.
(b) For $e_3 + e_4 = 1$, compute distances of i_1 and i_q from i, denoted as d_1 and d_2, and further get e_5 and e_6.

(b1) If $e_3 = 1, e_5 = 0$, do *A2*.
(b2) If $e_4 = 1, e_6 = 0$, do *A3*.
(b3) For $e_3 e_5 = 1$ and $e_6 = 0$, if $d_1 < d_3$, do *A1*; if $d_1 > d_3$, do *A2*; for $d_1 = d_3$, do *A1* if $i < i_q$, and do *A2* otherwise.
(b4) For $e_4 e_6 = 1$ and $e_5 = 0$, if $d_2 < d_3$, do *A1*; if $d_2 > d_3$, do *A3*; for $d_1 = d_3$, do *A1* if $i < i_1$, and do *A3* otherwise.
(b5) If $e_5 e_6 = 1$, do *A2* for $e_3 = 1$, and do *A3* for $e_4 = 1$.

(c) For $e_3 e_4 = 1$, compute d_1, d_2 and e_5, e_6.

(c1) For $e_5 = 1$ and $e_6 = 0$, do *A3*.
(c2) For $e_5 = 0$ and $e_6 = 1$, do *A2*.
(c3) For $e_5 = e_6 = 0$, compute the robustness values $r_1 := r(i, i_1, i_q)$ and $r_2 := r(i, i_q, i_1)$. If $r_1 > r_2$, do *A3*; if $r_1 < r_2$, do *A2*; for $r_1 = r_2$, do *A3* if $i_1 < i_q$ and do *A2* otherwise.
(c4) For $e_5 = e_6 = 1$, compute the robustness values $r_1 := r(i, i_1, i_q, i)$ and $r_2 := r(i, i_q, i_1, i)$. If $r_1 > r_2$, do *A3*; if $r_1 < r_2$, do *A2*; for $r_1 = r_2$, sort i, i_1, i_q in a ascending circular list, then, do *A3* if i_1 is right after i, and do *A2* if i_q is right after i.

Step 3: Let $\overline{N}_i = \overline{N}_i \backslash (R_i, \{i_1\})$, and reindex $\overline{N}_i = \{i_1, i_2, \ldots, i_m\}$, where $m = |\overline{N}_i|$. The algorithm terminates if $m = 1$; go to *Step 2* with $q = 2$ otherwise.

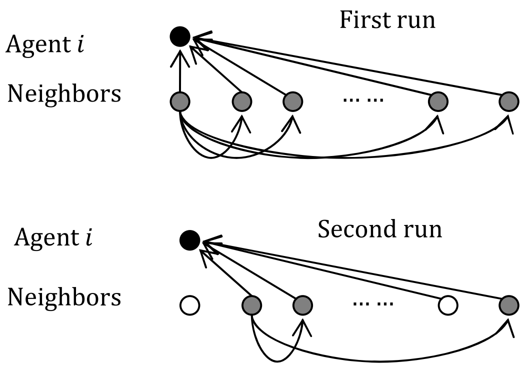


Fig. 14.4 Illustration of the algorithm iterations: *black dot* is agent i itself; *grey dots* are currently in the set $\overline{N}_i$ for redundancy check; and *white dots* have been removed from $\overline{N}_i$ as a result of the previous run

Step 1 of the algorithm is the initialization stage where $\overline{N}_i$ is initialized to contain all the neighbors of agent i. *Step 2* computes distances between i and its neighbors i_1, i_q in $\overline{N}_i$; checks the existence of triangle closures for agent i with these two neighbors; and determines redundant edges based on the rules described in Sect. 14.3.2. This step fixes one neighbor (i.e., i_1) and iterates through each one of the other neighbors (i.e, i_q) in $\overline{N}_i$, for $q = 2, 3, \dots, m$ until $q > m$ or (i, i_1) is redundant. This is referred to as *one run* in the paper, as illustrated in Fig. 14.4. At the end of each run, all the redundant edges are included in R_i, and, the neighbors incident to these edges and the node i_1 are removed from $\overline{N}_i$. *Step 3* checks the terminating condition (only one neighbor is left in $\overline{N}_i$) to see if *Step 2* (i.e., more runs) needs to be repeated.

The above algorithm will be performed on each agent in the network, and as a result, each agent obtains a set of redundant edges, i.e., R_i, $i = 1, \dots, n$. Note that all the edges in R_i have common terminal vertex i. For many applications, agent i could ignore the information received through these links to achieve the same task objective without significantly impairing system performance, while at the same time, saving computational power and time. Consequently, these links are dropped in the group topology of the network, resulting in a reduced graph of the initial communication topology.

Suppose that the communication graph of the network is $G(k) = (V, E)$ and the reduced group graph resulting from the above algorithm is $\hat{G}(k) = (V, \hat{E})$. We have the following relation between $G(k)$ and $\hat{G}(k)$. For each agent i, denote its in-degree in $G(k)$ as d_i and in $\hat{G}(k)$ as $\hat{d}_i$, then, $\hat{d}_i = d_i - |R_i|$. Group the redundant edges together as the set $R = \bigcup_{i=1,\dots,n} R_i$. Then, we have $\hat{E} = E \backslash R$.

14.4 Discussion and Simulation

Given the local triangle communication topology of three agents, it can be seen from the distributed algorithm that all the agents make unanimous decisions on deleting redundant edges without impairing the reachability relation of this local graph. However globally, the collective behavior of this algorithm guarantees to preserve

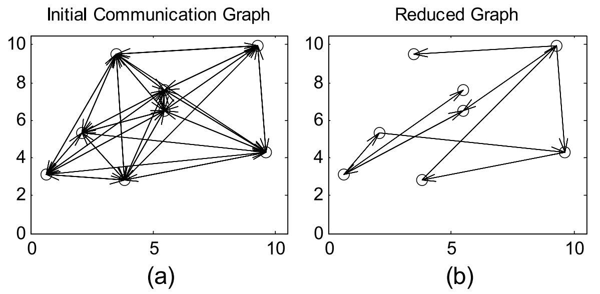


Fig. 14.5 An application of the distributed algorithm presented in the paper

the original reachability only under some additional conditions. This is illustrated in Fig. 14.5 with simulation results.

In the initial communication graph of Fig. 14.5(a), vertices are randomly deployed in the 2-dimensional Euclidean space, and the transmission range of each vertex is also randomly generated. The reduced graph resulting from the application of Algorithm 4 is shown in Fig. 14.5(b). It is observed that the reduced graph does not maintain reachability between all possible pairs of agents. Below we study a scenario where all reachability is preserved in the reduced network.

Suppose that the agents in the network agree on a common transmission range. In this case, each communication link becomes bidirectional. Then, whenever agent i receives information from agent j, agent j also receives information from i. Thus, the digraph $G(k)$ can be treated as an undirected graph. When Algorithm 4 applies to such networks, we have the following results.

Proposition 1 *Consider a multi-agent system (formulated as $G(k)$), in which all agents share the same transmission range. If $G(k)$ is completely-connected, the reduced graph $\hat{G}(k)$ resulting from Algorithm 4 is a directed cycle. In addition, the information flow of the cycle follows the ascending (or descending, with a corresponding change in the algorithm) order of the agent IDs.*

Proof Assume the index set (or IDs) of the agent is $\{1, \ldots, n\}$. Since $G(k)$ is completely connected, any agent i has $n-1$ neighbors in N_i. Further, the triangle topology of i with its any two neighbors, has the structure shown in Fig. 14.3(g) and the two cycles in the topology possess the same robustness. According to the algorithm, the cycle in the direction of ascending order of the agent IDs is reserved, after three edges are removed. Each time i picks two neighbors in N_i, one neighbor will be dropped. After exactly $n-2$ times such operations, only one neighbor remains at the termination of the algorithm, which is agent $i-1$ if $i > 1$ and n if $i = 1$.

In this way, each agent independently eliminates $n-2$ of its neighbors. Collectively, the information flow graph of the group is a directed cycle in the direction of $1 \rightarrow 2 \cdots \rightarrow n \rightarrow 1$. Changing the algorithm so that the triangle cycle in the other direction is preserved, we can have the cycle following the descending order of the agent IDs. □

Corollary 1 *Consider a multi-agent system in which each agent has the same transmission range, represented with a communication graph $G(k)$. If $G(k)$ is strongly connected, the reduced graph $\hat{G}(k)$ resulting from Algorithm 4 is also strongly connected.*

Proof The completely-connected bidirectional subgroups will be reduced to single directional cycles, and the rest of the edges connecting these subgroups remain bidirectional after reduction. Therefore, $\hat{G}(k)$ is still strongly connected. □

14.5 Conclusion

In this paper, we develop a distributed algorithm by which each agent selects its information sources (i.e., transmitting agents) based on only the local information of its neighbors. As a result, a reduced group graph is produced. In this graph, connectivity is reduced which allows agents to respond or maneuver with less computational effort and higher maneuverability. One direction in the future work will be to explore various applications with the aid of such reduced graphs. Other problems that need to be addressed in the future include more scenarios under which the proposed algorithm preserves the reachability of the original network.

References

1. Aho, A., Garey, M., Ullman, J.: The transitive reduction of a directed graph. SIAM J. Comput. **1**(2), 131–137 (1972)
2. Jadbabaie, A., Lin, J., Morse, A.S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Trans. Autom. Control **48**(9), 988–1001 (2003)
3. Moyles, D.M., Thompson, G.L.: An algorithm for finding a minimum equivalent graph of a digraph. J. Assoc. Comput. Mach. **16**(3), 455–460 (1969)
4. Spanos, D.P., Murray, R.M.: Robust connectivity of networked vehicles. In: 43rd IEEE Conference on Decision and Control, pp. 2893–2898 (2004)
5. Lafferriere, G., Williams, A., Caughman, J., Veerman, J.J.P.: Decentralized control of vehicle formations. Syst. Control Lett. **54**, 899–910 (2005)
6. Fiedler, M.: Algebraic connectivity of graphs. Czechoslov. Math. J. **23**(98), 298–305 (1973)
7. Ji, M., Egerstedt, M.: Distributed formation control while preserving connectedness. In: 45th IEEE Conference on Decision and Control, pp. 5962–5967 (2006)
8. Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi-agent systems. Proc. IEEE **95**(1), 215–233 (2007)
9. Khuller, S., Raghavachari, B., Young, N.: Approximating the minimum equivalent digraph. SIAM J. Comput. **24**(4), 859–872 (1995)
10. Ren, W., Beard, R.W.: Consensus seeking in multiagent systems under dynamically changing interaction topologies. IEEE Trans. Autom. Control **50**(5), 655–661 (2005)
11. Ren, W., Beard, R.W., Atkins, E.M.: A survey of consensus problems in multi-agent coordination. In: Proceedings of American Control Conference, pp. 1859–1864 (2005)
12. Kim, Y., Mesbahi, M.: On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian. IEEE Trans. Autom. Control 116–120 (2006)

Chapter 15
The Navigation Potential of Ground Feature Tracking

Meir Pachter and Güner Mutlu

Summary Navigation System aiding using bearing measurements of stationary ground features is investigated. The objective is to quantify the navigation information obtained by tracking ground features over time. The answer is provided by an analysis of the attendant observability problem. The degree of Inertial Navigation System aiding action is determined by the degree of observability provided by the measurement arrangement. The latter is strongly influenced by the nature of the available measurements—in our case, bearing measurements of stationary ground objects—the trajectory of the aircraft, and the length of the measurement interval. It is shown that when one known ground object is tracked, the observability Grammian is rank deficient and thus full Inertial Navigation System aiding action is not available. However, if baro altitude is available and an additional vertical gyroscope is used to provide an independent measurement of the aircraft's pitch angle, a data driven estimate of the complete navigation state can be obtained. If two ground features are simultaneously tracked the observability Grammian is full rank and all the components of the navigation state vector are positively impacted by the external measurements.

15.1 Introduction

Inertial Navigation System aiding using optical measurements [1–6] is appealing because passive bearings—only measurements preserve the autonomy of the integrated navigation system. In this paper, an attempt is made to gain an understanding of the nature of the navigation information provided by bearings—only measurements taken over time, of stationary ground objects, whose position is not necessarily known.

M. Pachter (✉) · G. Mutlu
Air Force Institute of Technology, AFIT, Wright Patterson AFB, OH 45433, USA
e-mail: meir.pachet@afit.edu

G. Mutlu
e-mail: guner.mutlu.tr@afit.edu

In this article, the crucial issue of processing the images provided by a down looking camera for the autonomous—without human intervention—measurement of optic flow, or, alternatively, image correspondence for feature tracking, or, mosaicing, [7, 8] is not addressed. We do however note that these tasks are nevertheless somewhat easier than full blown Autonomous Target Recognition (ATR) and/or machine perception. In this article it is assumed that autonomous feature detection and tracking is possible so that bearings measurement records of stationary ground objects are available. We exclusively focus on gauging the navigation potential of bearings—only measurements of stationary ground objects taken over time.

In this article, we refer to purely deterministic, i.e., noise free, corrupted bearing measurements, where many measurements would naturally help wash out the noise. Even so, both the number of bearing measurements taken and the number of simultaneously tracked ground objects is of interest. Obviously, increasing the number of tracked features should increase the navigation information content. Thus, our main thrust is focused on the quantification and measurement of the degree of observability of the optical bearings-only measurement arrangement. In this respect, we follow in the footsteps of [9] and [10]. The absolute minimal number of tracked features such that additional features do not further increase the degree of observability, is established.

Concerning observability: The measurement equations are time-dependent and therefore one cannot ascertain observability from an observability matrix. Hence, the information content of passive optical bearings-only measurements will be gauged by deriving the observability Grammian of the bearing-only measurement arrangement. It is however very important to first nondimensionalize the navigation state, the bearings measurement geometry, and also the time variable, so that the derived observability Grammian is nondimensional. This guarantees that the observability Grammian correctly reflects the geometry of the measurement arrangement, so that meaningful conclusions can be drawn. The rank, or the condition number of the observability Grammian is established—it is the ultimate purveyor of the navigation potential of vision.

A word of caution concerning information fusion is in order. It is often much too easy to first thing jump head first and set up a Kalman Filter (KF) for fusing, e.g., optical and inertial measurements—in which case one then refers to INS aiding using bearings-only measurements [11]. The point is that a KF output will always be available. Even in the extreme case of an optical measurement arrangement where observability is nonexistent, a KF can be constructed and a valid navigation state estimate output will be available, except that no aiding action is actually taking place: the produced navigation state estimate then exclusively hinges on prior information only—in this case, inertial measurements, whereas the optical measurements don't come into play.

In light of the above discussion, the real question is whether there is aiding action so that the optical measurements are actually brought to bear on a KF provided navigation state estimate, thus yielding enhanced estimation performance relative to a stand alone INS. One would like to determine how strong the aiding action is and into precisely which components of the navigation state the aiding action trickles down into. The answer is provided by our deterministic observability analysis.

A KF will help enhance the accuracy of the complete navigation state if, and only if, the system is observable, that is, the observability Grammian is full rank. We'll also address the case of partial observability.

Strictly speaking, the herein outlined analysis should be carried out whenever the use of a KF is contemplated, to get a better understanding of the data fusion process. For example, it is known that when during cruise, GPS position measurements are used to aid an INS, no aiding action will be realized in the critical aircraft heading navigation state variable, unless the aircraft is performing high-g horizontal turns. And in the extreme case of an object in free fall—for example, a bomb—it can be shown that GPS position measurements will not enhance the estimate of *any* of the object's Euler angles.

Naturally, the use of bearings-only measurements can yield information on angular components only of an air vehicle's navigation state. We refer to the measurement of the aircraft "drift" angles, namely, the angles included between the ground referenced velocity vector and the aircraft body axes. A dramatic improvement in the navigation state information can be obtained if additional passive measurements become available. A case in point is baro-altitude measurement. The latter is also used in inertial navigation—to the extent that high precision inertial navigation is not possible without passive baro altitude, or, GPS—provided, altitude information. Indeed, the combined use of optical measurements and baro-altitude for navigation has a long history, from the days when navigators where seated in the glazed nose of aircraft and would optically track a ground feature using a driftmeter, or, on the continent, a cinemoderivometer.

We are also interested in the use of additional passive measurements and side information. We refer to known landmarks, digital terrain elevation data, LADAR provided range measurements, GPS provided position measurements, and, finally, the mechanization of an autonomous navigation system akin to an INS which however exclusively and continuously uses bearing measurements of ground features—one would then refer to an Optical Navigation System (ONS).

By augmenting the navigation state with the coordinates of the tracked ground features/objects, Simultaneous Location and Mapping (SLAM) is achieved in a unified framework. Furthermore, the analysis can be extended to include the tracking of moving objects, whether on the ground, or in the air. Under standard kinematic assumptions, the motion of ownship relative to said tracked objects can be measured: think of obstacle avoidance or "see and avoid" guidance.

15.2 Modeling

We are cognizant of the fact that the degree of INS aiding provided by bearing measurements might strongly depend on the trajectory flown. For example, this certainly is the case when GPS position measurements are used for INS aiding. All this notwithstanding, in this paper we confine our attention to the most basic two-dimensional scenario where the aircraft is flying wings level at constant altitude in

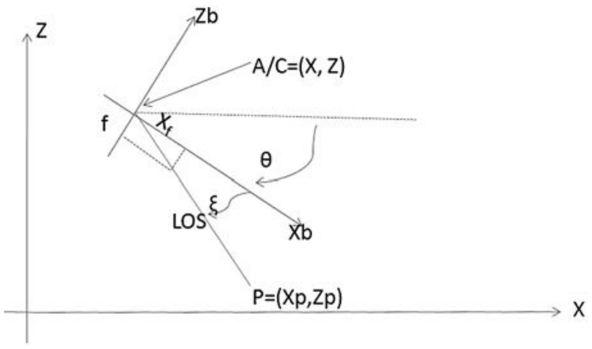


Fig. 15.1 Flight in the vertical plane

the vertical plane. The two-dimensional scenario under consideration is illustrated in Fig. 15.1.

In 2-D, the navigation state is

$$X = (x, z, v_x, v_z, \theta)$$

and the "disturbance"

$$d = (\delta f_x, \delta f_z, \delta\omega)$$

consists of the biases δf_x and δf_z of the accelerometers and the bias $\delta\omega$ of the rate gyro. The error equations are

$$\delta\dot{X} = A\delta X + \Gamma d \tag{15.1}$$

We assume the Earth is flat and nonrotating—this, in view of the short duration and the low speed of the Micro Air Vehicle (MAV) under consideration. Consequently, in level flight the dynamics are

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & g \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \Gamma = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{15.2}$$

The tracked ground object is located at $P = (x_p, z_p)$. From the 2-D geometry in Fig. 15.1, we derive the measurement equation:

$$\frac{x_p - x}{z - z_p} = \cot(\theta + \xi) \tag{15.3}$$

and thus the measurement

$$\tan(\xi) = \frac{z - z_p - (x_p - x)\tan\theta}{x_p - x + (z - z_p)\tan\theta} = \frac{f}{x_f}$$

or

$$x_f = \frac{x_p - x + (z - z_p)\tan\theta}{z - z_p - (x_p - x)\tan\theta}\, f \tag{15.4}$$

Now, the INS output $x_c = x + \delta x,\ z_c = z + \delta z, \theta_c = \theta + \delta\theta$ where x, z, and θ are the true navigation states. In order to linearize the measurement equation, we need the information x_p^-, z_p^-, that is, prior information on the position of the tracked ground feature.

Let

$$x_p^- = x_p + \delta x_p, \qquad z_p^- = z_p + \delta z_p$$

The true augmented state

$$x = x_c - \delta x, \qquad z = z_c - \delta z, \qquad v_x = v_{x_c} - \delta v_x, \qquad v_z = v_{z_c} - \delta v_z,$$
$$\theta = \theta_c - \delta\theta, \qquad x_p = x_p^- - \delta x_p, \qquad z_p = z_p^- - \delta z_p$$

where $\delta x, \delta z, \delta v_x, \delta v_z, \delta\theta, \delta x_p, \delta z_p$ are the augmented state's estimation errors.

During level flight, and using the small angle approximation, the measurement equation is

$$\frac{x_f}{f} = \frac{x_p - x + (z - z_p)\theta}{z - z_p - (x_p - x)\theta} \tag{15.5}$$

and the measurement, expressed as a function of the state perturbations δx, δz, $\delta\theta$, δx_p, δz_p is:

$$\begin{aligned}
\frac{x_f}{f} &= \frac{x_p^- - \delta x_p - x_c + \delta x + (z_c - \delta z - z_p^- + \delta z_p)\cdot(\theta_c - \delta\theta)}{z_c - \delta z - z_p^- + \delta z_p - (x_p^- - \delta x_p - x_c + \delta x)\cdot(\theta_c - \delta\theta)} \\
&= \frac{x_p^- - x_c + (z_c - z_p^-)\theta_c + \delta x - \theta_c\delta z + (z_p^- - z_c)\delta\theta - \delta x_p + \theta_c\delta z_p}{z_c - z_p^- + (x_c - x_p^-)\theta_c - \theta_c\delta x - \delta z + (x_p^- - x_c)\delta\theta + \theta_c\delta x_p + \delta z_p} \\
&= \frac{x_p^- - x_c + (z_c - z_p^-)\theta_c}{z_c - z_p^- + (x_c - x_p^-)\theta_c} + \frac{1}{[z_c - z_p^- + (x_c - x_p^-)\theta_c]^2} \\
&\quad \cdot \Big\{\big[z_c - z_p^- + (x_c - x_p^-)\theta_c + \big(x_p^- - x_c + (z_c - z_p^-)\theta_c\big)\theta_c\big]\delta x \\
&\quad + \big[-\big(z_c - z_p^- + (x_c - x_p^-)\theta_c\big)\theta_c + x_p^- - x_c + (z_c - z_p^-)\theta_c\big]\delta z \\
&\quad + \big[\big(z_c - z_p^- + (x_c - x_p^-)\theta_c\big)(z_p^- - z_c) \\
&\quad + \big(x_p^- - x_c + (z_c - z_p^-)\theta_c\big)(x_c - x_p^-)\big]\delta\theta \\
&\quad + \big[z_p^- - z_c + (x_p^- - x_c)\theta_c + \big(x_c - x_p^- + (z_p^- - z_c)\theta_c\big)\theta_c\big]\delta x_p \\
&\quad + \big[\big(z_c - z_p^- + (x_c - x_p^-)\theta_c\big)\theta_c + x_c - x_p^- + (z_p^- - z_c)\theta_c\big]\delta z_p\Big\}
\end{aligned}$$

$$= \frac{x_p^- - x_c + (z_c - z_p^-)\theta_c}{z_c - z_p^- + (x_c - x_p^-)\theta_c} + \frac{1}{[z_c - z_p^- + (x_c - x_p^-)\theta_c]^2}$$
$$\cdot \big\{\big[z_c - z_p^- + (z_c - z_p^-)\theta_c^2\big]\delta x + \big[x_p^- - x_c + (x_p^- - x_c)\theta_c^2\big]\delta z$$
$$- \big[(x_p^- - x_c)^2 + (z_p^- - z_c)^2\big]\delta\theta + \big[z_p^- - z_c + (z_p^- - z_c)\theta_c^2\big]\delta x_p$$
$$+ \big[x_c - x_p^- + (x_c - x_p^-)\theta_c^2\big]\delta z_p\big\}$$
$$= \frac{x_p^- - x_c + (z_c - z_p^-)\theta_c}{z_c - z_p^- + (x_c - x_p^-)\theta_c} + \frac{1}{[z_c - z_p^- + (x_c - x_p^-)\theta_c]^2}$$
$$\cdot \big[\big(1+\theta_c^2\big)(z_c - z_p^-)\delta x + \big(1+\theta_c^2\big)(x_p^- - x_c)\delta z$$
$$- \big[(x_p^- - x_c)^2 + (z_p^- - z_c)^2\big]\delta\theta + \big(1+\theta_c^2\big)(z_p^- - z_c)\delta x_p$$
$$+ \big(1+\theta_c^2\big)(x_c - x_p^-)\delta z_p\big]$$
$$= \frac{x_p^- - x_c + (z_c - z_p^-)\theta_c}{z_c - z_p^- + (x_c - x_p^-)\theta_c} + \frac{1+\theta_c^2}{[z_c - z_p^- + (x_c - x_p^-)\theta_c]^2}$$
$$\cdot \big\{(z_c - z_p^-)\delta x + (x_p^- - x_c)\delta z - \big[(x_p^- - x_c)^2 + (z_p^- - z_c)^2\big]\delta\theta$$
$$+ (z_p^- - z_c)\delta x_p + (x_c - x_p^-)\delta z_p\big\}$$

Hence, for level flight the linearized measurement equation is

$$\frac{x_f}{f} - \frac{x_p^- - x_c + (z_c - z_p^-)\theta_c}{z_c - z_p^- + (x_c - x_p^-)\theta_c}$$
$$= \frac{1}{[z_c - z_p^- + (x_c - x_p^-)\theta_c]^2} \cdot \big\{(z_c - z_p^-)\delta x + (x_p^- - x_c)\delta z - \big[(x_p^- - x_c)^2$$
$$+ (z_p^- - z_c)^2\big]\delta\theta + (z_p^- - z_c)\delta x_p + (x_c - x_p^-)\delta z_p\big\} \tag{15.6}$$

15.3 Special Cases

If the position (x_p, z_p) of the tracked ground object is *known*, the state is the navigation state x, z, v_x, v_z, θ and the linearized measurement equation is

$$\frac{x_f}{f} - \frac{x_p - x_c + (z_c - z_p)\theta_c}{z_c - z_p + (x_c - x_p)\theta_c}$$
$$= \frac{1}{[z_c - z_p + (x_c - x_p)\theta_c]^2}$$
$$\cdot \big\{(z_c - z_p)\delta x + (x_p - x_c)\delta z + \big[(x_p - x_c)^2 + (z_p - z_c)^2\big]\delta\theta\big\}$$

If only the *elevation* z_p of the tracked ground object is known, the augmented state is $x, z, v_x, v_z, \theta, x_p$. Without loss of generality, we set $z_p = 0$, and the linearized measurement equation is:

$$\frac{x_f}{f} - \frac{x_p^- - x_c + z_c\theta_c}{z_c + (x_c - x_p^-)\theta_c} = \frac{1}{[z_c + (x_c - x_p^-)\theta_c]^2} \cdot \big\{ z_c\delta x + (x_p^- - x_c)\delta z + \big[(x_p^- - x_c)^2 + z_c^2\big]\delta\theta - z_c\delta x_p \big\}$$

Remark For the purpose of analysis, on the right-hand side of the measurement equation one can replace x_c by the true—that is, the nominal—state component x, z_c by the true—that is, the nominal—state component z, and θ_c by $\theta \equiv 0$.

Hence, if the position of the tracked ground object is known, and, in addition, without loss of generality, we set $z_p = 0$, then the linearized measurement equation is

$$\frac{x_f}{f} - \frac{x_p - x_c + z_c\theta_c}{z_c + (x_c - x_p)\theta_c} = \frac{1}{z^2} \cdot \big\{ z\delta x + (x_p - x)\delta z - \big[(x_p - x)^2 + z^2\big]\delta\theta \big\} \tag{15.7}$$

If only the elevation z_p of the tracked ground object is known, and, as before without loss of generality, we set $z_p = 0$, then the state error is δx, δz, δv_x, δv_z, $\delta\theta$, δx_p, and the linearized measurement equation is

$$\frac{x_f}{f} - \frac{x_p^- - x_c + z\theta_c}{z_c + (x - x_p)\theta_c} = \frac{1}{z^2} \cdot \big\{ z\delta x + (x_p^- - x)\delta z - \big[(x_p^- - x)^2 + z^2\big]\delta\theta - z\delta x_p \big\}$$

If the position of the ground object is not known, the navigation state error is $\delta x, \delta z, \delta v_x, \delta v_z, \delta\theta, \delta x_p, \delta z_p$ and the linearized measurement equation—see, e.g., (15.6)—is

$$\frac{x_f}{f} - \frac{x_p^- - x_c + (z - z_p^-)\theta_c}{z - z_p^- + (x - x_p^-)\theta_c} = \frac{1}{(z - z_p^-)^2} \cdot \big\{ (z - z_p^-)\delta x + (x_p^- - x)\delta z - \big[(x_p^- - x)^2 + (z - z_p^-)^2\big]\delta\theta + (z - z_p^-)\delta x_p + (x - x_p^-)\delta z_p \big\}$$

Furthermore, for the purpose of analysis, on the right-hand side of the last two equations we can also replace x_p^- and z_p^- by the true position (x_p, z_p) of the tracked object. Hence, the respective measurement equations are

$$\frac{x_f}{f} - \frac{x_p^- - x_c + z\theta_c}{z_c + (x - x_p)\theta_c} = \frac{1}{z^2} \cdot \big\{ z\delta x + (x_p - x)\delta z - \big[(x_p - x)^2 + z^2\big]\delta\theta - z\delta x_p \big\} \tag{15.8}$$

and

$$\frac{x_f}{f} - \frac{x_p^- - x_c + (z - z_p^-)\theta_c}{z - z_p^- + (x - x_p^-)\theta_c}$$
$$= \frac{1}{(z - z_p)^2} \cdot \big\{(z - z_p)\delta x + (x_p - x)\delta z - \big[(x_p - x)^2 + (z - z_p)^2\big]\delta\theta$$
$$+ (z - z_p)\delta x_p + (x - x_p)\delta z_p\big\}$$

Furthermore, without loss of generality, in the last equation we set $z_p = 0$:

$$\frac{x_f}{f} - \frac{x_p^- - x_c + (z - z_p^-)\theta_c}{z - z_p^- + (x - x_p^-)\theta_c}$$
$$= \frac{1}{z^2} \cdot \big\{z\delta x + (x_p - x)\delta z - \big[(x_p - x)^2 + z^2\big]\delta\theta + z\delta x_p + (x - x_p)\delta z_p\big\}$$

During the initial geo-location phase where one is exclusively interested in the ground object's position, one takes the INS calculated aircraft navigation state variables x, z, θ at face value and the linearized measurement equation is used to estimate $\delta x_p, \delta z_p$:

$$\frac{x_f}{f} - \frac{x_p^- - x + (z - z_p^-)\theta}{z - z_p^- + (x - x_p^-)\theta} = \frac{1}{(z - z_p^-)^2} \cdot \big[(z_p^- - z)\delta x_p + (x - x_p^-)\delta z_p\big] \quad (15.9)$$

This equation would be used if a recursive geo-location algorithm is applied. Note however that if one is exclusively interested in geo-locating the ground object, a batch algorithm might be preferrable. Finally, one would use (15.9) and iterate

$$x_p^- := x_p^- - \delta x_p \quad (15.10)$$

$$z_p^- := z_p^- - \delta z_p \quad (15.11)$$

In practice, the first guesses of x_p^- and z_p^- are generated as follows. We have the INS provided "prior" information on the navigation state x, z, θ. Bearing measurements of the ground feature P are taken over time. One can use the first two bearing measurements to obtain the "prior" x_p^- and z_p^- information. Hence, initially one is exclusively concerned with the geo-location of the feature on the ground. To this end, one uses (15.3) and upon recording two measurements one obtains a set of two linear equations in the unknowns x_p^- and z_p^-:

$$x_p + \cot(\theta_{c_1} + \xi_{m_1})z_p = x_{c_1} + z_{c_1}\cot(\theta_{c_1} + \xi_{m_1}) \quad (15.12)$$

$$x_p + \cot(\theta_{c_2} + \xi_{m_2})z_p = x_{c_2} + z_{c_2}\cot(\theta_{c_2} + \xi_{m_2}) \quad (15.13)$$

whereupon

$$\begin{pmatrix} x_p^- \\ z_p^- \end{pmatrix} = \frac{1}{\cot(\theta_{c_2}+\xi_{m_2}) - \cot(\theta_{c_1}+\xi_{m_1})} \cdot \begin{bmatrix} \cot(\theta_{c_2}+\xi_{m_2}) & -\cot(\theta_{c_1}+\xi_{m_1}) \\ -1 & 1 \end{bmatrix}$$
$$\cdot \begin{pmatrix} x_{c_1} + z_{c_1}\cot(\theta_{c_1}+\xi_{m_1}) \\ x_{c_2} + z_{c_2}\cot(\theta_{c_2}+\xi_{m_2}) \end{pmatrix}$$

Obviously, one could use more than two bearing measurements and obtain an overdetermined linear system in x_p^- and z_p^-, but then one performs geo-location only and completely foregoes the task of INS aiding. Since at time instants $k = 1$ and $k = 2$ the INS is not aided, the original prior information on the navigation state error δx^-, δz^-, $\delta\theta^-$ must be propagated forward to time step $k = 2$. From this point on, the updated prior information δx^-, δz^-, $\delta\theta^-$ and the prior information x_p^- and z_p^- are employed to start the Kalman filter such that the aircraft navigation state and the ground object's position are simultaneously updated using the bearing measurements obtained at time $k = 3, 4, \ldots$. In conclusion: When unknown ground objects are tracked, they first must be geo-located. This delays the INS aiding action by at least two time steps. Furthermore, the aircraft's navigation state prior information must be propagated ahead two time steps, without it being updated with bearing measurements, while one exclusively relies on the INS. The measurement equation is then relinearized using the ground object's prior information obtained during the preliminary geo-location step, and the two time steps propagated ahead navigation state prior information. From this point on, the INS aiding action and the ground object's geo-location is simultaneously performed during the ground object's tracking interval.

If the ground object's elevation z_p is known, say from a digital terrain data-base, one can make do with one bearing measurement only:

$$x_p^- = x_c + (z_c - z_p)\cot(\theta_c + \xi_m) \tag{15.14}$$

15.4 Nondimensional Variables

The aircraft's nominal altitude is h and its nominal ground speed is v. Set

$$x \to \frac{x}{h}, \quad z \to \frac{z}{h}, \quad v_x \to \frac{v_x}{v}, \quad v_z \to \frac{v_z}{v}, \quad t \to \frac{tv}{h}, \quad T \to \frac{v}{h}T,$$
$$\delta f_x \to \frac{\delta f_x}{g}, \quad \delta f_z \to \frac{\delta f_z}{g}, \quad \delta\omega \to \frac{h}{v}\delta\omega$$

Introduce the nondimensional parameter ($\frac{1}{2}$ the ratio of the aircraft's potential energy to its kinetic energy):

$$a \equiv \frac{hg}{v^2}$$

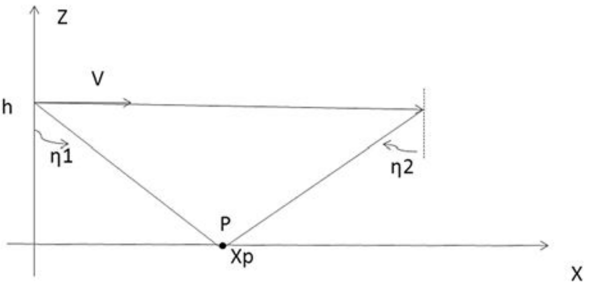


Fig. 15.2 Simple measurement scenario

Then the aircraft's nondimensional navigation state error dynamics are specified by the matrices

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & a \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \Gamma = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{15.15}$$

The scenario considered is wings level flight in the vertical plane (x, z).

For example, for a MAV, $v = 20\ [\frac{\mathrm{m}}{\mathrm{sec}}]$, $h = 40$ [m], and $g = 10\ [\frac{\mathrm{m}}{\mathrm{sec}^2}] \Rightarrow a = 1$.

The geometry of the measurement scenario is characterized by the two nondimensional parameters

$$\alpha_1 = \tan \eta_1$$

$$\alpha_2 = \tan \eta_2$$

Consequently, the nondimensional measurement interval

$$T = \tan \eta_1 + \tan \eta_2$$

Consider first a symmetric measurement scenario where $\eta_1 = \eta_2 = \eta$. Since $x = vt$, $x_p = h \tan \eta$, using nondimensional variables, we obtain from the measurement equation (15.9)

$$\begin{aligned} y &= \delta x + (\tan \eta - t)\delta z - \left[1 + (\tan \eta - t)^2\right]\delta\theta, \\ &0 \le t \le 2 \tan \eta \end{aligned} \tag{15.16}$$

i.e., the measurement matrix

$$C = 1, \left[\tan \eta - t, 0, 0, 2t \tan \eta - t^2 - \frac{1}{\cos^2 \eta}\right] \tag{15.17}$$

the measurement

$$y \equiv \frac{x_f}{f} - \frac{x_p - x_c + z_c\theta_c}{z_c + (x_c - x_p)\theta_c} \tag{15.18}$$

and the measurement interval

$$T = \frac{h}{v}(\tan\eta_1 + \tan\eta_2)$$

Note that, as is often the case in INS aiding, the measurement matrix C is time dependent.

15.5 Observability

The observability Grammian [12] is

$$W(T) = \int_0^T e^{A^T t} C^T(t) C(t) e^{At} dt \tag{15.19}$$

where T is the measurement interval.

We calculate

$$e^{At} = \begin{bmatrix} 1 & 0 & t & 0 & \frac{1}{2}at^2 \\ 0 & 1 & 0 & t & 0 \\ 0 & 0 & 1 & 0 & at \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{15.20}$$

The geometry of the symmetric measurement arrangement is specified by the nondimensional parameter $\alpha \equiv \tan\eta$. Then the nondimensional observation interval $T = 2\alpha$ and the measurement matrix

$$C(t) = \left[\left(1, \alpha - t, 0, 0, -1 - (\alpha - t)^2\right)\right] \tag{15.21}$$

We calculate

$$C(t)e^{At} = \left[1, \alpha - t, t, (\alpha - t)t, \frac{1}{2}at^2 - 1 - (\alpha - t)^2\right] \tag{15.22}$$

and

$$e^{A^T t} C^T(t) C(t) e^{At}$$
$$= \begin{bmatrix} 1 & \alpha - t & t & (\alpha - t)t & f(t) \\ \alpha - t & (\alpha - t)^2 & (\alpha - t)t & (\alpha - t)^2 t & (\alpha - t)f(t) \\ t & (\alpha - t)t & t^2 & (\alpha - t)t^2 & tf(t) \\ (\alpha - t)t & (\alpha - t)^2 t & (\alpha - t)t^2 & (\alpha - t)^2 t^2 & (\alpha - t)tf(t) \\ f(t) & (\alpha - t)f(t) & tf(t) & t(\alpha - t)f(t) & f^2(t) \end{bmatrix} \tag{15.23}$$

where $f(t) \equiv \frac{1}{2}at^2 - 1 - (a - t)^2$. We integrate the time-dependent entries of the matrix in (15.23) and obtain the observability Grammian elements

$$W_{1,1} = 2\alpha, \qquad W_{1,2} = 0, \qquad W_{1,3} = 2\alpha^2, \qquad W_{1,4} = -\frac{2}{3}\alpha^3,$$

$$W_{1,5} = \frac{4}{3}a\alpha^3 - 2\alpha - \frac{2}{3}\alpha^3,$$

$$W_{2,2} = \frac{2}{3}\alpha^3, \qquad W_{2,3} = -\frac{2}{3}\alpha^3, \qquad W_{2,4} = \frac{2}{3}\alpha^4,$$

$$W_{2,5} = 2\alpha^2 - 2\alpha^3 - \frac{2}{3}a\alpha^4,$$

$$W_{3,3} = \frac{8}{3}\alpha^4, \qquad W_{3,4} = -\frac{4}{3}\alpha^4, \qquad W_{3,5} = 2a\alpha^4 - \frac{2}{3}\alpha^4 - 2\alpha^2,$$

$$W_{4,4} = \frac{16}{15}\alpha^5, \qquad W_{4,5} = \frac{2}{5}\alpha^5 - \frac{6}{5}a\alpha^5 + \frac{2}{3}\alpha^4,$$

$$W_{5,5} = \left(\frac{2}{5} - \frac{16}{15}a + \frac{8}{5}a^2\right)\alpha^5 + \frac{4}{3}(1-2a)\alpha^3 + 2\alpha$$

Consider the MAV scenario where $a = 1$ and the measurement scenario shown in the Fig. 15.2, where $\alpha = 1$

The observability Grammian is then

$$W = \frac{2}{15}\begin{bmatrix} 15 & 0 & 15 & -5 & -10 \\ 0 & 5 & -5 & 5 & -5 \\ 15 & -5 & 20 & -10 & -5 \\ -5 & 5 & -10 & 8 & -1 \\ -10 & -5 & -5 & -1 & 12 \end{bmatrix} \tag{15.24}$$

The 5×5 real, symmetric, positive, semi-definite matrix W is not full rank: $\text{Rank}(W) = 3$. This implies: no observability.

Next, consider the alternative measurement geometry where the tracked ground feature P is at the origin ($x_p = 0$). Then $\eta = 0$ so that $\alpha = 0$ and

$$e^{A^T t}C^T(t)C(t)e^{At} = \begin{bmatrix} 1 & -t & t & -t^2 & f(t) \\ -t & t^2 & -t^2 & t^3 & -f(t)\cdot t \\ t & -t^2 & t^2 & -t^3 & f(t)\cdot t \\ -t^2 & t^3 & -t^3 & t^4 & -f(t)\cdot t^2 \\ f(t) & -f(t)\cdot t & f(t)\cdot t & -f(t)\cdot t^2 & 2f^2(t) \end{bmatrix}$$

where $f(t) = \frac{1}{2}(a-1)t^2 - 1$.

The nondimensional measurement interval $T = 2$, and the elements of the observability Grammian are

$$W_{1,1} = 2, \qquad W_{1,2} = -2, \qquad W_{1,3} = 2, \qquad W_{1,4} = -\frac{8}{3}, \qquad W_{1,5} = \frac{2}{3}(2a-7),$$

$$W_{2,2} = \frac{8}{3}, \qquad W_{2,3} = -\frac{8}{3}, \qquad W_{2,4} = 4, \qquad W_{2,5} = 2(3-a),$$

$$W_{3,3} = \frac{8}{3}, \qquad W_{3,4} = -4, \qquad W_{3,5} = 2(a-3),$$

$$W_{4,4} = \frac{32}{5}, \qquad W_{4,5} = \frac{8}{15}(17 - 6a),$$

$$W_{5,5} = \frac{32}{5}\left(\frac{1}{2}a - 1\right)^2 + \frac{8}{3}(2 - a) + 2$$

As before, assume $a = 1$. The observability Grammian is

$$W = \frac{2}{15}\begin{bmatrix} 15 & -15 & 15 & -20 & -25 \\ -15 & 20 & -20 & 30 & 30 \\ 15 & -20 & 20 & -30 & -30 \\ -20 & 30 & -30 & 48 & 44 \\ -25 & 30 & -30 & 44 & 47 \end{bmatrix} \tag{15.25}$$

The matrix W has two identical columns (columns 2 and 3) and two identical rows, rows 2 and 3. This implies $\text{Rank}(W) = 3$.

When both features are tracked, the observation matrix is the 2×5 matrix

$$C(t) = \begin{bmatrix} 1 & 1-t & 0 & 0 & 2t - t^2 - 2 \\ 1 & -t & 0 & 0 & -1 - t^2 \end{bmatrix} \tag{15.26}$$

As before, for $a = 1$, we obtain

$$e^{At} = \begin{bmatrix} 1 & 0 & t & 0 & \frac{1}{2}t^2 \\ 0 & 1 & 0 & t & 0 \\ 0 & 0 & 1 & 0 & t \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{15.27}$$

and we calculate

$$C(t)e^{At} = \begin{bmatrix} 1 & 1-t & t & t - t^2 & 2t - \frac{1}{2}t^2 - 2 \\ 1 & -t & t & -t^2 & -\frac{1}{2}t^2 - 1 \end{bmatrix} \tag{15.28}$$

and

$$e^{A^T t}C^T(t)C(t)e^{At} =$$

$$\begin{bmatrix} 2 & 1-2t & 2t & t-2t^2 & 2t - t^3 - 3 \\ 1-2t & 1+2t^2-2t & t-2t^2 & t-2t^2+2t^3 & t^3 - \frac{5}{2}t^2 + 5t - 2 \\ 2t & t-2t^2 & 2t^2 & t^2-2t^3 & 2t^2 - t^3 - 3t \\ t-2t^2 & t-2t^2+2t^3 & t^2-2t^3 & 2t^4-2t^3+t^2 & t^4 - \frac{5}{2}t^3 + 5t^2 - 2t \\ 2t-t^2-3 & t^3 - \frac{5}{2}t^2 + 5t - 2 & 2t^2 - t^3 - 3t & t^4 - \frac{5}{2}t^3 + 5t^2 - 2t & \frac{1}{2}t^4 - 2t^3 + 7t^2 - 8t + 5 \end{bmatrix}$$

Integration yields the entries of the observability Grammian

$$W_{1,1} = 30, \quad W_{1,2} = -15, \quad W_{1,3} = 30, \quad W_{1,4} = -25, \quad W_{1,5} = -35,$$

$$W_{2,2} = 25, \quad W_{2,3} = -25, \quad W_{2,4} = 35, \quad W_{2,5} = 25,$$

$$W_{3,3} = 40, \qquad W_{3,4} = -40, \qquad W_{3,5} = -35,$$
$$W_{4,4} = 56, \qquad W_{4,5} = 43,$$
$$W_{5,5} = 59$$

The matrix W is full rank. Hence, we have observability.

Concerning a shortcut: A sufficient condition for observability is $\text{Rank}(W) = 5$, where $W = W_{p_1} + W_{p_2}$ and W_{p_i} is the observability Grammian when ground object i is tracked, $i = 1, 2$; this is indeed the case.

15.6 Only the Elevation z_p of the Tracked Ground Object is Known

The augmented state's error is $(\delta x, \delta z, \delta v_x, \delta v_z, \delta\theta, \delta x_p)^T \in \Re^6$. Set $x_p \to \frac{x_p}{h}$.

The augmented system's dynamics are specified by

$$A_a := \begin{bmatrix} A & \vdots & 0 \\ \dots & \vdots & \dots \\ 0 & \vdots & 0 \end{bmatrix}_{6\times 6}, \qquad \Gamma_a := \begin{bmatrix} \Gamma \\ \cdots \\ 0 \end{bmatrix}_{6\times 3} \tag{15.29}$$

$$C_a(t) := \big(C(t), -1\big) \tag{15.30}$$

where, recall, for wings level, constant altitude flight,

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & a \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \Gamma = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{15.31}$$

and

$$C(t) = \begin{bmatrix} 1, & \alpha - t, & 0, & 0, & 2\alpha t - t^2 - 1 - \alpha^2 \end{bmatrix} \tag{15.32}$$

We calculate

$$\begin{aligned} e^{A_a t} &:= \begin{bmatrix} e^{At} & \vdots & 0 \\ \dots & \vdots & \dots \\ 0 & \vdots & 1 \end{bmatrix} \\ C_a e^{A_a t} &= \big(Ce^{At}, -1\big) \\ e^{A^T t} C_a^T(t) C_a(t) e^{A_a t} &= \begin{bmatrix} e^{A^T t} C^T C e^{At} & -e^{A^T} C^T \\ -Ce^{At} & 1 \end{bmatrix} \end{aligned} \tag{15.33}$$

and

$$C(t)e^{At} = \left(1, \alpha - t, t, \alpha t - t^2, \frac{1}{2}at^2 - 1 - \alpha^2 - t^2 + 2\alpha t\right)$$

Let

$$w^T \equiv \int_0^{2\alpha} C(t)e^{At}dt = \left(2\alpha, 0, 2\alpha^2, -\frac{2}{3}\alpha^3, \frac{2}{3}(2a-1)\alpha^3 - 2\alpha\right)$$
$$W_a = \begin{pmatrix} W & -w \\ -w^T & 2\alpha \end{pmatrix}$$

When $a = 1$,

$$w^T = \left(2\alpha, 0, 2\alpha^2, -\frac{2}{3}\alpha^3, \frac{2}{3}\alpha^3 - 2\alpha\right)$$

and for $\alpha = 1$,

$$w^T = \left(2, 0, 2, -\frac{2}{3}, -\frac{4}{3}\right)$$

When $\alpha = 0$,

$$w^T = \left(2, -2, 2, -\frac{8}{3}, -\frac{10}{3}\right)$$

Hence, when $a = 1$ and $\alpha = 1$,

$$W_1 = \frac{2}{15}\begin{bmatrix} 15 & 0 & 15 & -5 & -10 & -15 \\ 0 & 5 & -5 & 5 & -5 & 0 \\ 15 & -5 & 20 & -10 & -5 & -15 \\ -5 & 5 & -10 & 8 & -1 & 5 \\ -10 & -5 & -5 & -1 & 12 & 10 \\ -15 & 0 & -15 & 5 & 10 & 30 \end{bmatrix} \tag{15.34}$$

When $a = 1$ and $\alpha = 0$

$$W_2 = \frac{2}{15}\begin{bmatrix} 15 & -15 & 15 & -20 & -25 & -15 \\ -15 & 20 & -20 & 30 & 30 & 15 \\ 15 & -20 & 20 & -30 & -30 & -15 \\ -20 & 30 & -30 & 48 & 44 & 20 \\ -25 & 30 & -30 & 44 & 47 & 25 \\ -15 & 15 & -15 & 20 & 25 & 15 \end{bmatrix} \tag{15.35}$$

$\text{Rank}(W_1) = 4$ and $\text{Rank}(W_2) = 3$. $\text{Rank}(W_1 + W_2) = 6$.

15.7 Partial Observability

When the observability Grammian W is rank deficient, that is,

$$\text{Rank}(W) = r < n$$

where n is the state space dimension, the system is not observable; we then refer to the system as being partially observable. Strictly speaking, the aiding action of bearing-only measurements does not percolate into all the state components. One is thus interested in determining which states are (positively) impacted by the aiding action. The answer is provided by the Singular Value Decomposition (SVD) of the observability Garammian matrix W. The following holds.

The $n \times n$ real symmetric positive semi-definite matrix W can be factored as

$$W = HK$$

where H is a $n \times r$ matrix and K is a $r \times n$ matrix. Furthermore, $\text{Rank}(H) = r$ and $\text{Rank}(K) = r$; in other words, this is a full rank factorization.

From the SVD, we conclude that the measurement process allows us to estimate the parameter $\theta \in \Re^r$:

$$\theta = \left(H^T H\right)^{-1} H^T \int_0^T e^{A^T t} C^T(t) y(t)\, dt$$

The latter is related to the navigation state X as follows.

$$\theta = K X_0 \tag{15.36}$$

The navigation information provided by the bearing measurements is encapsulated in (15.36). The complete *initial* state X_0 cannot be calculated from the measurement record $y(t)$, $0 \le t \le T$ and in order to obtain a data driven estimate of the full state vector, $n - r$ additional independent measurements of the navigation state are needed. In 2-D, and when the position of the ground object is known, the dimension of the state, $n = 5$. When one known ground feature is tracked, $r = 3$. This tells us that two additional independent measurements of the navigation state are needed. The availability of the passively measured baro altitude immediately comes to mind, so that one additional independent measurement is needed for establishing the navigation state. The latter could be the aircraft's pitch angle θ, which is independently provided by a vertical gyroscope.

15.8 Conclusion

In this paper, the simplest 2-D scenario of INS aiding using bearing measurements of stationary ground features is investigated. The measurements are taken over time and the attendant observability problem is formulated and analyzed. The degree of

INS aiding action is determined by the degree of observability provided by the measurement arrangement. The latter is strongly influenced by the nature of the available measurements—in our case, bearing measurements of stationary ground objects—the trajectory of the aircraft, and the length of the measurement interval. Whereas observability guarantees that all the navigation state's components are positively affected by the external measurements, we are also interested in the possibility of partial observability where not all the navigation state components' estimates are impacted by the external measurements. It is shown that when one known ground object is tracked, the observability Grammian is rank deficient and thus full INS aiding action is not available. However, if baro altitude is available and an additional vertical gyroscope is used to provide an independent measurement of the aircraft's pitch angle, a data driven estimate of the complete navigation state can be obtained. If two ground features are simultaneously tracked the observability Grammian is full rank and all the components of the navigation state vector are positively impacted by the external measurements.

References

1. Pacher, M.: INS aiding by tracking an unknown ground object—theory. In: Proceedings of the 2003 American Control Conference, Denver, CO, June 4–6, 2003
2. Pacher, M., Polat, M.: Bearing—only measurements for INS aiding: theory for the three dimensional case. In: Proceedings of the 2003 AIAA Guidance, Navigation and Control Conference, Austin, Texas, August 11–14, 2003, AIAA paper No. 2003-5354
3. Pacher, M.: Optical flow for INS aiding: the 3D case. In: Proceedings of the 44th Israel Conference on Aerospace Sciences, Tel Aviv, Israel, February 25–26, 2004
4. Pacher, M.: Bearing—only measurements for INS aiding: the 3D case. In: Proceedings of the American Control Conference, Boston, MA, June 30–July 2, 2004
5. Pacher, M.: INS aiding using bearing—only measurements of known ground object. In: Proceedings of the 17th IFAC Symposium on Automatic Control in Aerospace, June 25–29, 2007, Tulouse, France
6. Pacher, M., Porter, A., Polat, M.: INS aiding using bearing—only measurements of an unknown ground object. ION J. Navig. **53**(1), 1–20 (2006)
7. Veth, M., Raquet, J., Pacher, M.: Stochastic constraints for fast image correspondance search with uncertain terrain model. IEEE Trans. Aerosp. Electron. Syst. **42**(3), 973–982 (2006)
8. Veth, M., Raquet, J., Pacher, M.: Correspondance search mitigation using feature space antialiasing. In: Proceedings of the 63rd Annual Meeting of the Institute of Navigation, Cambridge, MA, April 23–25, 2007
9. Bar-Itzhack, I.Y., Berman, N.: Control theoretic approach to inertial navigation systems. J. Guid. Control Dyn. **11**(3), 237–245 (1988)
10. Rhee, I., Abdel-Hafez, M.F., Speyer, J.L.: Observability of an integrated GPS/INS during maneuvers. IEEE Trans. Aerosp. Electron. Syst. **40**(2), 526–534 (2004)
11. Nielsen, M., Raquet, J., Pacher, M.: Development and flight test of a robust optical–inertial navigation system using low cost sensors. In: ION GNNS 2008, Savannah, GA, September 16–19, 2008
12. Brockett, R.: Finite Dimensional Linear Systems. Wiley, New York (1970)

Chapter 16
Minimal Switching Time of Agent Formations with Collision Avoidance

Dalila B.M.M. Fontes
and Fernando A.C.C. Fontes

Summary We address the problem of dynamically switching the topology of a formation of a number of undistinguishable agents. Given the current and the final topologies, each with n agents, there are $n!$ possible allocations between the initial and final positions of the agents. Given the agents maximum velocities, there is still a degree of freedom in the trajectories that might be used in order to avoid collisions. We seek an allocation and corresponding agent trajectories minimizing the maximum time required by all agents to reach the final topology, avoiding collisions. Collision avoidance is guaranteed through an appropriate choice of trajectories, which might have consequences in the choice of an optimal permutation. We propose here a dynamic programming approach to optimally solve problems of small dimension. We report computational results for problems involving formations with up to 12 agents.

16.1 Introduction

We consider the problem of cooperation among a collection of vehicles performing shared tasks using vehicles formation to coordinate their actions. In this work, we consider a formation of autonomous undistinguishable agents that, for some external reason, has to reconfigure the geometry of its formation. The problem we address is, given the information of the current and final formation geometry as well as each agent velocity, to decide which agents are allocated to the new positions in the formation, guaranteeing that there are no collisions in the transition process. In

D.B.M.M. Fontes (✉)
LIAAD—INESC Porto L.A. and Faculdade de Economia, Universidade do Porto,
Rua Dr. Roberto Frias, 4200-464 Porto, Portugal
e-mail: fontes@fep.up.pt

F.A.C.C. Fontes
ISR Porto and Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias,
4200-465 Porto, Portugal
e-mail: faf@fe.up.pt

particular, we seek the agent-target allocation that minimizes the time required for all agents to assume their new position.

Recent technological advances, such as the growth in computation and communication power and the advent of miniaturization technologies have boosted the interest in vehicles which can interact autonomously with the environment and other vehicles to perform tasks beyond the ability of individual vehicles. Research in coordination and control of teams of several agents (that may be robots, ground, air or underwater vehicles) has been growing fast in the past few years. The main reason behind such growth is the wide range of military and civil applications where such teams can be used efficiently. Application areas include unmanned aerial vehicles (UAVs) [4, 17], autonomous underwater vehicles (AUVs) [15], automated highway systems (AHSs) [3, 16], and mobile robotics [19, 20]. While each of these application areas poses its own unique challenges, several common threads can be found. In most cases, the vehicles are coupled through the task they are trying to accomplish, but are otherwise dynamically decoupled, meaning the motion of one does not directly affect the others. For a recent survey in cooperative control of multiple vehicles systems, see for example the work by Murray [10].

Team formation is a common and critical task in many cooperative agent applications, since shape formation may be considered as the starting point for a team of agents to perform cooperative tasks. Also, formation switching or reconfiguration arises in a variety of applications due to the need to adapt to environmental changes or to new tasks, possibly determined by the accomplished ones. The cooperative behavior we focus on in this paper is formation switching with collision avoidance.

Possible applications arise from reactive formation switching or reconfiguration of a team of autonomous agents, when performing tasks such as surveillance, patrol, intruder detection, containment of chemical spills, forest fires, etc. For example, when a team of agents that moves in formation through a trajectory has to change to another formation to avoid obstacles and then change back to the original formation. Figure 16.1 depicts an example when reorganization is needed since the formation has to go through a narrow passage.

Some application examples are the following. Consider that a team of agents is performing surveillance using a specific formation and detects some intrusion. In such event it should change to another formation more appropriate to the new task at hand. This new formation may, or may not, be a predefined or structured formation. An example of a nonpredefined case is described in [18], where the formation mission involves containment of chemical spillage. The agents task, which initially is monitoring, after detection occurs becomes to determine the perimeter of the spill. Another type of application requiring switching within specific formations happens, for example, when a robot soccer team loses the ball. In such even, the team has to switch from an attack formation to a defense formation with a different geometry more appropriate to the new task.

In a previous work [6], we have addressed the problem of formation switching, that is the problem of determining the actions that have to be taken by each individual agent so that the overall group moves into a specific formation. Among the possible actions to reorganize the formation into a new desired geometry, we would

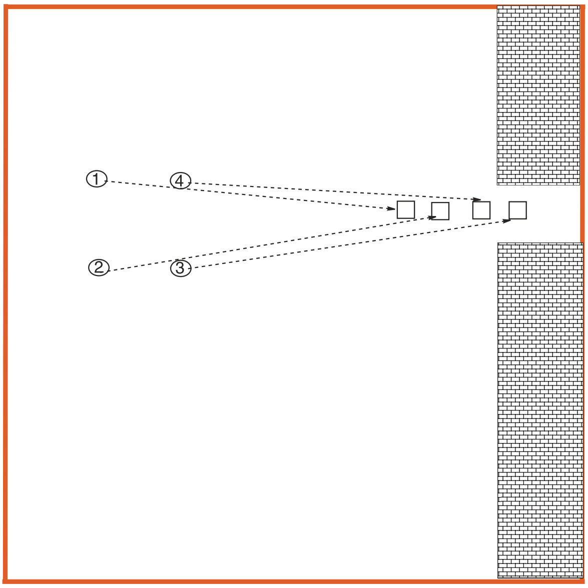


Fig. 16.1 Reconfiguration of a formation to avoid obstacles

found the ones that optimize any predefined additive performance measure, e.g., minimal distance traveled by all agents in the formation.

Here however, we also wish to guarantee that the actions found are collision free and our objective is to minimize the overall time required for all agents to assume their new positions. Therefore, given the current vehicles formation, i.e., current positions, and the positions they are supposed to assume, we wish to determine where which vehicle should go to such that the overall formation switching time is minimal, while at the same time enforcing collision avoidance. While we realize the importance of dynamic analysis of the control trajectory of each agent, our focus here is on the static optimization problem of deciding *where* each agent should go to rather than *how* it should go there. The problem addressed here should be seen as a component of a framework for multiagent coordination, incorporating also the trajectory control component [7], that allows to maintain or change formation while following a specified path in order to perform cooperative tasks.

Research on the static optimal formation switching problem is not abundant, although it has been addressed by some authors. Desai et al., in [5], model mobile robots formation as a graph. The authors use the so-called "control graphs" to represent the possible solutions for formation switching. In this method, for a graph having n vertices there are $n!(n-1)!/2^{n-1}$ control graphs, and switching can only happen between predefined formations. The authors do not address collision issues.

Hu and Sastry [8] study the problems of optimal collision avoidance and optimal formation switching for multiple agents on a Riemannian manifold. It is assumed that the underlying manifold admits a group of isometries, with respect to which the Lagrangian function is invariant. A reduction method is used to derive optimality conditions for the solutions.

In [18], Yamagishi describes a decentralized controller for the reactive formation switching of a team of autonomous mobile robots. The focus is on how a structured formation of agents can reorganize into a nonrigid formation based on changes in the environment. The controller utilizes nearest-neighbor artificial potentials (social rules) for collision-free formation maintenance and environmental changes act as a stimulus for switching between formations.

A similar problem, where a set of agents must perform a fixed number of different tasks on a set of targets, has been addressed by several authors. The methods developed include exhaustive enumeration (see Rasmussen et al. [12]), branch-and-bound (see Rasmussen and Shima [11]), network models (see Schumacher et al. [13, 14]), and dynamic programming (see Jin et al. [9]).

We propose a dynamic programming approach to solve the problem of formation switching with collision avoidance, that is the problem of deciding which agent moves to which place in the next formation guaranteeing that at any time the distance between any two of them is at least some predefined value. Conditions spotted during the execution of the current task are used to determine the following tasks and therefore formation, at least partially. The formation switching performance is given by the time required for all agents to reach their new position, which is given by the maximum traveling time amongst individual agent traveling times. Since we want to minimize the time required for all agents to reach their new position, we have to solve a minmax problem. However, the methodology we propose can be used with any separable performance function. Since dynamic programming is the main tool used in the algorithm, the computational times grow, as expected, exponentially with the problem dimension. Here, we report computational results for problems involving formations of up to 12 agents. Naturally, the maximum dimension that is possible to address by the proposed methodology depends on several factors, namely, whether the solution is to be computed ahead of time or in real time. In the latter case, we also have to take into account the time constants of the application at hand, as well as the available memory dimension and computational power.

This paper is organized as follows. In Sect. 16.2, the problem of optimal reorganization of agent formations with collision avoidance is described and a dynamic programming formulation of such problem is given and discussed. In Sect. 16.4, we discuss computational implementation issues of the dynamic programming algorithm, namely an efficient implementation of the main recursion as well as efficient data representations. A detailed description of the algorithms is also provided. Computational experiments are reported in Sect. 16.5, where the computational implementation options are analyzed and justified. Also, some examples are explored. Some conclusions are drawn in Sect. 16.5.

16.2 Problem Definition

In our problem, a team of n identical agents has to switch from their current formation to some other formation (i.e., agents have a specific goal configuration not related to the positions of the others), possibly unstructured, with collision avoidance. To address collision avoidance, we impose that the trajectories of the agents must satisfy the separation constraint that at any time the distance between any two of them is at least ϵ, for some positive ϵ. The optimal (joint) trajectories are the ones that minimize the maximum trajectory time of individual agents.

Our approach can be used either centralized or decentralized, depending on the agents capabilities. In the latter case, all the agents would have to run the algorithm, which outputs an optimal solution, always the same if many exist, since the proposed method is deterministic.

Regarding the new formation, it can be either a prespecified formation or a formation to be defined according to the information collected by the agents. In both cases, we do a pre-processing analysis that allows us to come up with the desired locations for the next formation.

This problem can be restated as the problem of allocating to each new position exactly one of the agents, located in the old positions. From all the possible solutions, we are only interested in the ones where collision is prevented. Among these, we want to find one that minimizes the time required for all agents to move to the target positions, that is an allocation which has the least maximum individual agent traveling time.

Consider that there are N agents to be relocated. Each agent $i \in I$ with $I = \{1, 2, \dots, N\}$ has associated a velocity value v_i and a vector containing the initial location coordinates (e.g., a triplet (x_i, y_i, z_i) if we consider the agent to move in a 3D space). Consider also a set J indexing the M possible target locations (again a triplet (x_j, y_j, z_j) is associated to each target $j \in J$). Let d_{ij} denote the distance of the predefined trajectory.

The problem can be formulated as follows:

$$\text{Min}\left\{\max\left[\sum_j t_{1j}\xi_{1j}, \sum_j t_{2j}\xi_{2j}, \dots, \sum_j t_{Nj}\xi_{Nj}\right]\right\}$$

subject to

$$\sum_i \xi_{ij} = 1 \quad \forall j \in J$$

$$\sum_j \xi_{ij} = 1 \quad \forall i \in I$$

$$\xi_{ij} \cdot \xi_{ab} \cdot c(i, j, a, b) = 0 \quad \forall i \in I,\ \forall j \in J,\ \forall a \in I \setminus \{i\},\ \forall b \in J \setminus \{j\}$$

$$\xi_{ij} \in \{0, 1\} \quad \forall i \in I, \forall j \in J$$

Here, the decision variable ξ takes the values

$$\xi_{ij} = \begin{cases} 1 & \text{if agent } i \text{ is to travel to target } j, \\ 0 & \text{otherwise} \end{cases}$$

and the variable t_{ij}, the time it takes for agent i to travel to position j, is defined as d_{ij}/v_i. The function c describes whether there is collision between the trajectories of agents i and a, traveling to positions j and b, respectively. This function takes the values

$$c(i, j, a, b) = \begin{cases} 1 & \text{if the trajectory from } i \text{ to } j\text{, intersects the trajectory from } a \text{ to } b\text{,} \\ 0 & \text{otherwise} \end{cases}$$

We say that the trajectories intersect if at any instant of time the position of the agents get closer than a predefined distance. The precise definition of when collision occurs will be discussed in the next section.

16.3 Dynamic Programming Formulation

Dynamic Programming (DP) is an effective method to solve combinatorial problems of a sequential nature. It provides a framework for decomposing an optimization problem into a nested family of subproblems. This nested structure suggests a recursive approach for solving the original problem using the solution to some subproblems. The recursion expresses an intuitive *principle of optimality* for sequential decision processes; that is, once we have reached a particular state, a necessary condition for optimality is that the remaining decisions must be chosen optimally with respect to that state. Richard Bellman coined the terms *dynamic programming* and *principle of optimality* and pioneered the development of the theory and applications. These seminal results are reported in his monograph [2].

16.3.1 Derivation of the Dynamic Programming Recursion

In this section, a DP formulation for the minimal switching time of agent formations with collision avoidance is proposed. In the derivation of the DP formulation that follows, no reference is made to the conditions that must be satisfied in order to guarantee collision avoidance. This is introduced after the assignment recursions have been explained. Let us for now consider that somehow this is being guaranteed. In the recursions, we will make use of the notation $a \vee b$ to denote the maximum value between a and b.

We want to allocate exactly one of the agents to each position in the new formation, guaranteeing that there are no agent collisions. In our model, a stage i contains all states S such that $|S| \geq i$, meaning that i agents have been allocated to the targets in S. The DP model has N stages, with a transition occurring from a stage $i-1$ to a stage i, when a decision is made about the allocation of agent i.

Define $f(i, S)$ to be the best allocation of agents $1, 2, \ldots, i$ to i targets in set S, that is as the allocation requiring the least maximum time the agents take to go to their new positions. Such value is found by determining the least maximum agent traveling time between its current position and its target position. For each agent, i, the traveling time to the target position j is given by d_{ij}/v_i. By definition, the minimum traveling time of the $i-1$ agents to the target positions in set $S \setminus \{j\}$ without

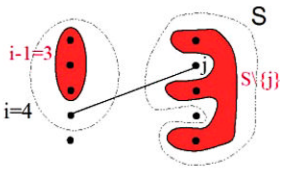


Fig. 16.2 An example of a possible decision at stage 4 of the DP Recursion

collisions is given by $f(i-1, S\setminus\{j\})$. From the above, the minimum traveling time of all i agents to the target positions in S they are assigned to, given that agent i travels at velocity v_i, without agent collisions, is obtained by examining all possible target locations $j \in S$ (see Fig. 16.2).

The *dynamic programming recursion* is then defined as

$$f(i,S) = \min_{j\in S}\big\{d_{ij}/v_i \vee f\big(i-1, S\setminus\{j\}\big) \vee M * \mathcal{C}\big(i,j,i-1,S\setminus\{j\}\big)\big\} \qquad (16.1)$$

where M is a very high positive number and $\mathcal{C}$ is the collision function that takes the value 1 if collision occurs under some conditions, which are to be defined later.

The *initial conditions* for the above recursion are provided by

$$f(1,S) = \min_{j\in S}\{d_{1j}/v_1\}, \quad \forall S \subseteq J \qquad (16.2)$$

and all other states are initialized as not yet computed.

Hence, the optimal value for the performance measure, that is the minimum traveling time needed for all N agents to assume their new positions in J, is given by

$$f(N,J) \qquad (16.3)$$

16.3.2 Collision Avoidance

Let us now discuss the collision constraints. Collisions may occur between two (or more) agents when they travel from their current position to the target positions they are to be in. Therefore, each time an agent-target assignment is made we must check if the agent traveling through this newly defined trajectory is at any time closer than a predefined allowed distance to any of agent trajectories already defined. In order to do so, we analyze the collision conditions between pairs of agents as follows.

Consider the following pair of agents starting their trajectory at the same time: agent i which travels at velocity v_i to position j that is reached in time T_i and agent a which travels at velocity v_a to position b that is reached in time T_a. Collision between these two agents occurs if the following condition is satisfied (see Fig. 16.3)

$$\left\| (x_i,y_i) + v_i \frac{(x_j,y_j)-(x_i,y_i)}{\|(x_j,y_j)-(x_i,y_i)\|} t - \left[(x_a,y_a) + v_a \frac{(x_b,y_b)-(x_a,y_a)}{\|(x_b,y_b)-(x_a,y_a)\|} t \right] \right\| < \epsilon \qquad (16.4)$$

for some $t \in [0, \min\{T_i, T_a\}]$.

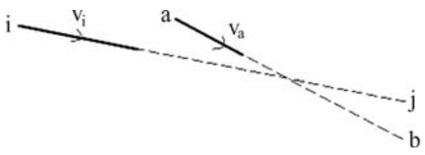


Fig. 16.3 Collision condition between two agents

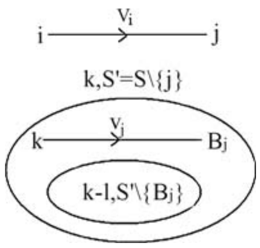


Fig. 16.4 Data involved in the collision recursion

We define the function $c(i, j, a, b)$ that takes value 1 if the condition in (16.4) is met and value 0 otherwise. To analyze if the agent traveling through a newly defined trajectory collides with any agent traveling through previously determined trajectories, we define a recursive function. This function checks the satisfaction of the separation constraint (or collision condition), given by (16.4), in turn, between the agent which had the trajectory defined last and each of the agents for which trajectory decisions have already been made.

Consider that we are in state (i, S) and that we are assigning agent i to target j. Further let v_{i-1} be the traveling velocity for agent $i - 1$. Since we are solving state (i, S) we need state $(i - 1, S \setminus \{j\})$, which has already been computed. (If this is not the case, then we must compute it first.) In order to find out if this new assignment is possible, we need to check if at any point in time agent i will collide with any of the agents $1, 2, \ldots, i - 1$, for which we have already determined the target assignment.

Let us define a recursive function $\mathcal{C}(i, j, k, S)$ that assumes the value one if a collision occurs between agent i traveling to j and any of the agents $1, 2, \ldots k$, with $k < i$, traveling to their targets, in set S, and assumes the value zero if no such collisions occurs. This function works in the following way (see Fig. 16.4):

1. First it checks $c(i, j, k, \mathcal{B}_j)$, that is, it verifies if there is collision between trajectory $i \to j$ at velocity v_i and trajectory $k \to \mathcal{B}_j$ at velocity v_k, where $\mathcal{B}_j$ is the optimal target for agent k when targets in set $S \setminus \{j\}$ are available for agents $1, 2, \ldots, k$. If this is the case, it returns the value 1.
2. Otherwise, if they do not collide, it verifies if trajectory $i \to j$ at velocity v_i collides with any of the remaining agents. That is, it calls the collision function $\mathcal{C}(i, j, k - 1, S')$, where $S' = S \setminus \{\mathcal{B}_j\}$.

The collision recursion is therefore written as:

$$\mathcal{C}(i, j, k, S) = c(i, j, k, \mathcal{B}_j) \vee \mathcal{C}\big(i, j, k - 1, S \setminus \{\mathcal{B}_j\}\big) \tag{16.5}$$

where

$$\mathcal{B}_j = Best_j(k, v_k, S)$$

The initial conditions for recursion (16.5) are provided by

$$\mathcal{C}(2, j, 1, S) = \min_{i \in S} \{c(2, j, 1, i)\}, \quad \forall j \in J, \ S \subseteq J \setminus \{j\} \tag{16.6}$$

and all other states are initialized as not yet computed.

16.4 Computational Implementation

A pure forward Dynamic Programming (DP) algorithm is easily derived from the DP recursion, (16.1) and (16.2). Such implementation may result in considerable waste of computational effort since, generally, complete computation of the state space is not required. Furthermore, since the computation of a state requires information contained in other states, rapid access to state information should be sought.

The DP procedure we have implemented exploits the recursive nature of the DP formulation by using a backward–forward procedure. Its main advantage is that the exploration of the state space graph, i.e., the solution space, is based upon the part of the graph which has already been explored. Thus, states which are not feasible for the problem are not computed, since only states which are needed for the computation of a solution are considered. The algorithm is dynamic as it detects the needs of the particular problem and behaves accordingly.

States at stage 1 are either nonexistent or initialized as given in (16.2). The DP recursion, (16.1), is then implemented in a backward-forward recursive way. It starts from the final state (N, J) and while moving backward visits, without computing, possible states until a state already computed is reached. Initially, only states in stage 1, initialized by (16.2) are already computed. Then, the procedure is performed in reverse order, i.e., starting from the state last identified in the backward process, it goes forward through computed states until a state (i, S') is found which has not yet been computed. At this point, again it goes backward until a computed state is reached. This procedure is repeated until the final state (N, J) is reached with a value that cannot be improved by any other alternative solution. The main advantage of this backward-forward recursive algorithm is that only intermediate states needed are visited and from these only the feasible ones that may yield a better solution are computed. As will be shown later, the number of states computed using this method is very small. For our test problems it varies between 25% (for the smallest problems) and 8.33% (for the largest problems) of the state space.

As said before, due to the recursive nature of (16.1), state computation implies frequent access to other states. Recall that a state is represented by a number and a set. Therefore, set operations like searching, deletion, and insertion of a set element must be performed efficiently. A computationally efficient way of storing and operating sets is the bit-vector representation, also called the boolean array, whereby a *computer word* is used to keep the information related to the elements of the set.

In this representation a universal set $U = \{1, 2, \ldots, n\}$ is considered. Any subset of U can be represented by a binary string (a computer word) of length n in which the i-th bit is set to 1 if i is an element of the set, and set to 0 otherwise. So, there is a one-to-one correspondence between all possible subsets of U (in total 2^n) and all binary strings of length n. Since there is also a one-to-one correspondence between binary strings and integers, the sets can be efficiently stored and worked out simply as integer numbers. A major advantage of such implementation is that the set operations, *location*, *insertion*, or *deletion* of a set element can be performed by directly addressing the appropriate bit. For a detailed discussion of this representation of sets see, for example, the book by Aho et al. [1].

The flow of the algorithm is managed by Algorithm 1, which starts by labeling all states (subproblems) as not yet computed, assigning to them the value infinity. Then, it initializes states in stage 1, that is subproblems involving 1 agent, as given by (16.2). After that, it calls Algorithms 2 and 4 in turn with parameters (N, J). Algorithm 2, that implements recursion (16.1), calls Algorithm 3 every time it attempts to define one more agent-target allocation. This algorithm is used to find out whether the newly established allocation satisfies the collision regarding all previously defined allocation or not, feeding the result back to Algorithm 2. Algorithm 4, also implements a recursive function with which the solution structure, i.e., agent-target allocation is retrieved.

Algorithm 1: DP for finding agent-target allocations.

```
1 Read agents set, locations and velocities; the
  target set and locations; and the distance
  functions;
2 Compute the distance for every pair agent-target
  ($d_{ij}$);
3 Label all states as not yet computed
      $f(N,S)=\infty$  for all $N=1,2,\ldots,n$ and $S \in J$;      (16.7)
4 Initialize states at stage one as
      $f(1,S)=\min_{j\in S}\{d_{1j}/v_1\}, \quad \forall S \subseteq J.$      (16.8)
5 Call $Compute(N, J)$;
6 Call $Allocation(N, J)$;
```

Algorithm 2 is a recursive algorithm that computes the optimal solution cost, i.e., it implements (16.1). This function receives two arguments: the agents to be allocated and the set of target locations available to them, both represented by integer numbers (the latter using the bit-vector representation, as discussed previously). It starts by checking whether the specific state (i, S) has already been computed or not. If so the program returns to the point where the function was called, otherwise the state is computed. To compute state (i, S), all possible target locations $j \in S$ that might lead to a better subproblem solution are identified. The function is then called with arguments $(i-1, S')$, where $S' = S \setminus \{j\}$, for every j such that allocating agent i to target j does not lead to any collision with previously defined allocations. This condition is verified by Algorithm 3.

Algorithm 2: Recursive function: compute optimal performance.

```
1  Recursive $Compute(i, S)$;

2  if $f(i, S) \neq \infty$ then
3      return $f(i, S)$ to caller;
4  end

5  Set $min = \infty$;

6  for $each\ j \in S'$ do
7      $S' = S \setminus \{j\}$;
8      Call $Collision(i, j, i-1, S')$
9      if $Col(i, j, i-1, S') = 0$ then
10         Call $Compute(i-1, S')$;
11         $t_{ij} = d_{ij}/v_i$;
12         $aux = max(f(i-1, S'), t_{ij})$;
13         if $aux \leq min$ then
14             $min = aux$; $best_j = j$;
15         end
16     end
17 end

18 Store information:
19     $target$   $\mathcal{B}_j(i, S) = best_j$;
20     $value$    $f(i, S) = min$;

21 Return: $f(i, S)$;
```

Algorithm 3 is a recursive algorithm that checks the collision of a specific agent-target allocation with a set of allocations previously established, i.e., it implements (16.5). This function receives four arguments: the newly defined agent-target allocation $i \rightarrow j$ and the previously defined allocations to check with, that is agents $1, 2, \ldots, k$ and their target locations S. It starts by checking the collision condition, given by (16.4), for the allocation pair $i \to j$ and $k \to \mathcal{B}_j$, where $\mathcal{B}_j$ is the optimal target for agent k when agents $1, 2, \ldots, k$ are allocated to targets in S. If there is collision it returns 1; otherwise it calls itself with arguments $(i, j, k-1, S \setminus \{\mathcal{B}_j\})$.

Algorithm 3: Recursive function: find if the allocation $i \to j$ collides with any of the existing allocations.

```
1  **Recursive** $Collision(i, j, k, S)$;
2  **if** $Col(i, j, k, S) \neq \infty$ **then**
3      return $Col(i, j, k, S)$ to caller;
4  **end**
5  $B_j = \mathcal{B}_j(k, S)$;
6  **if** *collision condition is not satisfied* **then**
7      $Col(i, j, k, S) = 1$;
8      return $Col(i, j, k, S)$ to caller;
9  **end**
10 $S' = S \setminus \{B_j\}$;
11 Call $Collision(i, j, k-1, S')$;
12 Store information: $Col(i, j, k, S) = 0$;
13 **Return:** $Col(i, j, k, S)$;
```

Algorithm 4 is also a recursive algorithm and it backtracks through the information stored while solving subproblems, in order to retrieve the solution structure, i.e., the actual agent-target allocation. This algorithm works backward from the final state (N, J), corresponding to the optimal solution obtained, and finds the partition by looking at the target stored for this state $\mathcal{B}_j(N, J)$, with which it can build the structure of the solution found. Algorithm 4 also receives two arguments: the agents and the set of target locations. It starts by checking whether the agent current locations set is empty. If so, the program returns to the point where the function was called; Otherwise the backtrack information of the state is retrieved and the other needed states evaluated.

At the end of the algorithm, $f(I, J)$ gives the performance associated with the best agent-target allocation.

Algorithm 4: Recursive function: retrieve agent-target allocation.

```
1 Recursive $Allocation(i, S)$;
2 if $S \neq \varnothing$ then
3     $j = target\mathcal{B}_j(i, S)$;
4     $Alloc(i) = j$;
5     $S' = S \setminus \{j\}$;
6     Call $Allocation(i-1, S')$;
7 end
8 Return: $Alloc$;
```

Table 16.1 Agents random initial location

	Location	
	x_i	y_i
Agent 1	30	113
Agent 2	183	64
Agent 3	348	349
Agent 4	30	200

16.5 Computational Experiments

In this section, we report on the computational experiments performed. We consider several standard structured formations for a team of agents: line, column, square, diamond, and wedge. We also considered nonrigid formations, which are randomly generated. The agents initial positions were randomly generated and are fixed for all experiments. In our experiments, we have decided to use d_{ij} as the Euclidian distance although any other distance measure may have been used. All agents are considered to travel at the same velocity v. Therefore, traveling times for each possible agent-target allocation are computed as $t_{ij} = d_{ij}/v$. The separation constraints impose, at any point in time, the distance between any two agent trajectories to be at least 10 points; otherwise, it is considered that those two agents collide.

Let us first show how a solution is recomputed in order to guarantee collision avoidance, using a small example. Consider 4 agents A, B, C, and D, positioned as given in Table 16.1 and four target positions 1, 2, 3, and 4 as in Table 16.2.

In Figs. 16.5 and 16.6, we give the graphical representation of the optimal agent-target allocation found, when collisions are allowed and no collisions are allowed, respectively. Since the separation constraints are active, the time required for all agents to travel to the targets they have been assigned to is larger when they are enforced. This value goes up from 4.4197 units of time when collisions are allowed to 4.4702 when they are not.

Table 16.2 Target random locations

	Location	
	x_i	y_i
Target 1	95	258
Target 2	147	78
Target 3	248	362
Target 4	293	228

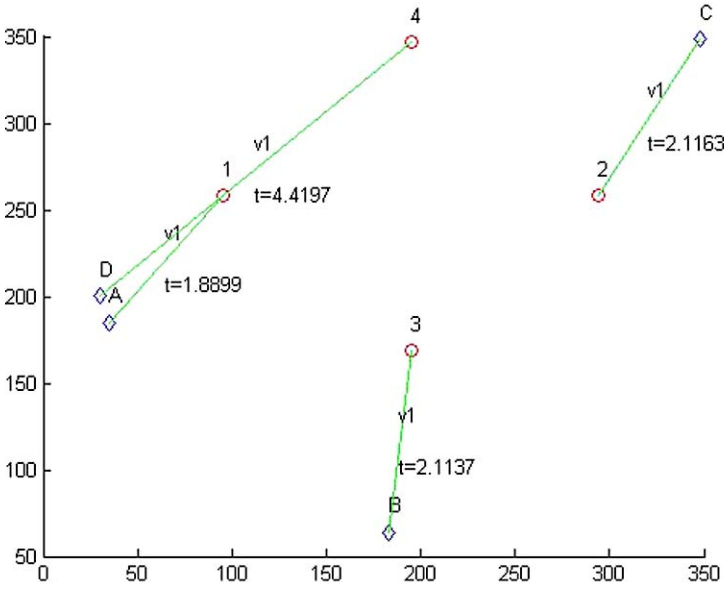


Fig. 16.5 Collisions allowed

The algorithm was implemented in Matlab and all examples run in between 0.034 seconds and 13.117 seconds. We note that the Matlab functions were interpreted and not compiled. In the case of faster solutions being required, the code could be compiled. As it can be seen, in the last two columns of Table 16.3, the total percentage of the state space used is very small and decreases with problem size. The full representation of the state space comprises $n \times (2^n - 1)$ states. However, in our implementation just a small percentage of the state space is computed. This value is 25% for the smallest problems and comes down to just over 8% for the largest ones. The efficiency of the proposed method can also be seen from the ratio between the number of subproblems used and the number of potential feasible solutions for the problem ($n!$). This value comes down from 62.50% to less than 0.001%.

The benefits of our particular implementation can be seen both from (1) the fact that computation of an optimal solution is very fast; and (2) from the fact that only a small part of the states is actually computed, either when this number is compared

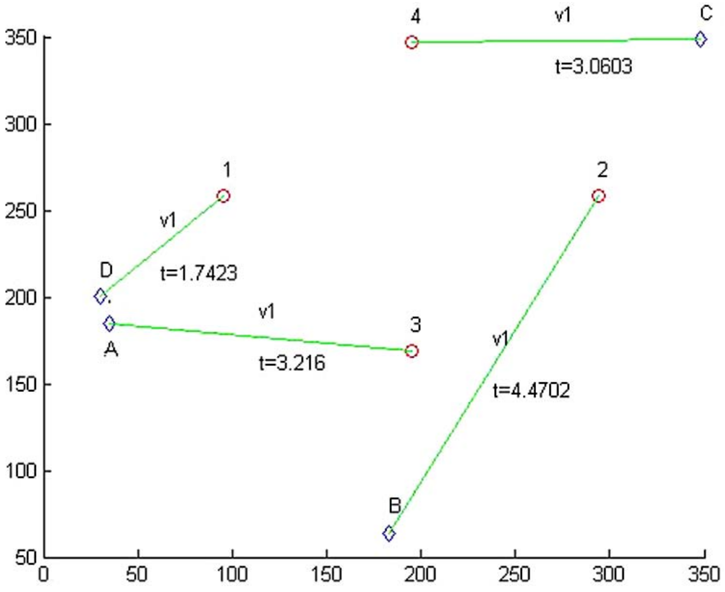


Fig. 16.6 No collisions allowed

Table 16.3 Running time and efficiency of the computational implementation

Number of agents	Running time (s)	Used states, percentage of	
		State space	Feasible sol.
4	0.034	25.00	62.50
5	0.064	20.00	25.83
6	0.109	16.67	8.75
7	0.212	14.28	2.52
8	0.519	12.50	0.63
9	1.181	11.11	0.14
10	2.706	10.00	0.03
11	5.947	9.09	0.01
12	13.117	8.33	8E−04

to the total number of states in our representation, or when it is compared to the number of possible feasible solutions.

16.6 Conclusion

We have developed an optimization algorithm to decide how to reorganize a formation of vehicles into another formation of different shape with collision avoidance, which is a relevant problem in cooperative control applications. The method proposed here should be seen as a component of a framework for multiagent coordination/cooperation, which must necessarily include other components such as a trajectory control component.

The algorithm proposed is based on a dynamic programming approach that is very efficient for small dimensional problems. As explained before, the original problem is solved by combining, in an efficient way, the solution to some subproblems. The method efficiency improves with the number of times the subproblems are reused, which obviously increases with the number of feasible solutions. This can be seen from the very small percentage of states use when compared with the number of states represented or the number of potential feasible solutions.

Moreover, the proposed methodology is very flexible, in the sense that it easily allows for the inclusion of additional problem features, e.g., imposing geometric constraints on each agent or on the formation as a whole, deciding on the agents velocity, among others.

References

1. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data Structures and Algorithms. Addison-Wesley, Reading (1983)
2. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton (1957)
3. Bender, J.: An overview of systems studies of automated highway systems. IEEE Trans. Veh. Technol. **40**(1 Part 2), 82–99 (1991)
4. Buzogany, L.E., Pachter, M., d'Azzo, J.J.: Automated control of aircraft in formation flight. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 1349–1370, Monterey, California (1993)
5. Desai, J.P., Ostrowski, P., Kumar, V.: Modeling and control of formations of nonholonomic mobile robots. IEEE Trans. Robot. Autom. **17**(6), 905–908 (2001)
6. Fontes, D.B.M.M., Fontes, F.A.C.C.: Optimal reorganization of agent formations. WSEAS Trans. Syst. Control **3**(9), 789–798 (2008)
7. Fontes, F.A.C.C., Fontes, D.B.M.M., Caldeira, A.C.D.: Model predictive control of vehicle formations. In: Pardalos, P., Hirsch, M.J., Commander, C.W., Murphey, R. (eds.) Optimization and Cooperative Control Strategies. Lecture Notes in Control and Information Sciences, vol. 381. Springer, Berlin (2009). ISBN:978-3-540-88062-2
8. Hu, J., Sastry, S.: Optimal collision avoidance and formation switching on Riemannian manifolds. In: IEEE Conference on Decision and Control, vol. 2, pp. 1071–1076. IEEE Press, New York (1998)
9. Jin, Z., Shima, T., Schumacher, C.J.: Optimal scheduling for refueling multiple autonomous aerial vehicles. IEEE Trans. Robot. **22**(4), 682–693 (2006)
10. Murray, R.M.: Recent research in cooperative control multivehicle systems. J. Dyn. Syst. Meas. Control **129**, 571–583 (2007)
11. Rasmussen, S.J., Shima, T.: Branch and bound tree search for assigning cooperating UVAs to multiple tasks. In: Institute of Electrical and Electronic Engineers, American Control Conference 2006, Minneapolis, Minnesota, USA (2006)
12. Rasmussen, S.J., Shima, T., Mitchell, J.W., Sparks, A., Chandler, P.R.: State-space search for improved autonomous UAVs assignment algorithm. In: IEEE Conference on Decision and Control, Paradise Island, Bahamas (2004)
13. Schumacher, C.J., Chandler, P.R., Rasmussen, S.J.: Task allocation for wide area search munitions via iterative network flow. In: American Institute of Aeronautics and Astronautics, Guidance, Navigation, and Control Conference 2002, Reston, Virginia, USA (2002)

14. Schumacher, C.J., Chandler, P.R., Rasmussen, S.J.: Task allocation for wide area search munitions with variable path length. In: Institute of Electrical and Electronic Engineers, American Control Conference 2003, New York, USA (2003)
15. Smith, T.R., Hanssmann, H., Leonard, N.E.: Orientation control of multiple underwater vehicles with symmetry-breaking potentials. In: IEEE Conference on Decision and Control, vol. 5, pp. 4598–4603 (2001)
16. Swaroop, D., Hedrick, J.K.: Constant spacing strategies for platooning in automated highway systems. J. Dyn. Syst. Meas. Control **121**, 462 (1999)
17. Wolfe, J.D., Chichka, D.F., Speyer, J.L.: Decentralized controllers for unmanned aerial vehicle formation flight. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 96–3833, San Diego, California (1996)
18. Yamagishi, M.: Social rules for reactive formation switching. Technical Report UWEETR-2004-0025, Department of Electrical Engineering, University of Washington, Seattle, Washington, USA (2004)
19. Yamaguchi, H.: A cooperative hunting behavior by mobile-robot troops. Int. J. Robot. Res. **18**(9), 931 (1999)
20. Yamaguchi, H., Arai, T., Beni, G.: A distributed control scheme for multiple robotic vehicles to make group formations. Robot. Auton. Syst. **36**(4), 125–147 (2001)

Chapter 17
A Moving Horizon Estimator Performance Bound

Nicholas R. Gans and Jess W. Curtis

Summary Moving Horizon implementations of the Kalman Filter are widely used to overcome weaknesses of the Kalman Filter, or in problems when the Kalman Filter is not suitable. While these moving horizon approaches often perform well, it is of interest to encapsulate the loss in performance that comes when terms in the Kalman Filter are ignored. This paper introduces two methods to calculate a worst case performance bound on a Moving Horizon Kalman Filter.

17.1 Introduction

Since its introduction, the Kalman Filter (KF) [2] has enjoyed widespread and successful use in estimating the state of a dynamic system assailed by noise. Given an accurate model and appropriate noise statistics, the KF provides state estimates that are mean-square-optimal. Fields that make wide use of the KF include aviation, localization, navigation, and economics. For its strengths, the performance of the KF will suffer if the provided system model is inaccurate, the noise statistics are inaccurate, or the noise signals can not be modeled as white, Gaussian random processes with known noise statistics. Furthermore, the recursive nature of the KF requires that measurement data arrive in order and at known period [6]. This makes the KF inappropriate for use in distributed and networked inappropriate for use in distributed and networked estimation schemes that may have transmission delays.

For these reasons, Moving Horizon Kalman Filter (MHKF) approaches (also referred to as receding horizon and limited memory approaches) have been investigated for over forty years [3]. These various approaches are often designed to address specific problems with the standard Kalman Filter. Problems addressed

N.R. Gans (✉)
National Research Council and Air Force Research Laboratory, Eglin Air Force Base, FL, USA
e-mail: ngans@ufl.edu

J.W. Curtis
Air Force Research Laboratory, Eglin Air Force Base, FL, USA
e-mail: jess.curtis@eglin.af.mil

through the use of a moving horizon include modeling error and bias [3, 8], unknown (possible nonGaussian) noise statistics [1], constrained state or noise distributions [7], and unknown initial state statistics [4]. Some authors note that the filter length is a tunable parameter [5], or that the system must be observable within the window length [1] but there appears to be little investigation into the effect of filter length on the accuracy of the estimate.

The investigation in this chapter is primarily interested in using the MHKF for overcoming data-rate and order dependency and hidden measurement-state correlation that may render standard Kalman filtering results inconsistent. Consider a central facility is estimating a dynamic system state via measurements from a collection of distributed sensors. If the communication between the sensors and the central facility is subject to random delays, and the central estimator is running a KF, the central estimator might receive a measurement from time $t-N$ at time t. It thus is currently carrying an estimate $\hat{x}_t$ and error covariance matrix P_t that do not allow a fusion with the delayed measurement. The problem of out-of-sequence measurements has been a well-studied topic in optimal estimation. Several approaches (both optimal and suboptimal) have been proposed to deal with the problem of filtering measurements that arrive for fusion out of their proper sequence [6].

One approach for solving this problem is to save a window of estimates $\hat{x}_i$ and their associated uncertainty matrices P_i. Then, in the event that a late measurement is received, the filter can be re-run from time $t-N$ to the present. The difficulty of this approach is that it requires a great deal more storage than a standard KF, and the computational requirements become likewise more severe (since instead of being required to perform a single time-update and measurement update, the central estimator might be required to do N times this computation at any given time).

One of the key questions in terms of analyzing this moving horizon strategy is how the performance degrades as a function of horizon length. In particular, is it possible to define a bound as a function of the horizon length such that if we desire some given level of estimation performance then a horizon length can be found to guarantee that minimal level of accuracy?

This paper attempts to address this question. Two error bounds are formulated that encapsulate worst case performance of a MHKF. Guarantee of worst case performance is essential in many robust estimation problems.

17.2 Linear State Estimation

Consider the linear, discrete-time system

$$\begin{aligned} x_{k+1} &= Ax_k + \omega_k \\ y_k &= Hx_k + \nu_k \end{aligned} \tag{17.1}$$

where $k \in [0, \infty]$ is the discrete time index, $x_k \in \Re^n$ is the system state at time k, $y_k \in \Re^m$ is the measurement vector at time k, and $\omega \in \Re^n$ and $\nu \in \Re^n$ are white, zero-mean, Gaussian, noise process with covariance matrices $Q \in \Re^{n\times n}$ and $R \in$

$\Re^{m\times m}$, respectively. The matrix $H \in \Re^{m\times n}$ maps a state vector into a measurement vector.

The well-known Kalman Filter equations [2] provide an optimal estimate in the sense that is provides the minimum mean square error estimate for linear estimators. The one-step KF equations are given as

$$\hat{x}_{k+1} = A\hat{x}_k + K_k(y_k - H\hat{x}_k) \tag{17.2}$$

$$K_k = AP_kH^T\big(R + HP_kH^T\big)^{-1} \tag{17.3}$$

$$P_{k+1} = A\big(P_k^{-1} + H^TR^{-1}H\big)^{-1}A^T + Q \tag{17.4}$$

where K_k is the Kalman gain and P_k is the covariance of the estimate error $e_k = \hat{x}_k - x_k$. It is known that if $\{H,\ A\}$ is observable and $\{A,\ Q\}$ controllable, then K_k and P_k will converge to a steady state values K and P such that

$$K = APH^T\big(R + HPH^T\big)^{-1}$$

$$P = A\big(P^{-1} + H^TR^{-1}H\big)^{-1}A^T + Q$$

17.2.1 Kalman Filter as an IIR Filter

Assume that the conditions for observability and controllability have been met, and that we have a collected measurements for time $i = \{0\ldots k\}$ (denote the batch of measurement by $\mathcal{Y}$), which will be used to estimate the current state at time k. Further assume that all of the measurement come from a single sensor with error covariance matrix R, and that we have an initial estimate of the state, $\hat{x}_{k-N}$, whose associated covariance matrix P_{k-N} is such that the filter is in steady state. That is, for all i

$$\hat{x}_{i+1} = A\hat{x}_i + K(y_{i+1} - HA\hat{x}_i)$$

where K is the constant, steady-state Kalman gain. An artificial measurement is created to capture this initial condition as

$$Ky_0 \overset{\triangle}{=} \hat{x}_0$$

Assume then that the measurement list contains this artificial measurement whose effect is to initialize the filter with our a priori information.

The time history of the state can be estimated as

$$\hat{x}_1 = AKy_0 + K(y_1 - HAKy_0)$$

$$= Ky_1 + (A - KHA)Ky_0 \tag{17.5}$$

$$\begin{aligned}\hat{x}_2 &= AKy_1 + A(A-KHA)Ky_0\\&\quad + K\big(y_2 - HAKy_1\\&\quad - HA(A-KHA)Ky_0\big)\\&= Ky_2 + (A-KHA)Ky_1\\&\quad + (A-KHA)(A-KHA)Ky_0 \qquad (17.6)\\&\quad \vdots\\\hat{x}_k &= \sum_{i=0}^{k}(A-KHA)^i Ky_{k-i} \qquad (17.7)\end{aligned}$$

This derivation provides a method to approximate a KF given a batch or list of measurements from a single known sensor. First, precompute the Kalman gain K and the discount matrix $A - KHA$. Then sum the individual contributions from each measurement as expressed in (17.7) to obtain a final estimate. If our initial information was such that the filter began in steady state, then this would yield an optimal estimate in the sense that the estimate x_k exactly equals the output from running a KF from time 0 up to k.

If there is insufficient a priori information to begin the estimation filter in steady state, the estimate produced by this technique would be suboptimal. This problem can be by addressed by simply assuming that the filter achieves steady-state operation after s time-steps (an assumption that is true in practice whenever the system is observable, stabilizable, and well modeled). This would imply that the measurements made on or after time s could be used in the final estimate as specified in (17.7). However, the first s terms would be wrong; alternatively we could redefine time-step s as the temporal origin and redefine all subsequent time-steps as $k' = k + s$.

17.2.2 Moving Average Implementation

Assume again that we have augmented the measurement list $\mathcal{Y}$ such that it contains a pseudo-measurement at time 0, which guarantees an optimal filter would operate in steady state for all time. Then, from (17.7), the optimal estimation of x_k is given by

$$\hat{x}_k = \sum_{i=0}^{k}(A-KHA)^{k-i} Ky_i \qquad (17.8)$$

Now suppose that we wish to implement a truncated summation (moving average) of the measurements as

$$\hat{x}_k = \sum_{i=k-N}^{k} (A - KHA)^{k-i} K y_i + \Phi(k - N) \tag{17.9}$$

where N is the length of the moving horizon and $\Phi(k-N)$ is the sum of the neglected terms. Define a new estimator given by

$$\hat{x}_k^{MH} \triangleq \sum_{i=k-N}^{k} (A - KHA)^{k-i} K y_i \tag{17.10}$$

Clearly, the error between this estimate and the KH estimate will be

$$\begin{aligned} e^{MH} &= \left\| \hat{x}_k^{MH} - \hat{x}_k \right\| \\ &= \left\| \Phi(k - N) \right\| \end{aligned}$$

17.3 MHE Performance Bound

The aim of this paper is to characterize the size of $\Phi(k-N)$, i.e., derive a bound on $\|\Phi(k-N)\|$ as a function of N. Note that e^{MH} does not refer to the true estimate error $\hat{x}_k^{MH} - x_k$, which is generally unknown.

The eigenvalues of the matrix $A - KHA$ will lie inside the unit circle, since we have assumed a steady-state KF. When the eigenvalues of $A - KHA$ have magnitude less than one, (17.10) represents a convergent geometric series with a finite sum.

Write the error e^{MH} as

$$\begin{aligned} e^{MH} &\triangleq \left\| \hat{x}_t - \hat{x}_t^{MH} \right\| \\ &= \left\| \sum_{i=0}^{t} (A - KHA)^{t-i} K y_i - \sum_{i=t-N}^{t} (A - KHA)^{t-i} K y_i \right\| \\ &= \left\| \sum_{i=0}^{t-N} (A - KHA)^{t-i} K y_i \right\| \\ &\leq \sum_{i=0}^{t-N} \left\| (A - KHA)^{t-i} K y_i \right\|, \\ &\leq \sum_{i=0}^{t-N} \left\| (A - KHA) \right\|^{t-i} \left\| K y_i \right\| \end{aligned}$$

Let $\bar{y}$ be the largest expected value of $\|Ky_i\|$ (or the norm of known sensor saturation) on the interval $i \in [0, t-N]$. Then we have that

$$e^{MH} \leq \bar{y} \sum_{i=0}^{t-N} \left\|(A - KHA)\right\|^{t-i} \tag{17.11}$$

If $\|A - KHA\| < 1$, the sum in (17.11) converges as follows (where $F \overset{\triangle}{=} \|A - KHA\|$)

$$\begin{aligned} S &= F^N + F^{N+1} + \cdots + F^t \\ FS &= F^{N+1} + \cdots + F^{t+1} \\ S - FS &= F^N - F^{t+1}, \\ (1-F)S &= \left(F^N - F^{t+1}\right) \\ S &= \frac{F^N - F^{t+1}}{1-F} \\ S &= \frac{\|A-KHA\|^N - \|A-KHA\|^{t+1}}{1 - \|A-KHA\|} \end{aligned}$$

The bound on e^{MH} can then be characterized as

$$\begin{aligned} e^{MH} &\leq \bar{y}\frac{\|A-KHA\|^N - \|A-KHA\|^{t+1}}{1 - \|A-KHA\|} \\ e^{MH} &\leq \mathcal{C}_1\left(\|A-KHA\|^N - \|A-KHA\|^{t+1}\right) \end{aligned}$$

where $\mathcal{C}_1 \overset{\triangle}{=} \frac{\bar{y}}{1-\|A-KHA\|}$. As $t+1-N > 0$ implies $\|A-KHA\|^{t+1-N} < 1$, it can be seen that

$$\begin{aligned} e^{MH} &\leq \mathcal{C}_1\|A-KHA\|^N\left(1 - \|A-KHA\|^{t+1-N}\right) \\ e^{MH} &\leq \mathcal{C}_1\|A-KHA\|^N \end{aligned}$$

This allows the derivation of an explicit algorithm for choosing a horizon length, given some maximum desired error level e_0

$$\ln e_0 \leq \ln \mathcal{C}_1 + N \ln \|A - KHA\| \tag{17.12}$$

so choose a horizon length N such that

$$N \geq \frac{\ln e_0 - \ln \mathcal{C}_1}{\ln \|A - KHA\|}$$

Note that in (17.12) the horizon length might be negative if the error is chosen too large. This situation implies that using a zero-length horizon would satisfy the performance requirement. Note also that N grows to infinity as e_0 goes to zero (because the denominator in (17.12) is always negative).

17.3.1 Situation When $\|A - KHA\| \geq 1$

Having all eigenvalues of $A - KHA$ less than one does not guarantee that $\|A - KHA\| < 1$. In this case, the matrix series in (17.10) will converge, but the series in (17.11) will not. This can be detected easily by testing whether the $\|A - KHA\| \geq 1$. If so, the bound calculation can be amended to address this issue.

Test $\|(A - KHA)^p\|$ for increasing values of p until $\|(A - KHA)^p\| < 1$. Take $\mathcal{C}_p = \frac{\bar{y}}{1-\|(A-KHA)^p\|}$ and

$$e^{MH} \leq \mathcal{C}_p \left\|(A - KHA)^p\right\|^{\frac{N}{p}}$$

The conditions under which the eigenvalues of $A - KHA$ are in the unit circle but $\|A - KHA\| > 1$ are not clear at this time, though it appears to depend on $\|A\|$ and $\|R\|$. This is an avenue of open investigation and future work.

17.3.2 Alternative Derivation

The bound developed in Sect. 17.3 is overly conservative, sometimes larger than the true error by several orders of magnitude. A similar approach can produce a tighter bound at the expense of greater computation time. However, given reasonable length horizon and dimension of state, the additional computation time is negligible.

Again take the error e^{MH} as

$$\begin{aligned} e^{MH} &\triangleq \left\|\hat{x}_t - \hat{x}_t^{MH}\right\|, \\ &= \left\|\sum_{i=0}^{t-N} (A - KHA)^{t-i} K y_i\right\| \end{aligned}$$

and set $F = (A - KHA)$. Then we have

$$e^{MH} = \left\|F^N K y_{t-N} + F^{N+1} K y_{t-N-2} + \cdots + F^t K y_0\right\|$$

There must exist a vector $\bar{y}$ such that

$$\begin{aligned} &\left\|F^N \bar{y} + F^{N+1} \bar{y} + \cdots + F^t \bar{y}\right\| \\ &= \left\|\left(F^N + F^{N+1} + \cdots + F^t\right)\bar{y}\right\| \\ &> \left\|F^N K y_{t-N} + F^{N+1} K y_{t-N-2} + \cdots + F^t K y_0\right\| \end{aligned} \tag{17.13}$$

We take advantage of two properties. First, given the singular value decomposition of a matrix $F = U\Sigma V$, the vector of V corresponding to the largest singular value will give max $\|Fv\|$. Second, given the svd's of matrices F, $F^2, \ldots, F^t$, as the power increases, the V matrices quickly converge to a constant, i.e., given $F^i = U_i \Sigma_i V_i$ and $F^{i+1} = U_{i+1}\Sigma_{i+1}V_{i+1}$, then $V_i \approx V_{i+1}$ for a appropriately large i. In practice, V_i and V_{i+1} are the same to within i decimal places. Therefore, for sufficiently large N, $\bar{y}$ will point along a direction close to the vector of V_N corresponding to the largest singular value, and $\|\bar{y}\|$ can be chosen appropriately, such as the maximum expected value or known saturation of the sensors. Then

$$\begin{aligned}
e^{MH} &= \left\|\left(F^N + F^{N+1} + \cdots + F^t\right)\bar{y}\right\| \\
&\le \left\|F^N\left(I + F + \cdots + F^{k-N}\right)\right\| \|\bar{y}\| \\
&\le \left\|F^N\left(F^{k-N+1} + I\right)(F+I)^{-1}\right\| \|\bar{y}\| \\
&\le \left\|\left(F^{k+1} + F^N\right)(F+I)^{-1}\right\| \|\bar{y}\|
\end{aligned}$$

It may be undesirable to have k explicitly in the expression for e^{MH}, since it will then have to be recomputed for each time. Under the assumption that $F^N \le F^{N+1}$ (in the sense that $\|F^N x\| \le \|F^{N+1}x\|$ for any vector x), e^{MH} can be upper bounded as

$$e^{MH} \le \|\bar{y}\| \left\|2F^N(F+I)^{-1}\right\|$$

17.4 Simulation and Analysis

Simulations and Monte Carlo analysis have been conducted to confirm the accuracy of the bound.

17.4.1 Simulation of Moving Horizon Estimator and Error Bound

First, an illustrative test of the Moving-horizon estimator is preformed, and its performance is compared to the maximum error e^{MH} bound. A two-dimensional linear time-invariant system of the form (17.1) is simulated with

$$A = \begin{bmatrix} 1 & .1 \\ 0 & 1 \end{bmatrix}, \qquad H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q = 0.75\begin{bmatrix} 0.033 & 0.05 \\ 0.05 & 1 \end{bmatrix}, \qquad R = 0.3\begin{bmatrix} 1 & 0.3 \\ 0.3 & 1 \end{bmatrix}$$

This represents 2nd order random walk process with sampling time 0.1 sec, full state feedback, and correlated sensor noise. For the moving-horizon estimator, a horizon length of 5 time-steps was used.

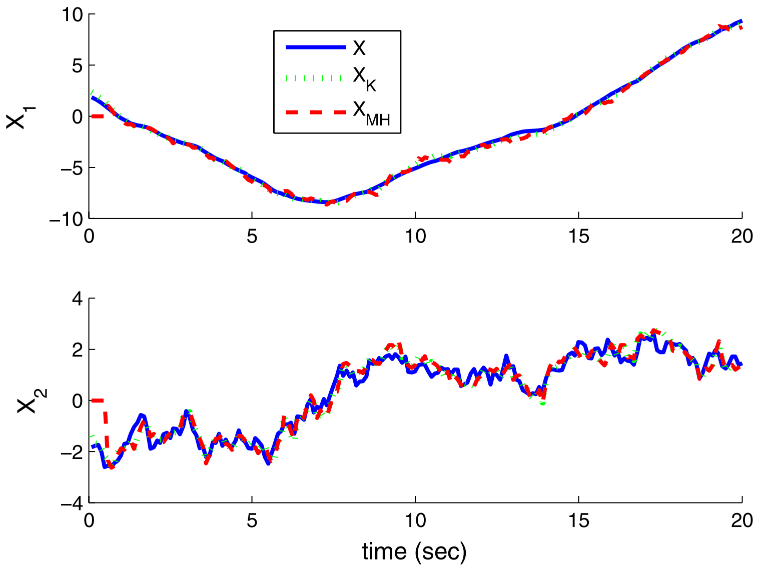


Fig. 17.1 Simulation of KF and MHKF estimators

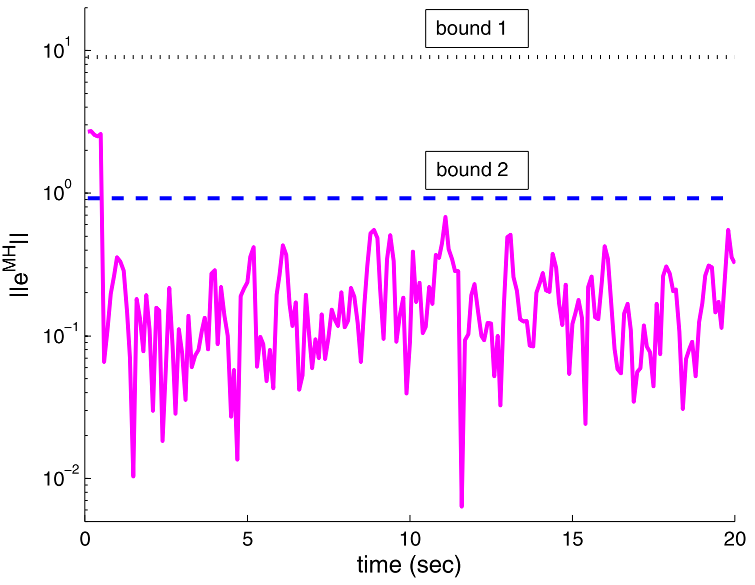


Fig. 17.2 The norm of estimation errors and the estimated bound

Figure 17.1 shows the system state simulated for 20 seconds (200 time samples) and the state estimates from both the Kalman filter and the moving horizon estima-

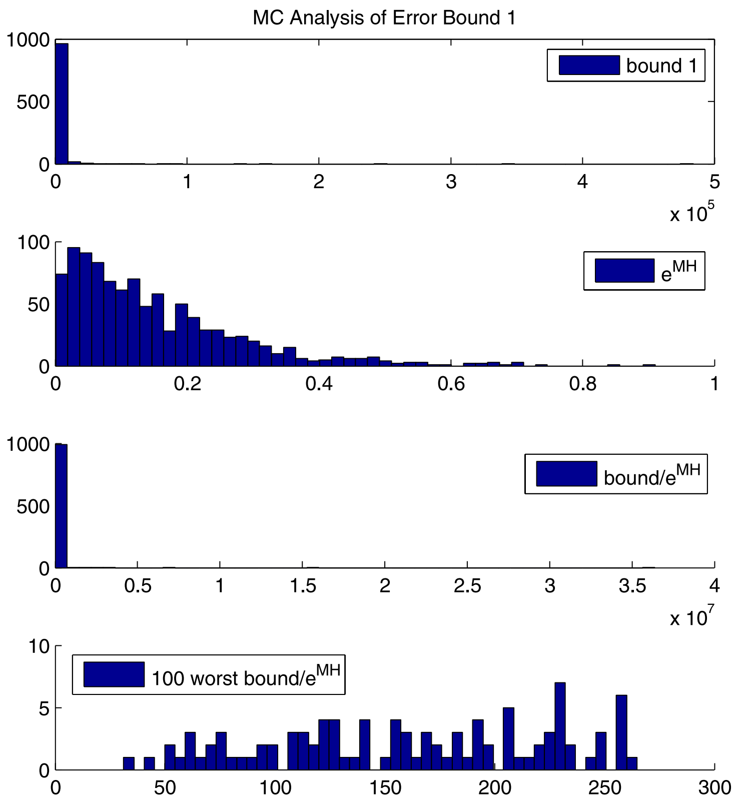


Fig. 17.3 Monte Carlo analysis of Bound 1

tor. Both the Kalman filter and the moving-horizon estimator track the true signal well.

The bound on e^{MH} for this system using the method in Sect. 17.3.1 is $e^{MH} \leq 8.96$. The bound using the method in Sect. 17.3.2 is $e^{MH} \leq 0.92$. This is illustrated in Fig. 17.2. A semi-log plot is used since the errors are rather small. Both bound estimates are met, and as expected, the second method of calculating the bound produces a lower bound.

17.4.2 Monte Carlo Analysis of Error Bound

The performance for a common system was demonstrated in Sect. 17.4.1. However, we wish to evaluate the performance for a wide variety of systems. To this end, a Monte Carlo analysis was performed with a large number of random systems.

One thousand (1000) random systems are generated as follows. First, generate four random matrices $Z_i \in \mathbb{R}^{2\times 2}$, $i \in \{1\dots 4\}$ where each element is a zero-mean

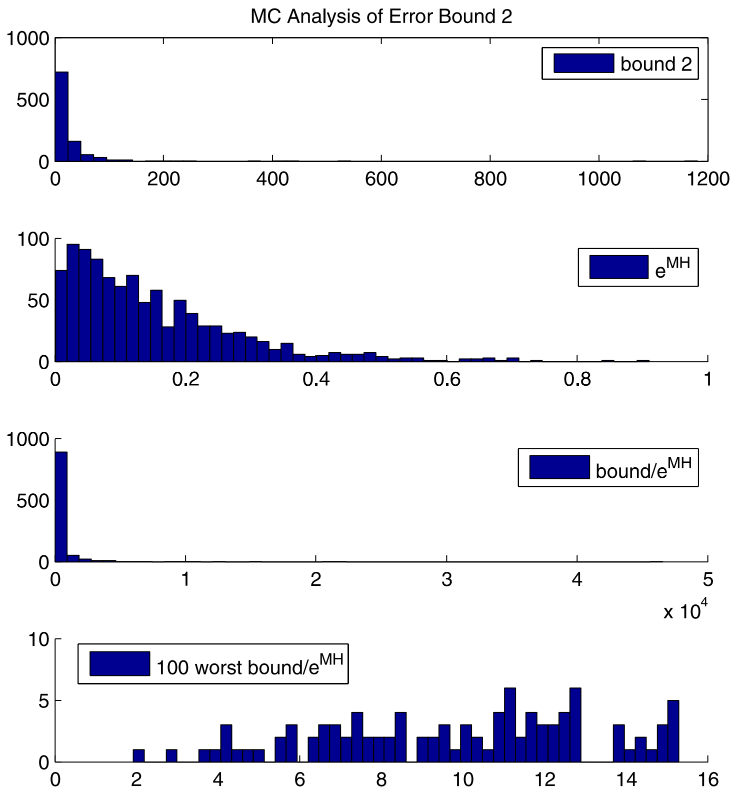


Fig. 17.4 Monte Carlo analysis of Bound 2

Gaussian random variable with unit variance. If the condition number of $Z_i > 10$, recalculate Z_i until a suitable matrix is found. Take initial system matrices

$$A_0 = \begin{bmatrix} 1 & .1 \\ 0 & 1 \end{bmatrix}, \qquad H_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q_0 = \begin{bmatrix} .1 & .01 \\ .01 & .1 \end{bmatrix}, \qquad R_0 = \begin{bmatrix} .1 & .01 \\ .01 & .1 \end{bmatrix}$$

and create random matrices

$$A = Z_1 A_0 Z_1^{-1}, \qquad H = Z_2 H_0 Z_2^{-1}$$

$$Q' = Z_3 Q_0 Z_3^{-1}, \qquad R' = Z_4 R_0 Z_4^{-1}$$

To ensure that the covariance matrices are positive-semidefinite and symmetric, the final values of Q and R are generated as

$$Q = \left(Q' Q'^T\right)^{\frac{1}{2}}, \qquad R = \left(R' R'^T\right)^{\frac{1}{2}}$$

where $(\cdot)^{\frac{1}{2}}$ represent the matrix square root.

Each of the 1000 random systems are tested for both the Kalman filter and moving-horizon estimators. Histograms are generated of the results in Figs. 17.3 and 17.4. It can be seen that both bounds have some outliers that are very large, though the second method of producing a bound is typically smaller. To best illustrate the tightness of the bound, we consider the ratio $\frac{bound}{e^{MH}}$. Any ratio less than one indicates the bound was violated. Motivated by robust control applications, where min/max performance is critical, we focus on the 100 smallest values of this ratio from the distribution, representing the cases where the bound was closest to the true value of e^{MH}. It is seen that both bounds are never violated by e^{MH}, and the second bound is roughly an order of magnitude tighter, even for the minimum ratios.

17.5 Future Work

There are several avenues of future work. The bounds in this paper represent useful measures to ensure robust performance, but for many cases the bounds are typically much larger than the true errors, so some method to get a tighter, more accurate bound is desirable. Another approach is to generate a stochastic bound, that is a tighter bound with a known probability that the bound is not exceeded. Experiments will also be performed to test the bound in real scenarios when the system is not perfectly modeled.

References

1. Alessandri, A., Baglietto, M., Battistelli, G.: Receding-horizon estimation for discrete-time linear systems. IEEE Trans. Autom. Control **48**(3), 473–478 (2003)
2. Brown, R.G., Hwang, P.Y.C.: Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises and Solutions, 3rd edn. Wiley, New York (1996)
3. Jazwinski, A.: Limited memory optimal filtering. IEEE Trans. Autom. Control **13**(5), 558–563 (1968)
4. Kwon, W.H., Kim, P.S., Park, P.G.: A receding horizon Kalman fir filter for discrete time-invariant systems. IEEE Trans. Autom. Control **44**(9), 1787–1791 (1999)
5. Ling, K.V., Lim, K.W.: Receding horizon recursive state estimation. IEEE Trans. Autom. Control **44**(9), 1750–1753 (1999)
6. Plett, G.L., Zarzhitsky, D., Pack, D.: Out of order sigma point Kalman filtering for target localization using cooperating unmanned aerial vehicles. In: Advances in Cooperative Control and Optimization, pp. 21–43 (2007)
7. Rao, C.V., Rawlings, J.B., Lee, J.H.: Constrained linear state estimation–a moving horizon approach. Automatica **37**(10), 1619–1628 (2001)
8. Wang, Z.-O., Zhang, J.: A Kalman filter algorithm using a moving window with applications. Int. J. Syst. Sci. **26**(8), 1465–1478 (1995)

Chapter 18
A *p*-norm Discrimination Model for Two Linearly Inseparable Sets

**Pavlo Krokhmal, Robert Murphey,
Panos M. Pardalos, and Zhaohan Yu**

Summary We propose a new p-norm linear discrimination model that generalizes the model of Bennett and Mangasarian (Optim. Methods Softw. 1:23–34, 1992) and reduces to linear programming problems with p-order conic constraints. We demonstrate that the developed model possesses excellent methodological and computational properties (e.g., it does not allow for a null separating hyperplane when the sets are linearly separable, etc.). The presented approach for handling linear programming problems with p-order conic constraints relies on construction of polyhedral approximations for p-order cones. A case study on several popular data sets that illustrates the advantages of the developed model is conducted.

18.1 Introduction

Consider two discrete sets $\mathcal{A}, \mathcal{B} \in \mathbb{R}^n$ comprised of k and m points, respectively: $\mathcal{A} = \{\mathbf{a}_1, \dots, \mathbf{a}_k\}$, $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_m\}$. One of the principal tasks arising in machine learning and data mining is that of *discrimination* of these sets, namely, constructing a surface $f(\mathbf{x}) = 0$ such that $f(\mathbf{x}) < 0$ for any $\mathbf{x} \in \mathcal{A}$ and $f(\mathbf{x}) > 0$ for all $\mathbf{x} \in \mathcal{B}$.

P. Krokhmal (✉) · Z. Yu
Department of Mechanical and Industrial Engineering, University of Iowa, Iowa City, IA 52242, USA
e-mail: krokhmal@engineering.uiowa.edu

Z. Yu
e-mail: zhaohan-yu@uiowa.edu

R. Murphey
Air Force Research Lab, Eglin AFB, FL, USA
e-mail: robert.murphey@eglin.af.mil

P.M. Pardalos
Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, USA
e-mail: pardalos@ufl.edu

Of particular interest is the linear separating surface (hyperplane):

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} - \gamma = 0 \tag{18.1}$$

Clearly, existence of such a separating hyperplane is not guaranteed;[1] in general, a separating hyperplane that minimizes some sort of *misclassification error* is desired.

Observe that if points $\mathbf{y}^{(1)}$, $\mathbf{y}^{(2)} \in \mathbb{R}^n$ satisfy the inequalities

$$\mathbf{w}^\top \mathbf{y}^{(1)} - \gamma > 0, \qquad \mathbf{w}^\top \mathbf{y}^{(2)} - \gamma < 0$$

for some $\mathbf{w}$ and γ, then they are located on the opposite sides of the hyperplane $\mathbf{w}^\top \mathbf{x} - \gamma = 0$. Consequently, the discrete sets $\mathcal{A}$, $\mathcal{B} \subset \mathbb{R}^n$ are considered *linearly separable* if and only if there exists $\mathbf{w} \in \mathbb{R}^n$ such that

$$\mathbf{w}^\top \mathbf{a}_i > \gamma > \mathbf{w}^\top \mathbf{b}_j \quad \text{for all } i = 1, \ldots, k, \ \ j = 1, \ldots, m$$

with an appropriately chosen γ, or, equivalently,

$$\min_{\mathbf{a}_i \in \mathcal{A}} \mathbf{a}_i^\top \mathbf{w} > \max_{\mathbf{b}_j \in \mathcal{B}} \mathbf{b}_j^\top \mathbf{w} \tag{18.2}$$

Definition (18.2) is not amenable for use in mathematical programming models since it involves strict inequalities. However, the fact that the separating hyperplane can be scaled by any nonnegative factor allows one to formulate the following result, whose proof we include for completeness of exposition.

Proposition 1 (Bennett and Mangasarian [5]) *Discrete sets $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^n$ represented by matrices $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_k)^\top \in \mathbb{R}^{k \times n}$ and $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)^\top \in \mathbb{R}^{m \times n}$, respectively, are linearly separable if and only if*

$$\mathbf{A}\mathbf{w} \geq \mathbf{e}\gamma + \mathbf{e}, \qquad \mathbf{B}\mathbf{w} \leq \mathbf{e}\gamma - \mathbf{e} \quad \textit{for some } \mathbf{w} \in \mathbb{R}^n, \ \gamma \in \mathbb{R}, \tag{18.3}$$

where $\mathbf{e}$ is the vector of ones of the appropriate dimension, $\mathbf{e} = (1, \ldots, 1)^\top$.

Proof Let $\mathcal{A}$ and $\mathcal{B}$ be linearly separable, then in accordance to definition (18.2), there exists $\mathbf{v} \in \mathbb{R}^n$ such that

$$\min_{i=1,\ldots,k} \mathbf{a}_i^\top \mathbf{v} =: a_* > b^* := \max_{j=1,\ldots,m} \mathbf{b}_j^\top \mathbf{v} \tag{18.4}$$

Denote $\mathbf{w} = 2\mathbf{v}/(a_* - b^*)$, and $\gamma = (a_* + b^*)/(a_* - b^*)$; then for any $\mathbf{a}_i \in \mathcal{A}$

$$\mathbf{a}_i^\top \mathbf{w} - \gamma - 1 = \frac{2}{a_* - b^*} \mathbf{a}_i^\top \mathbf{v} - \frac{2a_*}{a_* - b^*} = \frac{2}{a_* - b^*} \Big(\mathbf{a}_i^\top \mathbf{v} - \min_{i=1,\ldots,m} \mathbf{a}_i^\top \mathbf{v} \Big) \geq 0$$

which means that $\mathbf{A}\mathbf{w} - \mathbf{e}\gamma - \mathbf{e} \geq \mathbf{0}$. The second inequality in (18.3) follows analogously. □

[1] It is easy to see that a separating hyperplane exists if the convex hulls of sets $\mathcal{A}$ and $\mathcal{B}$ are disjoint.

In the next section, we introduce a new linear separation model that is based on p-order conic programming, and discuss its key properties.

18.2 The p-norm Linear Separation Model

In this chapter, we generalize the *robust linear discrimination* model proposed by Bennett and Mangasarian [5].

$$\begin{aligned} \min \quad & \frac{1}{k}\mathbf{e}^\top \mathbf{y} + \frac{1}{m}\mathbf{e}^\top \mathbf{z} && (18.5a)\\ \text{s.t.} \quad & \mathbf{y} \geq -\mathbf{A}\mathbf{w} + \mathbf{e}\gamma + \mathbf{e} && (18.5b)\\ & \mathbf{z} \geq \mathbf{B}\mathbf{w} - \mathbf{e}\gamma + \mathbf{e} && (18.5c)\\ & \mathbf{z}, \mathbf{y} \geq \mathbf{0} && (18.5d) \end{aligned}$$

The linear programming model (18.5) determines a hyperplane $\mathbf{w}^{*\top}\mathbf{x} - \gamma^* = 0$ that minimizes the average misclassification error. Indeed, in accordance to the definition (18.3), the points of sets $\mathcal{A}$ and $\mathcal{B}$ that violate (18.3) will correspond to the nonzero components of vectors $\mathbf{y}$ and $\mathbf{z}$ in the constraints (18.5b) and (18.5c), respectively.

This interpretation allows us to reformulate the optimization problem (18.5) in the form of a stochastic programming problem

$$\min_{(\mathbf{w},\gamma)\in\mathbb{R}^{n+1}} \left\{ \mathsf{E}\left[\left(-\mathbf{a}^\top\mathbf{w} + \gamma + 1\right)_+\right] + \mathsf{E}\left[\left(\mathbf{b}^\top\mathbf{w} - \gamma + 1\right)_+\right]\right\} \qquad (18.6)$$

where $\mathbf{a}$ and $\mathbf{b}$ are uniformly distributed random vectors with support sets $\mathcal{A}$ and $\mathcal{B}$, correspondingly:

$$\mathsf{P}\{\mathbf{a} = \mathbf{a}_i\} = \frac{1}{k}, \qquad \mathsf{P}\{\mathbf{b} = \mathbf{b}_j\} = \frac{1}{m} \quad \text{for all } \mathbf{a}_i \in \mathcal{A},\ \mathbf{b}_j \in \mathcal{B} \qquad (18.7)$$

and $(x)_\pm = \max\{0, \pm x\}$. In this sense, the misclassification errors of points from $\mathcal{A}$ and/or $\mathcal{B}$ can be viewed as realizations of random variables $X_\mathcal{A} = X_\mathcal{A}(\mathbf{w}, \gamma)$ and $X_\mathcal{B} = X_\mathcal{B}(\mathbf{w}, \gamma)$, whose smaller values are preferred, and thus the parameters $\mathbf{w}$ and γ must be selected so as $X_\mathcal{A}$ and $X_\mathcal{B}$ assume values that are "small".

As it is well known in stochastic programming and risk analysis, the "risk" associated with random outcome is often attributed to the "heavy" tails of the probability distribution. The risk-inducing "heavy" tails of probability distributions, are, in turn, characterized by the distribution's higher moments. Thus, if the misclassifications introduced by a separating hyperplane can be viewed as "random", the misclassification risk may be controlled better if one minimizes not the average (expected value) of the misclassification errors, but their moments of order $p > 1$. This gives rise to the following formulation for linear discrimination of sets $\mathcal{A}$ and $\mathcal{B}$:

$$\min_{(\mathbf{w},\gamma)\in\mathbb{R}^{n+1}} \delta_1 \left\| \left(-\mathbf{a}^\top\mathbf{w} + \gamma + 1\right)_+ \right\|_p + \delta_2 \left\| \left(\mathbf{b}^\top\mathbf{w} - \gamma + 1\right)_+ \right\|_p, \quad p \in [1, +\infty] \qquad (18.8)$$

where $\|\cdot\|_p$ is the "functional" $\mathcal{L}_p$ norm, which in the probabilistic context can be written as

$$\|X\|_p = \begin{cases} \left(\mathsf{E}|X|^p\right)^{1/p}, & p \in [1,\infty) \\ \sup|X|, & p = \infty \end{cases}$$

Assuming again that points of the sets $\mathcal{A}$ and $\mathcal{B}$ are "equiprobable" (or, in other words, all points of set $\mathcal{A}$, and, correspondingly, $\mathcal{B}$, have equal "importance"), linear discrimination problem (18.8) can be written as follows

$$\min \quad \delta_1 k^{-1/p}\xi + \delta_2 m^{-1/p}\eta \tag{18.9a}$$

$$\text{s.t.} \quad \xi \geq \|\mathbf{y}\|_p \tag{18.9b}$$

$$\eta \geq \|\mathbf{z}\|_p \tag{18.9c}$$

$$\mathbf{y} \geq -\mathbf{A}\mathbf{w} + \mathbf{e}\gamma + \mathbf{e} \tag{18.9d}$$

$$\mathbf{z} \geq \mathbf{B}\mathbf{w} - \mathbf{e}\gamma + \mathbf{e} \tag{18.9e}$$

$$\mathbf{z}, \mathbf{y} \geq \mathbf{0}, \quad \xi, \eta \geq 0 \tag{18.9f}$$

In the mathematical programming formulation (18.9), $\|\cdot\|_p$ denotes the "vector" norm in finite-dimensional space, i.e., for $\mathbf{x} \in \mathbb{R}^n$,

$$\|\mathbf{x}\|_p = \begin{cases} \left(|x_1|^p + \cdots + |x_n|^p\right)^{1/p}, & p \in [1,\infty) \\ \max\left\{|x_1|, \ldots, |x_n|\right\}, & p = \infty \end{cases}$$

(in the sequel, it shall be clear from the context whether the "functional" or "vector" definition of p-norm is used).

Model (18.9) constitutes a linear programming problem with p-order conic constraints (18.9b)–(18.9c). Using the "vector" norm notation, formulation (18.9) can be more succinctly presented as

$$\min_{(\mathbf{w},\gamma)\in\mathbb{R}^{n+1}} \frac{\delta_1}{k^{1/p}} \left\|(-\mathbf{A}\mathbf{w} + \mathbf{e}\gamma + \mathbf{e})_+\right\|_p + \frac{\delta_2}{m^{1/p}} \left\|(\mathbf{B}\mathbf{w} - \mathbf{e}\gamma + \mathbf{e})_+\right\|_p \tag{18.10}$$

The p-conic programming linear separation model (18.8)–(18.9) shares many key properties with the LP separation model of Bennett and Mangasarian [5], including the guarantee that the optimal solution of (18.9) is nonzero in $\mathbf{w}$ for linearly separable sets.

Proposition 2 *When sets $\mathcal{A}$ and $\mathcal{B}$, represented by matrices* $\mathbf{A}$ *and* $\mathbf{B}$, *are linearly separable* (*i.e., they satisfy* (18.2) *and* (18.3)), *the separating hyperplane* $\mathbf{w}^{*\top}\mathbf{x} = \gamma^*$ *given by an optimal solution of* (18.8)–(18.9) *satisfies* $\mathbf{w}^* \neq \mathbf{0}$.

Proof Zero optimal value of (18.9a) immediately implies that at optimality $\mathbf{y}^* = \mathbf{z}^* = \mathbf{0}$, or, equivalently,

$$-\mathbf{A}\mathbf{w}^* + \mathbf{e}\gamma^* + \mathbf{e} \leq \mathbf{0}, \qquad \mathbf{B}\mathbf{w}^* - \mathbf{e}\gamma^* + \mathbf{e} \leq \mathbf{0}$$

If one assumes that $\mathbf{w}^* = \mathbf{0}$, then the above inequalities require that

$$\gamma^* \leq -1, \qquad \gamma^* \geq 1$$

The contradiction furnishes the desired statement. □

Secondly, the p-norm separation model (18.9) can produce a $\mathbf{w} = \mathbf{0}$ solution only in a rather special case that is identified by Theorem 1 below.

Theorem 1 *Assume that the p-order conic programming problem* (18.9)–(18.10) *is strictly feasible and, without loss of generality,* $0 < \delta_1 < \delta_2$. *Then, for any* $p \in (1, \infty)$ *the p-order conic programming problem* (18.9) *has an optimal solution where* $\mathbf{w}^* = \mathbf{0}$ *if and only if*

$$\frac{\mathbf{e}^\top}{k}\mathbf{A} = \mathbf{v}^\top \mathbf{B}, \quad \text{where } \mathbf{e}^\top \mathbf{v} = 1,\ \mathbf{v} \geq \mathbf{0},\ \|\mathbf{v}\|_q \leq \frac{\delta_2}{\delta_1 m^{1/p}} \tag{18.11a}$$

where q satisfies $\frac{1}{p} + \frac{1}{q} = 1$. *In other words, the arithmetic mean of the points in* $\mathcal{A}$ *must be equal to some convex combination of points in* $\mathcal{B}$. *In the case of* $\delta_1 = \delta_2$ *condition,* (18.11a) *reduces to*

$$\frac{\mathbf{e}^\top}{k}\mathbf{A} = \frac{\mathbf{e}^\top}{m}\mathbf{B} \tag{18.11b}$$

i.e., the arithmetic means of the points of sets $\mathcal{A}$ *and* $\mathcal{B}$ *must coincide.*

Proof From formulation (18.10) of problem (18.9), it follows that in the case when $\mathbf{w} = \mathbf{0}$ at optimality, the corresponding optimal value of the objective (18.9a) is determined as

$$\min_{\gamma \in \mathbb{R}} f(\gamma) = \frac{\delta_1}{k^{1/p}} \left(\sum_{i=1}^{k} (1+\gamma)_+^p \right)^{1/p} + \frac{\delta_2}{m^{1/p}} \left(\sum_{j=1}^{m} (1-\gamma)_+^p \right)^{1/p}$$

Clearly,

$$f(\gamma) = \begin{cases} \delta_1(1+\gamma), & \gamma \geq 1, \\ \delta_1 + \delta_2 + \gamma(\delta_1 - \delta_2), & -1 < \gamma < 1, \\ \delta_2(1-\gamma), & \gamma \leq -1 \end{cases}$$

whence

$$\min_{\gamma \in \mathbb{R}} f(\gamma) = f(1) = 2\delta_1 \tag{18.12}$$

due to the assumption $0 < \delta_1 < \delta_2$. Next, consider the dual of the p-conic programming problem (18.9):

$$\begin{aligned} \max \quad & \mathbf{e}^\top\mathbf{u} + \mathbf{e}^\top\mathbf{v} \\ \text{s.t.} \quad & -\mathbf{A}^\top\mathbf{u} + \mathbf{B}^\top\mathbf{v} = \mathbf{0} \\ & \mathbf{e}^\top\mathbf{u} - \mathbf{e}^\top\mathbf{v} = \mathbf{0} \\ & \mathbf{0} \le \mathbf{u} \le -\mathbf{s} \\ & \mathbf{0} \le \mathbf{v} \le -\mathbf{t} \\ & \|\mathbf{s}\|_q \le \delta_1 k^{-1/p} \\ & \|\mathbf{t}\|_q \le \delta_2 m^{-1/p} \end{aligned} \tag{18.13}$$

where q is such that $\frac{1}{p} + \frac{1}{q} = 1$. Since (18.9) is strictly feasible and bounded from below, the duality gap for the primal-dual pair of p-order conic programming problems (18.9) and (18.13) is zero. Then, from the first two constraints of (18.13), we have $\mathbf{A}^\top\mathbf{u}^* = \mathbf{B}^\top\mathbf{v}^*$ as well as $\mathbf{e}^\top\mathbf{u}^* = \mathbf{e}^\top\mathbf{v}^*$, which given that the optimal objective value (18.13) is $2\delta_1$, implies that an optimal $\mathbf{u}^*$ must satisfy

$$\mathbf{e}^\top\mathbf{u}^* = \delta_1 \tag{18.14a}$$

Also, from (18.13) it follows that

$$\|\mathbf{u}^*\|_q \le \delta_1 k^{-1/p} \tag{18.14b}$$

Then, it is easy to see that the unique solution of system (18.14) is

$$\mathbf{u}^* = \frac{\delta_1}{k}\mathbf{e} = \left(\frac{\delta_1}{k}, \dots, \frac{\delta_1}{k}\right) \tag{18.15}$$

which corresponds to the point where the surface $(u_1^q + \cdots + u_k^q)^{1/q} = \delta_1 k^{-1/p}$ is tangent to the hyperplane $u_1 + \cdots + u_k = \delta_1$ in the positive orthant of $\mathbb{R}^k$.

Similarly, an optimal $\mathbf{v}^*$ must satisfy $\mathbf{e}^\top\mathbf{v}^* = \delta_1$ and $\|\mathbf{v}^*\|_q \le \delta_2 m^{-1/p}$. Note, however, that in the case when $\delta_2/\delta_1 > 1$ such $\mathbf{v}^*$ is not unique. By substituting the obtained characterizations for $\mathbf{u}^*$ and $\mathbf{v}^*$ in the constraint $\mathbf{A}^\top\mathbf{u}^* = \mathbf{B}^\top\mathbf{v}^*$ of the dual (18.13) and dividing by δ_1, we obtain (18.11a). When $\delta_1 = \delta_2$, the optimal $\mathbf{v}^*$ is unique: $\mathbf{v}^* = \frac{\delta_1}{m}\mathbf{e}$, and yields (18.11b). $\square$

Observe that Theorem 1 implies that in the case of $\delta_1 = \delta_2$ (i.e., when misclassification of points in one set is not favored over that for points of the other set), the p-norm discrimination model (18.9) produces a separating hyperplane with $\mathbf{w} = \mathbf{0}$ only when the "geometric centers" (arithmetic means) of the sets $\mathcal{A}$ and $\mathcal{B}$ coincide. In many situations, this would mean that there is significant "overlap" between the convex hulls of sets $\mathcal{A}$ and $\mathcal{B}$. In practice, this means that such sets, indeed, cannot be efficiently separated, at least by a hyperplane, thus occurrence of $\mathbf{w}^* = \mathbf{0}$ solution in (18.9) should not be regarded as the shortfall of the particular formulation (18.9), but rather the general inapplicability of the linear discrimination method to the specific sets $\mathcal{A}$ and $\mathcal{B}$.

In the case when a "bias" with regard to the importance of misclassification of points of sets $\mathcal{A}$ and $\mathcal{B}$ needs to be introduced by setting $\delta_2 > \delta_1$, occurrence of a $\mathbf{w}^* = \mathbf{0}$ solution in (18.9) does not necessarily imply that sets $\mathcal{A}$ and $\mathcal{B}$ are hardly amenable to linear separation. Indeed, in this case Theorem 1 only claims that the "geometric center" of set $\mathcal{A}$ must coincide with some convex combination of points of set $\mathcal{B}$, i.e., it must coincide with some point inside the convex hull of set $\mathcal{B}$. In this case, linear discrimination can still be a feasible approach, albeit at a cost of significant misclassification errors.

In order to have the stricter condition (18.11b) for the occurrence of $\mathbf{w}^* = \mathbf{0}$ solution in the situation when the preferences for misclassification error are different for sets $\mathcal{A}$ and $\mathcal{B}$, the p-norm linear discrimination model can be extended to the case where misclassifications of points in $\mathcal{A}$ and $\mathcal{B}$ are measured using norms of different orders:

$$\min_{(\mathbf{w},\gamma)\in\mathbb{R}^{n+1}} k^{-1/p_1} \left\|(-\mathbf{A}\mathbf{w} + \mathbf{e}\gamma + \mathbf{e})_+\right\|_{p_1} + m^{-1/p_2} \left\|(\mathbf{B}\mathbf{w} - \mathbf{e}\gamma + \mathbf{e})_+\right\|_{p_2},$$
$$p_{1,2} \in [1, \infty) \tag{18.16}$$

Intuitively, a norm of higher order places more "weight" on outliers; indeed, application of the $p = 1$ norm would minimize the average misclassification error, in effect regarding all misclassifications as equally important. In contrast, application of the $p = \infty$ norm would minimize the largest misclassification error. Thus, by selecting appropriately the orders p_1 and p_2 in (18.16) one may introduce tolerance preferences on misclassifications in sets $\mathcal{A}$ and $\mathcal{B}$. At the same time, it can be shown that the occurrence of $\mathbf{w}^* = \mathbf{0}$ solution in (18.16) would signal the presence of the aforementioned singularity about the sets $\mathcal{A}$, $\mathcal{B}$. Namely, we have the following corollary.

Corollary 1 *The statement of Theorem 1 carries over to model (18.16) practically without modifications.*

In the next section, we discuss the details of practical implementation of the p-norm linear discrimination model (18.9).

18.3 Implementation of p-order Conic Programming Problems via Polyhedral Approximations

Constraints (18.9b)–(18.9c) in the p-norm linear separation model (18.9) constitute p-order cones in $\mathbb{R}^{m+1}$ and $\mathbb{R}^{k+1}$, correspondingly, and are central to practical implementation of the p-norm separation method. Depending on the value of the parameter p, the following cases can be identified.

$p = 1$: in this case, given the nonnegativity of variables $\mathbf{y}$, $\mathbf{z}$, constraints (18.9b)–(18.9c) reduce to linear inequalities

$$\xi \geq \mathbf{e}^\top \mathbf{y}, \qquad \eta \geq \mathbf{e}^\top \mathbf{z}.$$

This particular case has been considered in [5]; in general, the amenability of the 1-norm, also known as the "Manhattan distance", etc., to implementation via linear constraints has been exploited in a of variety approaches and applications, too numerous to cite here.

$p = \infty$: in this case $\|\mathbf{x}\|_\infty = \max_i |x_i|$, whereby the conic constraints (18.9b)–(18.9c) reduce to a system of $m + k$ inequalities

$$\mathbf{e}\xi \geq \mathbf{y}, \qquad \mathbf{e}\eta \geq \mathbf{z}$$

Due to the linearity of the above constraints, we do not pursue this case further, as our main interest is in models with "true" (or nonlinear) p-cone constraints.

$p \in (1, \infty)$: this is the "general" case that constitutes the focus of the present endeavor. In addition to the data mining application described above, the general p-order conic programming has been considered in the context of Steiner minimum tree problem on a given topology [19], stochastic programming and risk optimization [8, 9]; an application of p-order conic programming to integer programming problems is discussed in [6].

Of particular importance is the case $p = 2$, when the constraints (18.9b)–(18.9c) represent second-order (quadratic, "ice cream", or Lorentz) cones,

$$\begin{aligned} \xi &\geq \|\mathbf{y}\|_2 = \left(\mathbf{y}^\top \mathbf{y}\right)^{1/2} \\ \eta &\geq \|\mathbf{z}\|_2 = \left(\mathbf{z}^\top \mathbf{z}\right)^{1/2} \end{aligned} \tag{18.17}$$

The second-order conic programming (SOCP), which deals with optimization problems that contain constraints of form (18.17), constitutes a well-developed subject of convex programming. A number of efficient SOCP algorithms have been developed in the literature (e.g., [12, 13], and others, see an overview in [1]), and some of them were implemented into software solver codes such as MOSEK, SeDuMi [2, 15].

From the computational standpoint, the case of general p, when the cone is not self-dual, has received much less attention in the literature compared to the conic quadratic programming. Interior-point approaches to p-order conic programming have been considered by Xue and Ye [19] with respect to minimization of sum of p-norms; a self-concordant barrier for p-cone has also been introduced in [10]. Glineur and Terlaky [7] proposed an interior point algorithm along with the corresponding barrier functions for a related problem of l_p-norm optimization (see also [17]). In the case when p is a rational number, the existing primal-dual methods of SOCP can be employed for solving p-order conic optimization problems using a reduction of p-order conic constraints to a system of linear and second-order conic constraints proposed in [11] and [3].

Our approach to handling p-order conic constraints (18.9b)–(18.9c) in the case of $p \in (1,2) \cup (2,+\infty)$ consists in constructing of polyhedral approximations for the p-order constraints and thus reducing the p-norm linear separation problem (18.9) to an LP. In many respects, the proposed approach builds upon the work of Ben-Tal and Nemirovski [4] where an efficient lifted polyhedral approximation for the second-order ($p = 2$) cones was developed, and whose motivation was to devise a practical method of solving problems with second-order conic constraints that would utilize the powerful machinery of LP solvers. Recently, Ben-Tal and Nemirovski's polyhedral approximation has been proven very effective in solving mixed integer conic quadratic problems in [18].

18.3.1 Polyhedral Approximations of p-order Cones

The proposed approach to solving the p-norm linear separation model (18.9) is based on the construction of polyhedral approximations for p-order conic constraints. Without loss of generality, we restrict our attention to a p-order cone in the positive orthant of $(N+1)$-dimensional space:

$$\mathcal{K}_p^{(N+1)} = \left\{ \mathbf{x} \in \mathbb{R}_+^{N+1} \;\middle|\; x_{N+1} \geq \left(x_1^p + \cdots + x_N^p\right)^{1/p} \right\} \tag{18.18}$$

where $\mathbb{R}_+ = [0,+\infty)$. By a polyhedral approximation of $\mathcal{K}_p^{(N+1)}$, we understand a (convex) polyhedral cone in $\mathbb{R}^{N+1+\kappa_m}$, where κ_m may be generally nonzero:

$$\mathcal{H}_{p,m}^{(N+1)} = \left\{ \begin{pmatrix} \mathbf{x} \\ \mathbf{u} \end{pmatrix} \in \mathbb{R}_+^{N+1+\kappa_m} \;\middle|\; \mathbf{H}_{p,m}^{(N+1)} \begin{pmatrix} \mathbf{x} \\ \mathbf{u} \end{pmatrix} \geq \mathbf{0} \right\} \tag{18.19}$$

having the properties that

(H1) Any $\mathbf{x} \in \mathcal{K}_p^{(N+1)}$ can be extended to some $(\mathbf{x},\mathbf{u}) \in \mathcal{H}_{p,m}^{(N+1)}$
(H2) For some prescribed $\varepsilon > 0$, any $(\mathbf{x},\mathbf{u}) \in \mathcal{H}_{p,m}^{(N+1)}$ satisfies

$$\left(x_1^p + \cdots + x_N^p\right)^{1/p} \leq (1+\varepsilon)x_{N+1} \tag{18.20}$$

Here, m is the parameter of construction that controls the approximation error ε.

In constructing the polyhedral approximations (18.19) for p-order cone (18.18), we follow the approach of Ben-Tal and Nemirovski [4] (see also [3, 11]), which allows for reducing the dimensionality of approximation (18.19) by replacing the $(N+1)$-dimensional conic constraint with an equivalent system of 3-dimensional conic constraints, and then constructing a polyhedral approximation for each of the 3D cones.

18.3.2 "Tower-of-Variables" (Ben-Tal and Nemirovski [4])

The "tower-of-variables" technique has been originally proposed by Ben-Tal and Nemirovski [4] for construction of a polyhedral approximation for quadratic ($p = 2$) cones, but it applies to p-order cones as well. Assuming that $N = 2^d$ for some integer d, a p-cone in $\mathbb{R}^{N+1}$

$$x_{N+1} \geq \left(x_1^p + \cdots + x_N^p\right)^{1/p} \tag{18.21}$$

can equivalently be represented as an intersection of three-dimensional p-cones in a higher-dimensional space $\mathbb{R}^{2N-1}$

$$x_j^{(\ell)} \geq \left\| \left(x_{2j-1}^{(\ell-1)}, x_{2j}^{(\ell-1)}\right) \right\|_p, \quad j = 1, \ldots, 2^{d-\ell}, \ \ell = 1, \ldots, d \tag{18.22}$$

The set of constraints (18.22) can be visualized as a "tower" or "pyramid" consisting of $d + 1$ "levels" denoted by the superscript $\ell = 0, \ldots, d$, with $2^{d-\ell}$ variables $x_j^{(\ell)}$ at level ℓ, such that the $2^d = N$ variables $x_j^{(0)} \equiv x_j$ represent the "foundation" of the "tower", and the variable $x_1^{(d)} \equiv x_{N+1}$ represents its "top".

Observe that the number of three-dimensional p-cones in the "tower-of-variables" representation (18.22) of $(2^d + 1)$-dimensional p-order cone is $2^d - 1 = N - 1$. Further, it is easy to demonstrate that a "tower-of-variables" representation can be constructed for p-order cones of general dimension $N \neq 2^d$, such that the number of resulting three-dimensional p-cones is still $N - 1$.

Proposition 3 *For any integer $N > 2$, a p-order cone in $\mathbb{R}^{N+1}$ can be represented as intersection of $N - 1$ three-dimensional p-order cones.*

Proof For an integer $N > 2$, let

$$N = \sum_{k=0}^{\bar{d}} \pi_k 2^k \tag{18.23}$$

where

$$\bar{d} = \lceil \log_2 N \rceil$$

and $\pi_k \in \{0, 1\}$, i.e., π_k is the k-th digit in the binary representation of the integer N. Let Π_N denote the (ordered) set of those k in (18.23) for which π_k are nonzero:

$$\Pi_N = \{k_0 < k_1 < \cdots < k_{s-1} \mid \delta_{k_i} = 1\}, \quad s = |\Pi_N| = \sum_{k=1}^{\bar{d}} \pi_k \tag{18.24}$$

Then, for each k such that $1 \leq k \leq \bar{d}$ and $\pi_k \neq 0$, the following conic inequalities can be written:

$$x_j^{(\ell)} \geq \left(|x_{2j-1}^{(\ell-1)}|^p + |x_{2j}^{(\ell-1)}|^p\right)^{1/p},$$

$$j = 1 + \sum_{r=k+1}^{\bar{d}} \pi_r 2^{r-\ell}, \ldots, \sum_{r=k}^{\bar{d}} \pi_r 2^{r-\ell}, \ \ell = 1, \ldots, k, \ k \in \Pi_N \backslash \{0\} \tag{18.25}$$

For every $k \in \Pi_N \backslash \{0\}$ expressions (18.25) define a "sub-tower of variables", each having $k + 1$ "levels" including the "foundation" ($\ell = 0$) comprised of 2^k variables $x_j^{(0)}$ and the "top" variable

$$x_{j_k}^{(k)}, \quad \text{where } j_k = \sum_{r=k}^{\bar{d}} \pi_r 2^{r-k} \tag{18.26}$$

To complete our representation of $(N + 1)$-dimensional p-order cone, we must formulate the corresponding conic inequalities that "connect" the "top" variables (18.26). This can be accomplished in a recursive manner as follows

$$x_{\nu_{r+1}}^{(\kappa_{r+1})} \geq \left(|x_{j_{k_r}}^{(k_r)}|^p + |x_{\nu_r}^{(\kappa_r)}|^p\right)^{1/p}, \quad r = 1, \ldots, \sum_{k=0}^{\bar{d}-2} \pi_k \tag{18.27}$$

where

$$\kappa_1 = k_0, \quad \nu_1 = j_{k_0} \quad \text{and}$$

$$\kappa_{r+1} = k_r + 1, \quad \nu_{r+1} = \frac{j_{k_r} + 1}{2} \quad \text{for } r = 1, \ldots, \sum_{k=0}^{\bar{d}-2} \pi_k$$

It is straightforward to verify that the projection of the set defined by (18.25), (18.27) on the space of variables $x_j^{(0)} \equiv x_j$, $(j = 1, \ldots, N)$, $x_1^{(\bar{d})} = x_{N+1}$ is equal to the set (18.21). It is also evident that when $N = 2^d = 2^{\bar{d}}$, the set Π_N will contain just one element: $\Pi_N = \{\bar{d}\}$, whereby (18.25) reduces to (18.22) with cones (18.27) being absent.

Observe that set (18.25) comprises $\sum_{k=1}^{\bar{d}} \pi_k$ "sub-towers", each containing $2^k - 1$ cones, and set (18.27) consists of $\sum_{k=1}^{\bar{d}-2} \pi_k$ cones, therefore the representation (18.25), (18.27) of the p-order cone in $\mathbb{R}^{N+1}$ contains

$$\sum_{k=1}^{\bar{d}} \pi_k (2^k - 1) + \sum_{k=1}^{\bar{d}-2} \pi_k = \sum_{k=1}^{\bar{d}} \pi_k 2^k - \pi_{\bar{d}-1} - \pi_{\bar{d}} = N - 1 \tag{18.28}$$

three-dimensional p-order cones. Indeed,

$$\pi_{\bar{d}-1} + \pi_{\bar{d}} = 1$$

since $\pi_{\bar{d}} = 1$, $\pi_{\bar{d}-1} = 0$ if $N = 2^d = 2^{\bar{d}}$, whereas $\pi_{\bar{d}} = 0$, $\pi_{\bar{d}-1} = 1$ for $N < 2^{\bar{d}}$. $\square$

18.3.3 Polyhedral Approximations of 3-dimensional p-order Cones

The "tower-of-variables" technique discussed in Sect. 18.3.2 reduces the problem of developing a polyhedral approximation (18.19) for the p-order cone $\mathcal{K}_p^{(N+1)}$ in the positive orthant of $\mathbb{R}^{N+1}$ to constructing a polyhedral approximation to the p-cone $\mathcal{K}_p^{(3)}$ in $\mathbb{R}^3$

$$\mathcal{H}_{p,m}^{(3)} = \left\{ \begin{pmatrix} \mathbf{x} \\ \mathbf{u} \end{pmatrix} \in \mathbb{R}_+^{3+\kappa_m} \,\middle|\, \mathbf{H}_{p,m}^{(3)} \begin{pmatrix} \mathbf{x} \\ \mathbf{u} \end{pmatrix} \geq \mathbf{0} \right\} \tag{18.29a}$$

with the approximation quality in (H2) measured as

$$\left(x_1^p + x_2^p\right)^{1/p} \leq \left(1 + \varepsilon(m)\right) x_3 \tag{18.29b}$$

Assuming for simplicity that $N = 2^d + 1$, and the $2^{d-\ell}$ three-dimensional cones at a level $\ell = 1, \ldots, d$ are approximated using (18.29a) with a common approximation error $\varepsilon(m_\ell)$, the approximation error ε of the p-order cone of dimension $2^d + 1$ equals to

$$\varepsilon = \prod_{\ell=1}^{d} \left(1 + \varepsilon(m_\ell)\right) - 1 \tag{18.30}$$

According to the preceding discussion, the cone $\mathcal{K}_p^{(3)}$ is already polyhedral for $p = 1$ and $p = \infty$:

$$\mathcal{K}_1^{(3)} = \left\{ \mathbf{x} \in \mathbb{R}_+^3 \mid x_3 \geq x_1 + x_2 \right\}, \qquad \mathcal{K}_\infty^{(3)} = \left\{ \mathbf{x} \in \mathbb{R}_+^3 \mid x_3 \geq x_1,\ x_3 \geq x_2 \right\}$$

In the case of $p = 2$, the problem of constructing a polyhedral approximation of the second-order cone $\mathcal{K}_2^{(3)}$ was addressed by Ben-Tal and Nemirovski [4], who suggested an efficient polyhedral approximation of $\mathcal{K}_2^{(3)}$,

$$\begin{aligned}
u_0 &\geq x_1, \\
v_0 &\geq x_2, \\
u_i &= \cos\left(\frac{\pi}{2^{i+1}}\right) u_{i-1} + \sin\left(\frac{\pi}{2^{i+1}}\right) v_{i-1}, \quad i = 1, \ldots, m, \\
v_i &\geq \left| -\sin\left(\frac{\pi}{2^{i+1}}\right) u_{i-1} + \cos\left(\frac{\pi}{2^{i+1}}\right) v_{i-1} \right|, \quad i = 1, \ldots, m, \\
u_m &\leq x_3, \\
v_m &\leq \tan\left(\frac{\pi}{2^{m+1}}\right) u_m, \\
0 &\leq u_i, v_i, \quad i = 0, \ldots, m
\end{aligned} \tag{18.31}$$

with an approximation error exponentially small in m:

$$\varepsilon(m) = \frac{1}{\cos(\frac{\pi}{2^{m+1}})} - 1 = O\left(\frac{1}{4^m}\right) \tag{18.32}$$

Their construction is based on a clever geometric argument that utilizes a well-known elementary fact that rotation of a vector in $\mathbb{R}^2$ is an affine transformation that preserves the Euclidian norm (2-norm) and that the parameters of this affine transform depend only on the angle of rotation.

Unfortunately, Ben-Tal and Nemirovski's polyhedral approximation (18.31) of $\mathcal{K}_2^{(3)}$ does not seem to be extendable to values of $p \in (1, 2) \cup (2, \infty)$, since rotation of a vector in $\mathbb{R}^2$ does not preserve its p-norm. Therefore, we adopt a "gradient" approximation of $\mathcal{K}_p^{(3)}$ using circumscribed planes:

$$\hat{\mathcal{H}}_{p,m}^{(3)} = \left\{\mathbf{x} \in \mathbb{R}_+^3 \mid \left(-\alpha_i^{(p)}, -\beta_i^{(p)}, 1\right)\mathbf{x} \geq 0, \quad i = 0, \ldots, m\right\} \tag{18.33a}$$

where

$$\alpha_i^{(p)} = \frac{\cos^{p-1}\frac{\pi i}{2m}}{(\cos^p \frac{\pi i}{2m} + \sin^p \frac{\pi i}{2m})^{\frac{p-1}{p}}}, \qquad \beta_i^{(p)} = \frac{\sin^{p-1}\frac{\pi i}{2m}}{(\cos^p \frac{\pi i}{2m} + \sin^p \frac{\pi i}{2m})^{\frac{p-1}{p}}} \tag{18.33b}$$

The following proposition establishes approximation quality for the uniform gradient approximation (18.33) of the cone $\mathcal{K}_p^{(3)}$.

Proposition 4 *For any integer $m \geq 1$, the polyhedral set $\hat{\mathcal{H}}_{p,m}^{(3)}$ defined by the gradient approximation* (18.33) *satisfies properties* ($H1$)–($H2$). *Particularly, for any $\mathbf{x} \in \mathcal{K}_p^{(3)}$ one has $\mathbf{x} \in \hat{\mathcal{H}}_{p,m}^{(3)}$, and any $(x_1, x_2, x_3) \in \hat{\mathcal{H}}_{p,m}^{(3)}$ satisfies $\|(x_1, x_2)\|_p \leq (1 + \varepsilon(m))x_3$, where*

$$\varepsilon(m) = \begin{cases} O\left(m^{-2}\right), & 2 \leq p < \infty \\ O\left(m^{-p}\right), & 1 \leq p < 2 \end{cases} \tag{18.34}$$

Proof The gradient approximation (18.33) of the p-cone $\mathcal{K}_p^{(3)}$ represents a polyhedral cone in $\mathbb{R}_+^3$ whose facets are the planes tangent to the surface of $\mathcal{K}_p^{(3)}$ Given that the plane tangent to the surface $x_3 = \|(x_1, x_2)\|_p$ at a point $(x_1^0, x_2^0, x_3^0) \in \mathbb{R}_+^3$ has the form

$$\left(x_3^0\right)^{p-1} x_3 = \left(x_1^0\right)^{p-1} x_1 + \left(x_2^0\right)^{p-1} x_2 \tag{18.35}$$

the gradient approximation (18.33) can be constructed using the following parametrization of the conic surface $x_3 = \|(x_1, x_2)\|_p$:

$$\frac{x_1}{x_3} = \frac{\cos\theta}{(\cos^p\theta + \sin^p\theta)^{1/p}}, \qquad \frac{x_2}{x_3} = \frac{\sin\theta}{(\cos^p\theta + \sin^p\theta)^{1/p}} \tag{18.36}$$

where θ is the azimuthal angle of the cylindrical (polar) coordinate system. Then, the property (H1), namely that any $\mathbf{x} \in \mathcal{K}_p^{(3)}$ also satisfies (18.33), follows immediately from the construction of the polyhedral cone (18.33). To demonstrate that (H2) holds, we note that the approximation error $\varepsilon(m)$ in (18.29b) can be taken as the smallest value that for any $\mathbf{x} \in \hat{\mathcal{H}}_{p,m}^{(3)}$ satisfies

$$\varepsilon(m) \geq \left\|(x_1/x_3, x_2/x_3)\right\|_p - 1 \tag{18.37}$$

By introducing $x = x_1/x_3$ and $y = x_2/x_3$, the analysis of approximation of the p-cone $\mathcal{K}_p^{(3)}$ by the gradient polyhedral cone $\hat{\mathcal{H}}_{p,m}^{(3)}$ (18.29) can be reduced to considering an approximation of the set $\mathcal{K}' = \{(x, y) \in \mathbb{R}_+^2 \mid x^p + y^p \leq 1\}$ by the polyhedral set $\mathcal{H}' = \{(x, y) \in \mathbb{R}_+^2 \mid \alpha_i^{(p)} x + \beta_i^{(p)} y \leq 1,\ i = 0, \dots, m\}$. Immediately, we have that $\varepsilon(m)$ in (18.37) is bounded for any integer $m \geq 1$.

To estimate $\varepsilon(m)$ in (29) for large values of m we observe that it achieves (local) maxima at the extreme points of $\mathcal{H}'$. Let (x_i, y_i) and (x_{i+1}, y_{i+1}) be the points of the "p-curve" $x^p + y^p = 1$ that correspond to polar angles $\theta_i = \frac{\pi i}{2m}$ and $\theta_{i+1} = \frac{\pi(i+1)}{2m}$ in (18.33b). Then, a vertex (x_i^*, y_i^*) of $\mathcal{H}'$ located at the intersection of the lines tangent to $\mathcal{K}'$ at these points is given by

$$x_i^* = \frac{y_{i+1}^{p-1} - y_i^{p-1}}{(x_i y_{i+1})^{p-1} - (x_{i+1} y_i)^{p-1}}, \qquad y_i^* = \frac{x_i^{p-1} - x_{i+1}^{p-1}}{(x_i y_{i+1})^{p-1} - (x_{i+1} y_i)^{p-1}} \tag{18.38}$$

and the approximation error $\varepsilon_i(m)$ within the sector $[\frac{\pi i}{2m}, \frac{\pi(i+1)}{2m}]$ is given by

$$\varepsilon_i(m) = \left((x_i^*)^p + (y_i^*)^p\right)^{1/p} - 1 \tag{18.39}$$

It is straightforward to verify that $\varepsilon_i(m) = \varepsilon(m) = \frac{1}{8}(\frac{\pi}{2m})^2$ for $p = 2$. Similarly, for a general $p \neq 2$ and values of m large enough the p-curve within the sector $[\frac{\pi i}{2m}, \frac{\pi(i+1)}{2m}]$ can be approximated by a circular arc with a radius equal to the radius of the curvature of the p-curve, and the approximation error $\varepsilon_i(m)$ within this segment is governed by the corresponding local curvature $\varkappa(\theta)$ of the p-curve $x^p + y^p = 1$.

Given that $\theta = \frac{\pi}{4}$ is the axis of symmetry of $\mathcal{K}'$, for general $p \neq 2$ it suffices to consider the approximation errors $\varepsilon_i(m)$ on the segments $[0, \frac{\pi}{2m}], \dots, [\frac{\pi}{4} - \frac{\pi}{2m}, \frac{\pi}{4}]$, where it can be assumed without loss of generality that the parameter of construction m is an even number.

It is easy to see that the curvature $\varkappa(\theta)$ of the p-curve $x^p + y^p = 1$ of $\mathcal{K}'$ is monotonic in θ when $p \neq 2$, which implies that the approximation error $\varepsilon_i(m)$ on the interval $[\frac{\pi i}{2m}, \frac{\pi(i+1)}{2m}]$ is monotonic in i for $i = 0, \dots, \frac{m}{2} - 1$. Indeed, using the polar parametrization (18.36) of the p-curve $x^p + y^p = 1$, the derivative of its curvature $\varkappa$ with respect to the polar angle θ can be written as

$$\frac{\mathrm{d}}{\mathrm{d}\theta}\varkappa(\theta) = \frac{(p-1)\gamma^{p-1}(1+\gamma^2)(1+\gamma^p)^{\frac{1}{p}}}{(\gamma^2+\gamma^{2p})^2(1+\gamma^{2p-2})^{\frac{1}{2}}} \times \left[(p-2)\left(\gamma^2-\gamma^{3p}\right)+(1-2p)\left(\gamma^{2p}-\gamma^{p+2}\right)\right] \quad (18.40)$$

where $\gamma = \tan\theta \in (0,1)$ for $\theta \in (0, \frac{\pi}{4})$. Then, the sign of $\frac{\mathrm{d}}{\mathrm{d}\theta}\varkappa(\theta)$ is determined by the term in brackets in (18.40); for $p > 2$ we have that $\gamma^2 > \gamma^{3p}$ and $\gamma^{2p} < \gamma^{p+2}$, whence the term in brackets is positive, i.e., $\varkappa(\theta)$ is increasing on $(0, \frac{\pi}{4})$. Similarly, for $1 < p < 2$, one has that the term in brackets in (18.40) is negative, meaning that $\varkappa(\theta)$ is decreasing in θ on $(0, \frac{\pi}{4})$.

Thus, the largest values of $\varepsilon_i(m)$ for $p \neq 2$ are achieved at the intervals $[0, \frac{\pi}{2m}]$ and $[\frac{\pi}{4} - \frac{\pi}{2m}, \frac{\pi}{4}]$. Taking

$$x_0 = 1, \quad y_0 = 0, \quad x_1 = \frac{\cos\frac{\pi}{2m}}{(\cos^p\frac{\pi}{2m} + \sin^p\frac{\pi}{2m})^{1/p}}, \quad y_1 = \frac{\sin\frac{\pi}{2m}}{(\cos^p\frac{\pi}{2m} + \sin^p\frac{\pi}{2m})^{1/p}}$$

and plugging these values into (18.38) and (18.39), we have

$$\varepsilon_0(m) \approx \frac{1}{p}\left(1-\frac{1}{p}\right)^p\left(\frac{\pi}{2m}\right)^p$$

Similarly to above, we obtain that the error at the interval $[\frac{\pi}{4} - \frac{\pi}{2m}, \frac{\pi}{4}]$ is

$$\varepsilon_{m/2-1}(m) \approx \frac{1}{8}(p-1)\left(\frac{\pi}{2m}\right)^2$$

Thus, for $p \in (1,2)$ we have $\varepsilon(m) = \max_i \varepsilon_i(m) = O(m^{-p})$, and for $p \in (2,\infty)$ the approximation accuracy satisfies $\varepsilon_m = \max_i \varepsilon_i(m) = \varepsilon_{m/2-1}(m) = O(m^{-2})$. $\square$

The gradient polyhedral approximation (18.33) of the p-cone $\mathcal{K}_p^{(3)}$ requires much larger number of facets than Ben-Tal and Nemirovski's approximation (18.31) of the quadratic cone $\mathcal{K}_2^{(3)}$ to achieve the same level of accuracy. On the other hand, the approximating LP based on the gradient approximation (18.33) has a much simpler structure, which makes its computational properties comparable with those of LPs based on lifted Ben-Tal and Nemirovski's approximation on the problems of smaller dimensionality that are considered in the case study.

18.4 Case Study

We test the developed p-norm separation model and the corresponding solution techniques on several real-world data sets from UCI Machine Learning Repository (University of California-Irvine). In particular, we compare the classification accuracy of the 1-norm model of Bennett and Mangasarian [5] versus p-norm model on the Wisconsin Breast Cancer data set used in the original paper [5].

The algorithm for p-norm linear discrimination model 18.9 has been implemented in C++, and ILOG CPLEX 10.0 solver has been used to solve the its LP approximation as described Sect. 18.3. The approximation accuracy has been set at 10^{-5}.

Wisconsin Breast Cancer Data Set This breast cancer database was obtained from the University of Wisconsin Hospitals by Dr. William H. Wolberg. Each entry in the data set is characterized by an ID number and 10 feature values, which were obtained by medical examination on certain breast tumors. The data set contains a total of 699 data points (records), but because some values are missing, only 682 data points are used in the experiment. The entire data set is comprised of two classes of data points: 444 (65.1%) data points represent benign tumors, and the rest of 238 (34.9%) points correspond to malignant cases.

To test the classification performance of the proposed p-norm classification model, we partitioned the original data set at random into the training and testing sets in the ratio of 2:1, such that the proportion between benign and malignant cases would be preserved. In other words, training set contained $2/3$ of benign and malignant points of the entire data set, and the testing set contained the remaining $1/3$ of benign and malignant cases. The p-norm discrimination model (i.e., its LP approximation) was solved using the training set as the data $\mathbf{A}$ and $\mathbf{B}$, and the obtained linear separator was then used to classify the points in the testing set. For each fixed value of p in (18.9), this procedure has been repeated 10 times, and the average misclassification errors rates for benign and malignant cases have been recorded. The cumulative misclassification rate was then computed as a weighted (0.651 to 0.349) average of the benign and malignant error rates (note that the weights correspond to the proportion of benign and malignant points in the entire database).

The value of the parameter p has been varied from $p = 1.0$ to $p = 5.0$ with 0.1 step. Then, an "optimal" value of the parameter p has been selected that delivered the lowest cumulative average misclassification rates.

In addition to varying the parameter p, we have considered different weights δ_1, δ_2 in the p-norm linear separation model (18.9). In particular, the following combinations have been used:

$$\delta_1 = k^p,\ \delta_2 = m^p, \qquad \delta_1 = k^{p-1},\ \delta_2 = m^{p-1}, \qquad \delta_1 = 1,\ \delta_2 = 1.$$

Finally, the same method was used to compute cumulative misclassification rates for $p = 1$, or the original model of Bennett and Mangasarian [5]. Table 18.1 displays the results of our computational experiments. It can be seen that application of higher norms allows one to reduce the misclassification rates.

Other Data Sets Similar tests have been run also run on Pima Indians Diabetes data set, Connectionist Bench (Sonar, Mines vs. Rocks) data set, and Ionosphere data set, all of which can be obtained from UCI Machine Learning Repository. Note that for these tests only $\delta_1 = \delta_2 = 1$ and $\delta_1 = k^p$, $\delta_2 = m^p$ weights in problem (18.9) are used. Table 18.2 reports the best average error rates obtained under various values of p for these three data sets, and compares them with the best results known in the literature for these particular data sets.

Table 18.1 Classification results of the p-norm separation model for the Wisconsin breast cancer data set

	$\delta_1 = k^p, \delta_2 = m^p$	$\delta_1 = k^{p-1}, \delta_2 = m^{p-1}$	$\delta_1 = \delta_2 = 1$
Optimal value of parameter p	2.0	1.9	1.5
Average cumulative error	3.22%	2.77%	2.82%
Improvement over $p = 1$ model	8.78%	3.48%	1.74%

Table 18.2 Classification results of the p-norm separation model for the other data sets

	$\delta_1 = k^p, \delta_2 = m^p$	$\delta_1 = \delta_2 = 1$	Best results known in the literature[1]
Ionosphere	16.07%	17.35%	12.3%
Pima	26.14%	28.98%	26.3%
Sonar	28.43%	27.83%	24%

[1]The results are based on [14, 16], and algorithms other than linear discrimination methods

18.5 Conclusions

We proposed a new p-norm linear discrimination model that generalizes the model of Bennett and Mangasarian [5] and reduces to linear programming problems with p-order conic constraints. It has been shown that the developed model allows one to finely tune preferences with regard to misclassification rates for different sets. In addition, it has been demonstrated that, similarly to the model of Bennett and Mangasarian [5], the p-norm separation model does not produce a separating hyperplane with null normal for linearly separable sets; a hyperplane with null normal can occur only in situation when the sets to be discriminated exhibit a particular form of linear dependence.

The computational procedures for handling p-order conic constraints rely on constructed polyhedral approximations of p-order cones, and thus reduce the p-norm separation models to linear programming problems.

Acknowledgement The authors would like to acknowledge support of the Air Force Office of Scientific Research.

References

1. Alizadeh, F., Goldfarb, D.: Second-order cone programming. Math. Program. **95**, 3–51 (2003)
2. Andersen, E.D., Roos, C., Terlaky, T.: On implementing a primal-dual interior-point method for conic quadratic optimization. Math. Program. **95**, 249–277 (2003)
3. Ben-Tal, A., Nemirovski, A.: Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications. MPS/SIAM Series on Optimization, vol. 2. SIAM, Philadelphia (2001)

4. Ben-Tal, A., Nemirovski, A.: On polyhedral approximations of the second-order cone. Math. Oper. Res. **26**, 193–205 (2001)
5. Bennett, K.P., Mangasarian, O.L.: Robust linear programming separation of two linearly inseparable sets. Optim. Methods Softw. **1**, 23–34 (1992)
6. Burer, S., Chen, J.: A p-cone sequential relaxation procedure for 0-1 integer programs. Working paper (2008)
7. Glineur, F., Terlaky, T.: Conic formulation for l_p-norm optimization. J. Optim. Theory Appl. **122**, 285–307 (2004)
8. Krokhmal, P.: Higher moment coherent risk measures. Quant. Finance **4**, 373–387 (2007)
9. Krokhmal, P.A., Soberanis, P.: Risk optimization with p-order conic constraints: A linear programming approach. Eur. J. Oper. Res. (2009). doi:10.1016/j.ejor.2009.03.053
10. Nesterov, Y.: Towards nonsymmetric conic optimization. CORE Discussion Paper No. 2006/28 (2006)
11. Nesterov, Y.E., Nemirovski, A.: Interior Point Polynomial Algorithms in Convex Programming. Studies in Applied Mathematics, vol. 13. SIAM, Philadelphia (1994)
12. Nesterov, Y.E., Todd, M.J.: Self-scaled barriers and interior-point methods for self-scaled cones. Math. Oper. Res. **22**, 1–42 (1997)
13. Nesterov, Y.E., Todd, M.J.: Primal-dual interior-point methods for self-scaled cones. SIAM J. Optim. **8**, 324–364 (1998)
14. Radivojac, P., Obradovic, Z., Dunker, A., Vucetic, S.: Feature selection filters based on the permutation test. In: 15th European Conference on Machine Learning, pp. 334–345 (2004)
15. Sturm, J.F.: Using SeDuMi 1.0x, a MATLAB toolbox for optimization over symmetric cones. Manuscript (1998)
16. Tan, P., Dowe, D.: MML inference of decision graphs with multi-way joins. In: Lecture Notes in Artificial Intelligence, vol. 2557, pp. 131–142. Springer, Berlin (2004)
17. Terlaky, T.: On l_p programming. Eur. J. Oper. Res. **22**, 70–100 (1985)
18. Vielma, J.P., Ahmed, S., Nemhauser, G.L.: A lifted linear programming branch-and-bound algorithm for mixed integer conic quadratic programs. INFORMS J. Comput. **20**, 438–450 (2008)
19. Xue, G., Ye, Y.: An efficient algorithm for minimizing a sum of p-norms. SIAM J. Optim. **10**, 551–579 (2000)

Chapter 19
Local Neighborhoods for the Multidimensional Assignment Problem

Eduardo L. Pasiliao Jr.

Summary The Multidimensional Assignment Problem (MAP) is an extension of the two-dimensional assignment problem in which we find an optimal matching of elements between mutually exclusive sets. Although the two-dimensional assignment problem is solvable in polynomial time, extending the problem to three dimensions makes it $\mathcal{NP}$-complete. The computational time to find an optimal solution of an MAP with at least three dimensions grows exponentially with the number of dimensions and factorially with the dimension size. Perhaps the most difficult implementation of the MAP is the data association problem that arises in multisensor multitarget tracking. We define new local search neighborhoods using the permutation formulation of the multidimensional assignment problem, where the feasible domain is defined by permutation vectors. Two types of neighborhoods are discussed, the intrapermutation and the interpermutation k-exchange. If the exchanges are restricted to elements within a single permutation vector, we classify the moves as intrapermutation. Interpermutation exchanges move elements from one permutation vector to another. Since combinatorial optimization heuristics tend to get trapped in local minima, we also discuss variable neighborhood implementations based on the new local search neighborhoods.

19.1 Introduction

Given a batch of n jobs and a group of n workers, the assignment problem gives each worker a job, so that all jobs are performed proficiently. This is a linear two-dimensional assignment and is the most basic type of assignment problem. We now formally pose the job-to-worker problem.

Let us denote the cost of worker i performing job j as c_{ij}. The binary decision variable x_{ij} is defined as

$$x_{ij} = \begin{cases} 1 & \text{if worker } i \text{ is assigned job } j \\ 0 & \text{otherwise} \end{cases}$$

E.L. Pasiliao Jr. (✉)
AFRL Munitions Directorate, Eglin AFB, FL 32542, USA
e-mail: pasiliao@eglin.af.mil

The assignment problem assigns each job to a worker so that all the jobs are done with minimum total cost. We formulate this problem as a 0-1 integer program below.

$$\begin{aligned} \min \sum_{i}^{n}\sum_{j}^{n} c_{ij}\cdot x_{ij} & \\ \text{s.t.} \quad \sum_{j}^{n} x_{ij} = 1 \quad & \forall i = 1,2,\ldots,n \\ \sum_{i}^{n} x_{ij} = 1 \quad & \forall j = 1,2,\ldots,n \end{aligned} \tag{19.1}$$

The constraints guarantee that only one worker is assigned to each job; and that all jobs are accomplished.

Since each worker may only perform a single job, we may denote $\phi(i)$ as the job assigned to worker i. We guarantee that each job is assigned to only one worker by requiring ϕ to be a permutation vector. The assignment problem is now described by the following permutation formulation:

$$\begin{aligned} \min \sum_{i}^{n} c_{i\phi(i)} & \\ \text{s.t.} \quad \phi(i) \neq \phi(j) \quad & \forall j \neq i \\ \phi(i) \in \{1,2,\ldots,n\} & \end{aligned} \tag{19.2}$$

The 0-1 integer programming and permutation formulation are equivalent, but they offer different approaches to finding an optimal assignment.

When the number of dimensions in an assignment problem is greater than two, the problem is referred to as a multidimensional assignment. A three-dimensional example would be the problem of assigning jobs, workers, and machines. Multidimensional assignment problems are often used to model data association problems. An example would be the multitarget multisensor tracking problem, described by Blackman [4], which finds an optimal assignment of sensor measurements to targets.

Surveys of multidimensional assignment problems are provided by Gilbert and Hofstra [8], Burkard, and Çela [6], and most recently by Spieksma [15]. An excellent collection of articles on multidimensional and nonlinear assignment problems is provided by Pardalos and Pitsoulis [14]. Çela [7] provides a short introduction to assignment problems and their applications; and Burkard and Çela [5] give an annotated bibliography.

This paper is organized as follows. Section 19.2 describes the local search heuristic that is used in searching the different local neighborhoods. Section 19.2.1 discusses the intrapermutation exchange neighborhoods. We also present heuristics for searching the intrapermutation 2- and n-exchange neighborhoods. Section 19.2.2 discusses the interpermutation exchange neighborhoods. The expanded neighborhoods for cases where the dimension sizes are nonhomogenous are also presented. Extensions of the neighborhood definitions, including path-relinking, variable depth, and variable neighborhood, are discussed in Sect. 19.3. Finally, the concluding remarks are given in Sect. 19.4. This paper studies the different local search heuristics that are easily applied to the multidimensional assignment problem as used to model the data association segment of the multitarget multisensor tracking problem. It gives an analysis of the computational complexities and quality of results from the different neighborhood definitions.

19.2 Neighborhoods

We define new local search neighborhoods using the permutation formulation of the multidimensional assignment problem. Two types of neighborhoods are discussed, the intrapermutation and the interpermutation k-exchange. If the exchanges are restricted to elements within a single permutation vector, we classify the moves as intrapermutation. Interpermutation exchanges move elements from one permutation vector to another. Since combinatorial optimization heuristics tend to get trapped in local minima, we also discuss variable neighborhood implementations based on the new local search neighborhoods.

This section defines different types of k-exchange neighborhoods that may be searched in an effort to try to improve a feasible solution. To define local search neighborhoods, we look to the following MAP formulation, where $\phi_0, \phi_1, \ldots, \phi_{M-1}$ are permutation vectors:

$$\begin{aligned} &\min \sum_{i=1}^{n_0} c_{\phi_0(i),\phi_1(i),\phi_2(i),\ldots,\phi_{M-1}(i)} \\ &\text{where} \quad \phi_m(i) \in \{1,2,\ldots,n_m\} \quad \forall m = 1,2,\ldots,M-1 \\ &\qquad\qquad \phi_m(i) \neq \phi_m(j) \quad \forall i \neq j \\ &\qquad\qquad n_0 \leq \min_m n_m \quad \forall m = 1,2,\ldots,M-1 \end{aligned} \tag{19.3}$$

The standard permutation formulation fixes the first permutation vector ϕ_0 which results in only $M-1$ permutation vectors. We choose not to fix ϕ_0 since doing so would reduce some of the neighborhood sizes that we define in the following subsections.

Figure 19.1 presents the general local search heuristic for any defined neighborhood. The procedure stores the best solution and defines a new neighborhood each a time a better solution is found. Every solution in the neighborhood is compared to the current best solution. If no improvement is found within the last neighborhood defined, the local search procedure returns the local minimum. A discussion of various local search techniques for combinatorial optimization problems is given by Ibaraki and Yagiura [16] and by Lourenço, Martin, and Stützle [13].

Fig. 19.1 Local search

```
PROCEDURE LocalSearch($S$)
 1  $S_{Best} \leftarrow S$
 2  $NS \leftarrow$ Neighborhood($S_{Best}$)
 3  WHILE $NS \neq \emptyset \mapsto$
 4      $S \leftarrow$ SelectSolution($NS$)
 5      $NS \leftarrow NS \setminus \{S\}$
 6      IF Obj($S$) < Obj($S_{Best}$) $\mapsto$
 7          $S_{Best} \leftarrow S$
 8          $NS \leftarrow$ Neighborhood($S_{Best}$)
 9      END
10  END
11  RETURN($S_{Best}$)
END LocalSearch
```

We will now explore intrapermutation and interpermutation k-exchange neighborhoods. From the permutation formulation of the multidimensional assignment problem, we are able to perform unique types of k-exchanges that would be difficult to implement on any other combinatorial optimization problem. The permutation formulation unique in how it can be easily reduced to a lower dimensional assignment problem by simply fixing some of the permutation vectors. One of these is the intrapermutation n-exchange, which would normally have a complexity of $\mathcal{O}(n!)$. Every possible permutation of a vector with n elements would have to be calculated. However, because of the permutation formulation, we are able to explore the n-exchange neighborhood with a complexity of only $\mathcal{O}(n^3)$. We are also able to perform interpermutation exchanges in which entire permutation vectors are exchanged in one move. We explore and present experimental results for the different types of k-exchanges. Variable depth interchange, path relinking, and variable neighborhoods, which are extensions of the k-exchange neighborhoods, are discussed in Sect. 19.3.

19.2.1 Intrapermutation Exchanges

Here, we discuss the first case of intrapermutation exchanges where only the elements that are already in the current solution are candidates for exchange. Although $n_0 \leq n_m \; \forall m = 1, 2, \ldots, M-1$, the neighborhoods defined here only allow exchanges between elements that are currently in the solution. The maximum number of assignments in an MAP is equivalent to the size of the smallest dimension, n_0. There are $n_m - n_0$ elements in each dimension that are not in the solution and, therefore, do not affect the objective function.

Let ϕ be a permutation vector of n_0 elements. If we define ψ as another permutation of the same elements from ϕ, then the set of all differences between these two distinct permutation vectors is given as

$$\delta_{n_0}(\phi, \psi) = \left\{ i \; : \; \phi(i) \neq \psi(i), \; \forall \, i = 1, 2, \ldots, n_0 \right\} \tag{19.4}$$

Define the distance between ϕ and ψ as

$$d_{n_0}(\phi, \psi) = \left|\delta_{n_0}(\phi, \psi)\right| \tag{19.5}$$

We can now define a k-exchange neighborhood for any one of the M permutation vectors, ϕ_m, of n_0 elements that constitute a feasible solution to the MAP. This neighborhood includes all n_0-permutations, $N_{k,n_0}(\phi_m)$, such that the distance between ϕ_m and any $\psi_m \in N_{k,n_0}(\phi_m)$ is no greater than the value of k,

$$N_{k,n_0}(\phi_m) = \left\{\psi_m \ : \ d_{n_0}(\phi_m, \psi_m) \le k; \ 0 \le k \le n_0\right\} \tag{19.6}$$

Continuing this approach to a set of permutations, we naturally define the k-exchange neighborhood for $\Phi = \{\phi_0, \phi_1, \dots, \phi_{M-1}\}$ as

$$N^{\mathrm{I}}_{k,n_0}(\Phi) = \left\{\Psi \ : \ d_{n_0}(\phi_m, \psi_m) \le k; \ \phi_m \in \Phi, \ \psi_m \in \Psi\right\} \tag{19.7}$$

where Ψ is a set of M permutation vectors corresponding to the original set Φ. Note that the size of the k-exchange neighborhood for an M-dimensional assignment problem with $M-1$ permutation vectors is given by

$$\left|N^{\mathrm{I}}_{k,n_0}(\Phi)\right| = \prod_{m=0}^{M-1} \left|N_{k,n_0}(\phi_m)\right| \tag{19.8}$$

Although we normally fix the first permutation ϕ_0 in the MAP formulation, the product above is applied for $m \ge 0$. Performing exchanges on ϕ_0 produces additional feasible solutions since the exchanges are limited to $k < n_0$. The neighborhood size for the MAP using this definition of a neighborhood grows exponentially with the number of dimensions.

If we define the set of all differences between two sets of permutation vectors Φ and Ψ as

$$\Delta_M(\Phi, \Psi) = \left\{m \ : \ d_{n_0}(\phi_m, \psi_m) > 0, \ \forall m = 1, 2, \dots, M; \ \phi_m \in \Phi, \ \psi_m \in \Psi\right\} \tag{19.9}$$

then the distance between Φ and Ψ is given by

$$D_M(\Phi, \Psi) = \left|\Delta_M(\Phi, \Psi)\right| \tag{19.10}$$

We can now define another k-exchange neighborhood for a set of permutation vectors as

$$N^{\mathrm{II}}_{k,n_0}(\Phi) = \left\{\Psi \ : \ d_{n_0}(\phi_m, \psi_m) \le k, \ D_M(\Phi, \Psi) \le 1; \ \phi_m \in \Phi, \ \psi_m \in \Psi\right\} \tag{19.11}$$

This neighborhood is much smaller than $N^{\mathrm{II}}_{k,r}(\Phi)$ because it limits the number of k-exchanges to one permutation vector at a time. The previous neighborhood definition allows k-exchanges in multiple permutation vectors. Note that the size of the neighborhood for an M-dimensional assignment problem in which only a single k-exchange is allowed at a time is given by

$$\left|N_{k,n_0}^{\mathrm{II}}(\Phi)\right| = \sum_{m=0}^{M-1} \left|N_{k,n_0}(\phi_m)\right| \tag{19.12}$$

The neighborhood size for the MAP using the second definition of a neighborhood only grows linearly with the number of dimensions.

The following relationships hold between the two k-exchange neighborhood definitions:

$$N_{k,n_0}^{\mathrm{II}}(\Phi) \subseteq N_{k,n_0}^{\mathrm{I}}(\Phi) \longrightarrow \left|N_{k,n_0}^{\mathrm{II}}(\Phi)\right| \leq \left|N_{k,n_0}^{\mathrm{I}}(\Phi)\right|$$

All single vector exchange neighborhoods are also included in the multiple vector k-exchange neighborhoods.

19.2.1.1 2-Exchange

Although a large k-exchange neighborhood may produce better solutions, it also incurs added complexity. The extra computation time may or may not be justified depending on the time necessary to obtain a new initial feasible solution. If it takes a considerable amount of time to produce a good initial solution, then spending more time searching a larger neighborhood would be the preferred approach. However, if a new initial solution may be found in a relatively short amount of time, it would be better to limit the neighborhood to a small size. Because of the computational difficulty in implementing exchanges for $k > 2$ elements, we focus this section on 2-exchange neighborhoods.

The 2-exchange neighborhood for a permutation vector ϕ_m is defined as

$$N_{2,n_0}^{\mathrm{I}}(\Phi) = \left\{\Psi \ : \ d_{n_0}(\phi_m, \psi_m) \leq 2; \ \phi_m \in \Phi, \ \psi_m \in \Psi\right\} \tag{19.13}$$

Note that the size of the 2-exchange neighborhood for a single permutation vector is given by

$$\left|N_{2,n_0}(\phi_m)\right| = \binom{n_0}{2} \tag{19.14}$$

which is simply the number of ways to choose unordered pairs from a set of n_0 elements. The size of a 2-exchange neighborhood for an M-dimensional assignment problem with $M-1$ permutation vectors is given by

$$\left|N_{2,n_0}^{\mathrm{I}}(\Phi)\right| = \binom{n_0}{2}^M \tag{19.15}$$

for a neighborhood that allows a 2-exchange in more than one permutation vector and

$$\left|N_{2,n_0}^{\mathrm{II}}(\Phi)\right| = M \times \binom{n_0}{2} \tag{19.16}$$

```
PROCEDURE 2-Exchange($S, C$)
    $S_{\text{Best}} \Leftarrow S$
    FOR $m = 0 : M - 1 \mapsto$
        FOR $i_1 = 1 : n_0 - 1 \mapsto$
            FOR $i_2 = i_1 + 1 : n_0 \mapsto$
                $old \Leftarrow c_{\phi_1(i_1),\phi_2(i_1),\ldots,\phi_{M-1}(i_1)} + c_{\phi_1(i_2),\phi_2(i_2),\ldots,\phi_{M-1}(i_2)}$
                $\phi_m(i_1) \Leftrightarrow \phi_m(i_2)$
                $new \Leftarrow c_{\phi_1(i_1),\phi_2(i_1),\ldots,\phi_{M-1}(i_1)} + c_{\phi_1(i_2),\phi_2(i_2),\ldots,\phi_{M-1}(i_2)}$
                IF $new < old \mapsto$
                    $S_{\text{Best}} \Leftarrow \{\phi_0, \phi_1, \ldots, \phi_{M-1}\}$
                    RETURN($S_{\text{Best}}$)
                $\phi_m(i_1) \Leftrightarrow \phi_m(i_2)$
            END
        END
    END
    RETURN($S_{\text{Best}}$)
END LocalSearch
```

Fig. 19.2 Intrapermutation 2-exchange local search

for a neighborhood that is limited to a single 2-exchange at a time.

Figure 19.2 presents the implementation of the intrapermutation 2-exchange neighborhood. The heuristic permutes two elements at a time for each dimension until all possible exchanges have been explored. If an improvement is found, this solution is then stored as the current solution and a new neighborhood is defined and searched. When no better solution can be found, the search is terminated and the local minimum is returned.

19.2.1.2 *n*-Exchange

Extending the intrapermutation k-exchange neighborhood to its limit, we present the following neighborhood definition:

$$N_{n_0}(\phi_m) = \left\{\psi_m \; : \; d_{n_0}(\phi_m, \psi_m) \leq n_0; \; 0 \leq n_0 \leq n_m \; \forall m > 1\right\} \tag{19.17}$$

Note that the size of the n_0-exchange neighborhood for a single permutation vector is given by

$$\left|N_{n_0,n_0}(\phi_m)\right| = n_0! \tag{19.18}$$

which is simply the number of ways to order n_0 elements.

The first n_0-exchange neighborhood for a set of permutation vectors, Φ, is given by

$$N^{\mathrm{I}}_{n_0}(\Phi) = \left\{\Psi \; : \; d_{n_0}(\phi_m, \psi_m) \leq n_0; \; \phi_m \in \Phi, \; \psi_m \in \Psi\right\} \tag{19.19}$$

where Ψ is a set of M permutations vectors corresponding to the original set Φ. The size of this n_0-exchange neighborhood for an M-dimensional assignment problem with $M-1$ permutation vectors is given by

$$\left|N^{\mathrm{I}}_{n_0,n_0}(\Phi)\right| = \prod_{m=1}^{M-1} \left|N_{n_0}(\phi_m)\right| = [n_0!]^{M-1} \tag{19.20}$$

The product above is applied for $m > 1$ since we may fix the first permutation ϕ_0, without decreasing the neighborhood size. This is different from the k-exchange, where $k < n_0$. If $k = n_0$, all the feasible solutions may be found without manipulating the ϕ_0.

The second neighborhood definition for a set of permutation vectors is given by

$$N^{\mathrm{II}}_{n_0}(\Phi) = \left\{\Psi \ : \ d_{n_0}(\phi_m, \psi_m) \le n_0, \ d_M(\Phi, \Psi) \le 1; \ \phi_m \in \Phi, \ \psi_m \in \Psi\right\} \tag{19.21}$$

This n_0-exchange neighborhood has a size of

$$\left|N^{\mathrm{II}}_{n_0}(\Phi)\right| = \sum_{m=0}^{M-1} \left|N_{n_0}(\phi_m)\right| = M \times [n_0!] \tag{19.22}$$

The summation here is applied for $m \ge 0$, which is different from the previous definition of an n-exchange neighborhood. The difference is that we are only able to manipulate one permutation at a time. In the previous definition, we were allowed to manipulate multiple permutations simultaneously. Because of this difference, we may not fix the first permutation ϕ_0 without reducing the neighborhood size.

Figure 19.3 describes the local search procedure to improve the current solution by searching within its predefined neighborhood. The heuristic performs an n_0-exchange by fixing $M-1$ permutation vectors and solving for the single decision vector with a two-dimensional assignment problem solver. A 2AP is solved for

```
PROCEDURE $n$-Exchange($S$, $C$)
1   $S_{\text{Best}} \Leftarrow S$
2   FOR $m = 0 : M-1 \mapsto$
3       $S \Leftarrow S \setminus \phi_m$
4       FOR $i_1 = 1 : n_0 \mapsto$
5           FOR $i_2 = 1 : n_0 \mapsto$
6               $A_{i_1,i_2} \Leftarrow c_{\phi_1(i_1),\ldots,\phi_{m-1}(i_1),\, i_2,\phi_{m+1}(i_1),\ldots,\phi_{M-1}(i_1)}$
7           END
8       END
9       $\phi_m \leftarrow$ Solve2AP($A$)
10      $S \Leftarrow S \cup \phi_m$
11      IF Cost($S$) $<$ Cost($S_{\text{Best}}$) $\mapsto$
12              $S_{\text{Best}} \Leftarrow S$
13              RETURN($S_{\text{Best}}$)
14  END
15  RETURN($S_{\text{Best}}$)
END LocalSearch
```

Fig. 19.3 Intrapermutation n-exchange local search

$$\Phi = \{\phi_0, \phi_1, \phi_2, \phi_3\}$$

Permute ϕ_0

$$x^*_{ij} \leftarrow \min\left\{\sum_{ij} c_{i,\phi_1(j),\phi_2(j),\phi_3(j)} \cdot x_{ij} \ : \ \sum_j x_{ij} = \sum_i x_{ij} = 1\right\}$$

$\phi_0(j) \Leftarrow i \quad$ for $x^*_{ij} = 1 \ \forall i, j$

Permute ϕ_1

$$x^*_{ij} \leftarrow \min\left\{\sum_{ij} c_{\phi_0(j),i,\phi_2(j),\phi_3(j)} \cdot x_{ij} \ : \ \sum_j x_{ij} = \sum_i x_{ij} = 1\right\}$$

$\phi_1(j) \Leftarrow i \quad$ for $x^*_{ij} = 1 \ \forall i, j$

Permute ϕ_2

Permute ϕ_3

Fig. 19.4 Intrapermutation n-exchange example, $C \in \Re^{3\times4\times4\times5}$

each dimension of the MAP until either a better solution is found or all the dimensions are explored. If an improvement is found, this solution is then stored as the current solution and a new neighborhood is defined and searched. When no better solution is available, the search is terminated and the local minimum is returned.

An example of how the n-exchange neighborhood for $C \in \Re^{3\times4\times4\times5}$ is given in Fig. 19.4. The coefficients of the 2AP are found by fixing $M - 1 = 3$ permutation vectors or dimensions at a time. Therefore, a 2AP solver must be implemented for each dimension.

19.2.2 Interpermutation Exchanges

This section discusses exchanges that are performed between the permutation vectors of the MAP permutation formulation. By defining a limited number of exchanges that may be made between the elements of a permutation vector, we can define a local search neighborhood.

So far we have only investigated k-exchanges within a given permutation vector ϕ_m. We now explore the possibility of finding better solutions by interchanging the indices themselves. This section discusses the first case of interpermutation exchanges; all the dimensions in the MAP have exactly the same size. Again we use the permutation formulation of the MAP with M permutation vectors.

If we define the set of all differences between two sets of permutation vectors Φ and Ψ as

$$\Delta_M(\Phi,\Psi) = \left\{m \ : \ d_{n_0}(\phi_m,\psi_m) > 0, \ \forall m = 1,2,\ldots,M; \ \phi_m \in \Phi, \ \psi_m \in \Psi\right\} \tag{19.23}$$

where $d_{n_0}(\phi_m,\psi_m)$ is the distance between permutation vectors ϕ_m and ψ_m, then the distance between Φ and Ψ is given by

$$D_M(\Phi,\Psi) = \left|\Delta_M(\Phi,\Psi)\right| \tag{19.24}$$

We can now define an interpermutation k-exchange neighborhood for Φ as the set of all permutation vector sets such that the distance between Φ and any $\Psi \in N_{k,M}(\Phi)$ is no greater than the value of k,

$$N_{k,M}(\Phi) = \left\{\Psi \ : \ d_M(\Phi,\Psi_m) \le k; \ 0 < k < M\right\} \tag{19.25}$$

Note that the size of the 2-exchange neighborhood is given by

$$\left|N_{2,M}(\Phi)\right| = \binom{M}{2} \tag{19.26}$$

which is simply the number of ways to choose unordered pairs from a set of M permutation vectors.

The pseudo-code for performing an interpermutation standard 2-exchange is shown in Fig. 19.5. The procedure explores every possible 2-exchange between permutations until a better solution is found. Lines 6–9 perform the exchanges between two permutation vectors, while lines 10–13 check if the exchange produced a better solution. If no improvement is found, then we find another pair of vectors to exchange.

The standard interpermutation 2-exchange is implemented in a straightforward manner when all of the dimensions in the MAP have the same size. However, it is often the case when the dimension sizes are not homogeneous. For this situation, we need a method for deciding which elements of the larger permutation vectors will be included into and removed from the current solution.

Performing 2-exchanges between two dimensions of the same size is straightforward since every element in the first dimension is feasible in the second dimension considered. The difficulty arises when the dimensions have different sizes. When $n_0 < n_m$ for some dimension m, then exchanging one permutation for another may produce infeasible results. As an example, suppose that we have a four-dimensional assignment problem with cost array $C \in \Re^{3\times4\times4\times5}$. A current feasible solution may be given as the following:

$$\begin{array}{cccc} \phi_0 & \phi_1 & \phi_2 & \phi_3 \\ \begin{bmatrix}1\\2\\3\end{bmatrix} & \begin{bmatrix}1\\4\\2\end{bmatrix} & \begin{bmatrix}2\\4\\1\end{bmatrix} & \begin{bmatrix}1\\3\\5\end{bmatrix} \end{array}$$

$$f = c_{1121} + c_{2443} + c_{3215}$$

```
PROCEDURE 2-Exchange($S$, $C$)
1   $S_{\text{Best}} \Leftarrow S$
2   FOR $m_1 = 0 : M-2 \mapsto$
3       $S \Leftarrow S \setminus \phi_{m_1}$
4       FOR $m_2 = m_1 + 1 : M-1 \mapsto$
5           $S \Leftarrow S \setminus \phi_{m_2}$
6           FOR $i = 1 : n_0 \mapsto$
7               $\psi_{m_1}(i) \Leftarrow \phi_{m_2}(i)$
8               $\psi_{m_2}(i) \Leftarrow \phi_{m_1}(i)$
9           END
10          IF Cost($S \cup \psi_{m_1} \cup \psi_{m_2}$) $<$ Cost($S_{\text{Best}}$) $\mapsto$
11              $S_{\text{Best}} \Leftarrow S \cup \psi_{m_1} \cup \psi_{m_2}$
12              RETURN($S_{\text{Best}}$)
13          END
14          $S \Leftarrow S \cup \phi_{m_2}$
15      END
16      $S \Leftarrow S \cup \phi_{m_1}$
17  END
18  RETURN($S_{\text{Best}}$)
END LocalSearch
```

Fig. 19.5 Interpermutation standard 2-exchange local search

If we perform an exchange between permutation vectors ϕ_1 and ϕ_2, where $n_1 = n_2$, then we have a standard interpermutation 2-exchange with a resulting change in the objective function.

$$\begin{array}{cccc} \phi_0 & \phi_2 & \phi_1 & \phi_3 \\ \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} & \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix} & \begin{bmatrix} 1 \\ 4 \\ 2 \end{bmatrix} & \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \end{array}$$
$$f = c_{1211} + c_{2443} + c_{3125}$$

The cost coefficients c_{1121} and c_{3215} have been removed from the solution set and replaced with c_{1211} and c_{3125}.

The interpermutation 2-exchange is smaller than its intrapermutation counterpart for most MAP instances with the exception occurring when the sizes are much smaller than the number of dimensions. However, because of the added complexity in implementing the intrapermutation 2-exchange, the interpermutation 2-exchange is a more efficient neighborhood to search since only two cost elements are analyzed every time an exchange is made. The number of reassignments is also smaller for the intrapermutation version. For the interpermutation 2-exchange, all n_0 feasible cost elements are analyzed for each exchange.

19.3 Extensions

The main types of local neighborhood search applied to multidimensional assignment problems are k-exchanges. Extensions of these neighborhoods include variable neighborhood, variable depth interchange, and path relinking. We outline the MAP implementation procedures for all three extensions in this section.

19.3.1 Variable Depth Interchange

Balas and Saltzman [3] used a variable depth interchange neighborhood in their branch-and-bound algorithm for the three-index assignment problem. It is based on the heuristic applied by Lin and Kernighan [12] on the traveling salesman problem and described by Ahuja, Ergun, Orlin, and Punnen [1] in their survey of neighborhood search algorithms. Figure 19.6 presents a pseudo-code of variable depth interchange for multidimensional assignment problems. The heuristic can escape local minima by allowing moves to nonimproving solutions until a threshold is reached. There is, however, no guarantee of escaping any local minima for this problem.

19.3.2 Path Relinking

Path relinking is a strategy proposed by Laguna and Marti [11] to connect multiple high-quality solutions in an effort to find an even better solution. Aiex, Pardalos, Resende, and Toraldo [2] implemented a Greedy Randomized Adaptive Search Procedure (GRASP) with path relinking to the three-dimensional assignment problem. Their path relinking approach uses two types of moves in which either one or two pairs of indices are interchanged between two cost coefficients from different feasible solution sets. Because the moves are performed within a single permutation vector of the MAP permutation formulation, we will refer to their approach as an intrapermutation path relinking. We also propose a type of interpermutation path relinking where permutations between two good solutions are exchanged. This approach is not as exhaustive as the intrapermutation approach, but it is also significantly less complex while still managing to combine qualities from one or more high-quality solutions.

```
PROCEDURE VariableDepthInterchange($S$)
 1  $S_{Best} \leftarrow S$
 2  $G \leftarrow 0$
 3  $i^* \leftarrow$ RandomSelect$\{1, 2, \ldots, n_0\}$
 4  WHILE $G > 0 \mapsto$
 5      FORALL $m = 0, 1, \ldots, M-1 \mapsto$
 6          $g_m = \min_j \{c_{\phi_0(i^*),\ldots,\phi_{m-1}(i^*),\ \phi_m(j),\ \phi_{m+1}(i^*),\ldots,\phi_{M-1}(i^*)}\}$
 7      $c_{\phi_0(i^*),\ldots,\phi_{m-1}(i^*),\ \phi_m(j^*),\ \phi_{m+1}(i^*),\ldots,\phi_{M-1}(i^*)} \leftarrow g^* = \min_m g_m$
 8      $G \leftarrow G - g^* + c_{\phi_0(i^*),\ldots,\phi_{m-1}(i^*),\ \phi_m(j^*),\ \phi_{m+1}(i^*),\ldots,\phi_{M-1}(i^*)}$
 9      $S \leftarrow S \setminus \{c_{\phi_0(i^*),\phi_1(i^*),\ldots,\ldots,\phi_{M-1}(i^*)}\}$
10      $S \leftarrow S \cup \{c_{\phi_0(i^*),\ldots,\phi_{m-1}(i^*),\ \phi_m(j^*),\ \phi_{m+1}(i^*),\ldots,\phi_{M-1}(i^*)}\}$
11      $S \leftarrow S \setminus \{c_{\phi_0(j^*),\phi_1(j^*),\ldots,\ldots,\phi_{M-1}(j^*)}\}$
12      $S \leftarrow S \cup \{c_{\phi_0(j^*),\ldots,\phi_{m-1}(j^*),\ \phi_m(i^*),\ \phi_{m+1}(j^*),\ldots,\phi_{M-1}(j^*)}\}$
13      $i^* \leftarrow j^*$
14      IF Obj($S$) < Obj($S_{Best}$) $\mapsto$      $S_{Best} \leftarrow S$
15  END
16  RETURN($S_{Best}$)
END VariableDepthInterchange
```

Fig. 19.6 Variable depth interchange

19.3.2.1 Intrapermutation Relinking

The intrapermutation path-relinking approach described by Aiex, Pardalos, Resende, and Toraldo [2] for the three-dimensional assignment problem is generalized for the multidimensional assignment problem in Fig. 19.7. This heuristic systematically transforms an initial solution Φ_I into a guiding solution Φ_G by performing exchanges within the permutation vectors of the initial solution. Each solution is a set of M permutation vectors. The transformation happens one vector at a time starting from $m = 0$. Lines 6 through 13 are performed as long as the m-th vector of the initial solution is different from the corresponding vector of the guiding solution. Index u is the location of an element in Φ_I that is different from Φ_G. Index v is the location of the element that needs to moved to index location u.

The objective function value of Φ_I is evaluated after every exchange. We keep the solution that provides the lowest objective value z. A constraint to the new solution is that it not be the same as the guiding solution. This forces a new solution to be kept even if the guiding solution has a better value. Finally, a local search is performed on the best of the new solutions.

Fig. 19.7 Intrapermutation path relinking

```
PROCEDURE IntraPermPR($\Phi_I, \Phi_G$)
1   LocalSearch($\Phi_I$)
2   $\Phi_{Best} \leftarrow$ MinObj($\Phi_I, \Phi_G$)
3   $z \leftarrow \infty$
4   $\Phi \leftarrow \emptyset$
5   DO $m = 0, \dots, M-1 \mapsto$
6       WHILE $\phi_{I,m} \neq \phi_{G,m} \mapsto$
7           $u \leftarrow \phi_{I,m}(u) \neq \phi_{G,m}(u)$
8           $v \leftarrow \phi_{I,m}(v) = \phi_{G,m}(u)$
9           $\phi_{I,m}(u) \leftrightarrow \phi_{I,m}(v)$
10          IF $z >$ Obj($\Phi_I$) AND $\Phi_I \neq \Phi_G \mapsto$
11              $\Phi = \Phi_I$
12              $z =$ Obj($\Phi$)
13          END
14      END
15  END
16  LocalSearch($\Phi$)
17  IF Obj($\Phi_{Best}$) $>$ Obj($\Phi$) $\mapsto$
18      $\Phi_{Best} = \Phi$
19  RETURN($\Phi_{Best}$)
END PathRelinking
```

19.3.2.2 Interpermutation Relinking

A variation to the intrapermutation path-relinking is the interpermutation approach described in Fig. 19.8. It also makes use of an initial and a guiding solution. Instead of matching the elements of an initial solution permutation vector to the corresponding guiding solution vector one element at a time, the interpermutation path relinking matches the permutation vector in one move. Lines 5–14 exchange permutation vectors between the initial and guiding solutions. The exchanges are only done for $M-1$ vectors since an extra exchange would result in the re-evaluations of the two original solutions. The interpermutation approach also differs from the former approach in that it does not matter which original solution is labeled as a guiding solution. This approach will transform the guiding into the initial solution as well as transform the initial into the guiding solution. After each permutation exchange, we evaluate the current function value of the altered initial and guiding solutions. At the end of the procedure, we perform a local search for the best of the new solutions.

The intrapermutation approach makes a maximum of $\prod_{m=0}^{M-1}(n_m - 1) - 1$ new solutions while the interpermutation approach has a maximum of $2 \cdot (M-1)$. Path relinking the initial and guiding solutions essentially defines a new neighborhood

Fig. 19.8 Interpermutation path relinking

```
PROCEDURE InterPermPR($\Phi_I, \Phi_G$)
1   LocalSearch($\Phi_I$)
2   $\Phi_{Best} \leftarrow$ MinObj($\Phi_I, \Phi_G$)
3   $z \leftarrow \infty$
4   $\Phi \leftarrow \emptyset$
5   DO $m = 0, \ldots, M-2 \mapsto$
6       $\phi_{I,m} \leftrightarrow \phi_{G,m}$
7       IF $z >$ Obj($\Phi_I$) $\mapsto$
8           $\Phi = \Phi_I$
9           $z =$ Obj($\Phi$)
10      END
11      IF $z >$ Obj($\Phi_G$) $\mapsto$
12          $\Phi = \Phi_G$
13          $z =$ Obj($\Phi$)
14      END
15  END
16  LocalSearch($\Phi$)
17  IF Obj($\Phi_{Best}$) $>$ Obj($\Phi$) $\mapsto$
18      $\Phi_{Best} = \Phi$
19  RETURN($\Phi_{Best}$)
END PathRelinking
```

Fig. 19.9 Interpermutation path relinking example, $C \in \Re^{3\times4\times4\times5}$

$$\Phi \leftarrow \text{Initial Solution}$$

$$\Theta \leftarrow \text{Guiding Solution}$$

$$\Psi = \min \left\{ \begin{array}{ll} \{\theta_0, \phi_1, \phi_2, \phi_3\}, & \{\phi_0, \theta_1, \theta_2, \theta_3\} \\ \{\theta_0, \theta_1, \phi_2, \phi_3\}, & \{\phi_0, \phi_1, \theta_2, \theta_3\} \\ \{\theta_0, \theta_1, \theta_2, \phi_3\}, & \{\phi_0, \phi_1, \phi_2, \theta_3\} \end{array} \right\}$$

$$\Psi^* \Leftarrow \text{LOCALSEARCH}(\Psi)$$

$$Z^* \Leftarrow \min\{\Phi, \Theta, \Psi^*\}$$

to be searched. The benefits and drawbacks in selecting path-relinking paths are the same as in selecting a k-exchange neighborhood. Larger neighborhoods may provide a better overall solution, but they do so with increased computational complexity. Intrapermutation path relinking has more solutions in its path and, therefore, has a larger neighborhood to be searched than interpermutation path relinking.

An example of how the interpermutation path relinking for $C \in \Re^{3\times 4\times 4\times 5}$ is given in Fig. 19.9. This type of path relinking allows the initial solution to also serve as a guide to the guiding solution. The permutation vectors from one solution are incrementally replaced by vectors from the other solution. The new feasible solution with the smallest objective value is chosen. After performing a local search, we compare the two starting solutions and the new locally optimal solution and keep the best one.

19.3.3 Variable Neighborhood

Algorithms that make use of local search procedures are typically multistart heuristics. In this type of approach, one of the decisions that needs to be made is how much of the overall computation time should be allocated to the generation of an initial solution and to the exploration of neighborhoods. By selecting faster heuristics for both stages, we can obtain a larger number of local minima. The solutions from this approach tend to also have larger variations. We would be performing searches in regions that are nowhere near a good solution. Good feasible solutions tend to cluster together within a local neighborhood. We would therefore like to have an initial solution that is already good and a local search neighborhood that is large enough to find better solutions.

Because an optimal solution in one neighborhood definition is not usually optimal in other neighborhoods, we propose a variable neighborhood approach in implementing the intra and interpermutation exchange neighborhoods. The variable neighborhood metaheuristic and its applications to different combinatorial optimization problems is described by Hansen and Mladenović [9, 10]. Variable neighborhood works by exploring multiple neighborhoods one at a time. Starting from an initial solution, we define and search the first neighborhood to find a local minimum. From that local minimum, we define and search a different neighborhood to find an even better solution. The metaheuristic continues until all neighborhood definitions have been explored.

Let $\mathcal{N}_2$, $\mathcal{N}_n$, and $\mathcal{N}_p$ represent the intrapermutation 2- and n-exchange and interpermutation 2-exchange neighborhoods, respectively. The initial local search would be bounded by the smallest neighborhood, $\mathcal{N}_2$. If no better solution is found, then the search expands to $\mathcal{N}_n$ and then $\mathcal{N}_p$. If the local search finds a better solution within any neighborhood, the new solution is kept as the best one and the neighborhood is reset to $\mathcal{N}_2$. If no improvements are found in $\mathcal{N}_p$, then the local search is ended. Only time constraints limit the number of neighborhoods that may be used in a variable neighborhood heuristics.

We present the sizes of the different exchange neighborhoods previously defined. The primed neighborhoods are extended versions, which allow outside elements to permute into the solution.

$$\boxed{\begin{aligned}
|\mathcal{N}_2| &= M \times \binom{n_0}{2} \\
|\mathcal{N}_2'| &= M \times \binom{n_0}{2} + M \times (n_m - n_0) \cdot \left[n_0 + \frac{n_0!}{(n_0-2)!} \right] \\
&\quad + M \times (n_m - n_0) \cdot (n_m - n_0 - 1) \cdot \binom{n_0}{2} \\
|\mathcal{N}_n| &= M \times n_0! \\
|\mathcal{N}_n'| &= M \times \frac{n_m!}{(n_m - n_0)!} \\
|\mathcal{N}_p| &= \binom{M}{2}
\end{aligned}}$$

The neighborhoods in the variable neighborhood approach are typically applied in order of nondecreasing size. This makes sense since we are looking for a solution that is optimal for multiple neighborhoods. If we are exploring the intersection of different neighborhoods, we could save computation time by searching the smallest neighborhoods first. The local minima of the larger neighborhoods are less likely to be located within the intersection. A variation would be to search the most efficient neighborhoods first. Exploring an efficient neighborhood provides a large improvement from the initial solution within a small amount of time. Small neighborhoods are not necessarily efficient if they are computationally difficult to implement. This variation gives priority to neighborhoods that are easiest to explore.

19.4 Discussion

The paper defines local search neighborhoods that may be implemented in solving the multidimensional assignment. It also provides techniques for extending the neighborhoods to provide algorithms with capabilities of escaping local minima. These techniques may be implemented to improve heuristic solutions or to obtain tighter bounds for implicit enumeration algorithms.

For the local neighborhoods, two types of k-exchange are defined based on the permutation formulation of the MAP. Intrapermutation exchanges are performed between elements of the same vector, while interpermutation exchanges moves elements from one vector to another. The intrapermutation exchanges are more efficient than the interpermutation 2-exchanges, while the interpermutation n-exchange is the least efficient. The n-exchange search, where n is the dimension size, is solvable in $\mathcal{O}(n^3)$ time through the use of a two-dimensional assignment problem solver. Expanded k-exchanges for MAP instances with nonhomogenous dimension sizes are introduced in the paper as a method for expanding the standard neighborhoods.

The standard exchanges are only between elements that are already in the solution, while expanded exchanges allow outside elements to come into the solution. The expanded versions can find better solutions due to its larger neighborhood but with more computation effort.

Escape from local minima is possible through the use of local neighborhood extensions such as variable depth interchange, path relinking, and variable neighborhood search. Path relinking may be implemented with either interpermutation and intrapermutation linking. Variable neighborhood search is done using different combinations of the intrapermutation 2- and n-exchanges and interpermutation 2-exchange neighborhoods.

The neighborhoods and their extensions may be implemented to find better solutions of heuristics or to tighten lower bounds in exact algorithms. Careful consideration must be given to the selection of local neighborhood definitions and to the techniques for escaping local minima. The choice of local search has the potential to be more important than the selection of heuristic or exact algorithm to solve the multidimensional assignment problem.

References

1. Ahuja, R.K., Ergun, Ö., Orlin, J.B., Punnen, A.P.: A survey of very large-scale neighborhood search techniques. Discrete Appl. Math. **123**(1–3), 75–102 (2002)
2. Aiex, R.M., Pardalos, P.M., Resende, M.G.C., Toraldo, G.: GRASP with path relinking for the three-index assignment problem. INFORMS J. Comput. **17**(2), 224–247 (2005)
3. Balas, E., Saltzman, M.J.: An algorithm for the three-index assignment problem. Oper. Res. **39**(1), 150–161 (1991)
4. Blackman, S.S.: Multiple-Target Tracking with Radar Applications. Artech House, Norwood (1986)
5. Burkard, R.E., Çela, E.: Quadratic and three-dimensional assignment problems: An annotated bibliography. In: DellAmico, M., Maffioli, F., Martello, S. (eds.) Annotated Bibliographies in Combinatorial Optimization, pp. 373–392. Wiley, New York (1997). Chap. 21
6. Burkard, R.E., Çela, E.: Linear assignment problems and extensions. In: Du, D., Pardalos, P.M. (eds.) Handbook of Combinatorial Optimization, pp. 75–149. Kluwer Academic, Dordrecht (1999). Chap. 21
7. Çela, E.: Assignment problems. In: Pardalos, P.M., Resende, M.G.C. (eds.) Handbook of Applied Optimization, pp. 661–678. Oxford University Press, New York (2002). Chap. 17.9
8. Gilbert, K., Hofstra, R.: Multidimensional assignment models. Decis. Sci. **19**, 306–321 (1988)
9. Hansen, P., Mladenović, N.: Variable neighborhood search: Principles and applications. Eur. J. Oper. Res. **130**, 449–467 (2001)
10. Hansen, P., Mladenović, N.: Variable neighborhood search. In: Pardalos, P.M., Resende, M.G.C. (eds.) Handbook of Applied Optimization, pp. 221–234. Oxford University Press, New York (2002). Chap. 3.6.9
11. Laguna, M., Martí, R.: GRASP and path relinking for 2-layer straight line crossing minimization. INFORMS J. Comput. **11**, 44–52 (1999)
12. Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the traveling salesman problem. Oper. Res. **21**, 498–516 (1973)
13. Lourenço, H.R., Martin, O.C., Stützle, T.: Iterated local search. In: Glover, F., Kochenberger, G. (eds.) Handbook of Metaheuristics. International Series in Operations Research and Management Science, vol. 57, pp. 321–353. Kluwer Academic, Dordrecht (2002)

14. Pardalos, P.M., Pitsoulis, L.S. (eds.): Nonlinear Assignment Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7. Kluwer Academic, Dordrecht (2000)
15. Spieksma, F.C.R.: Multi index assignment problems: Complexity, approximation, applications. In: Pardalos, P.M., Pitsoulis, L.S. (eds.) Nonlinear Assignment Problems: Algorithms and Applications. Combinatorial Optimization, vol. 7, pp. 1–12. Kluwer Academic, Dordrecht (2000). Chap. 1
16. Yagiura, M., Ibaraki, T.: Local search. In: Pardalos, P.M., Resende, M.G.C. (eds.) Handbook of Applied Optimization, pp. 104–123. Oxford University Press, New York (2002). Chap. 3.5