Lecture Notes in Computer Science 5175

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

James Bailey David Maier
Klaus-Dieter Schewe Bernhard Thalheim
Xiaoyang Sean Wang (Eds.)

Web Information Systems Engineering – WISE 2008

9th International Conference
Auckland, New Zealand, September 1-3, 2008
Proceedings

Springer

Volume Editors

James Bailey
The University of Melbourne, Dept. of Computer Science and Software Engineering
Melbourne, VIC 3010, Australia
E-mail: baileyj@unimelb.edu.au

David Maier
Portland State University, Department of Computer Science
PO Box 751, Portland, OR 97207, USA
E-mail: maier@cs.pdx.edu

Klaus-Dieter Schewe
Information Science Research Centre
20A Manapouri Cr., Palmerston North, New Zealand
E-mail: kdschewe@acm.org

Bernhard Thalheim
Christian Albrechts University Kiel, Department of Computer Science
Hermann-Rodewald-Str. 3, 24118Kiel, Germany
E-mail: thalheim@is.informatik.uni-kiel.de

Xiaoyang Sean Wang
University of Vermont, Department of Computer Science
33 Colchester Avenue, Votey Bldg. 351B, Burlington, VT 05405, USA
E-mail: xywang@cs.uvm.edu

Library of Congress Control Number: Applied for

CR Subject Classification (1998): H.4, H.2, H.3, H.5, K.4.4, C.2.4

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-540-85480-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-85480-7 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12464579 06/3180 5 4 3 2 1 0

Preface

WISE 2008 was held in Auckland, New Zealand, during September 1–3, at The Auckland University of Technology City Campus Conference Centre. The aim of this conference was to provide an international forum for researchers, professionals, and industrial practitioners to share their knowledge in the rapidly growing area of Web technologies, methodologies, and applications. Previous WISE conferences were held in Hong Kong, China (2000), Kyoto, Japan (2001), Singapore (2002), Rome, Italy (2003), Brisbane, Australia (2004), New York, USA (2005), Wuhan, China (2006) and Nancy, France (2007).

The call for papers created considerable interest. Around 110 paper submissions were received and the international Program Committee selected 31 papers out of the 110 submissions (an acceptance rate of 28.2%). Of these, 17 papers were chosen for standard presentation and the remaining 14 papers for short presentation. The authors of the accepted papers range across 13 countries: Australia, China, Germany, Greece, Italy, Japan, Korea, New Zealand, Poland, Singapore, Spain, Switzerland and the USA. The technical track of the WISE 2008 program offered nine paper presentation sessions. The selected papers covered a wide and important variety of issues in Web information systems engineering such as querying; search; ranking; trust; peer-to-peer networks; information filtering; information integration; agents and mining. A few selected papers from WISE 2008 will be published in a special issue of the *World Wide Web Journal*, by Springer. In addition, a $1000 prize was awarded to the authors of the paper selected for the "Yahiko Kambayashi Best Paper Award." We thank all authors who submitted their papers and the Program Committee members and external reviewers for their excellent work. Finally, WISE 2008 included two keynote talks, one panel, two tutorials, and three workshops.

We would also like to acknowledge the two General Conference Co-chairs, Klaus-Dieter Schewe and Xiaoyang Sean Wang, and the two Local Organization Co-chairs, Dave Parry and Russel Pears. We also thank Sven Hartmann and Xiaofang Zhou as Workshop and Tutorial Co-chairs, Markus Kirchberg as Publicity Chair, and Gustavo Rossi as Panel Chair. We are grateful to the the WISE Society for their assistance with the conference and also to Markus Kirchberg for his work in editing the proceedings. We hope that the present proceedings contain many ideas that will help push the boundaries of the Web for tomorrow's society.

September 2008

James Bailey
David Maier
Bernhard Thalheim

Conference Organization

General Conference Co-chairs

Schewe, Klaus-Dieter	Information Science Research Centre, New Zealand
Wang, Xiaoyang Sean	University of Vermont, Burlington, USA

Program Committee Co-chairs

Bailey, James	The University of Melbourne, Australia
Maier, David	Portland State University, Portland, USA
Thalheim, Bernhard	Christian Albrechts University, Kiel, Germany

Workshop and Tutorial Co-chairs

Hartmann, Sven	Clausthal University of Technology, Germany
Zhou, Xiaofang	The University of Queensland, Brisbane, Australia

Local Organization Co-chairs

Parry, Dave	Auckland University of Technology, New Zealand
Pears, Russel	Auckland University of Technology, New Zealand

Panel Chair

Rossi, Gustavo	Universidad Nacional de La Plata, Argentina

Industrial Program Co-chairs

Hosking, John	University of Auckland, New Zealand
Pastor, Oscar	Universidad Politécnica de Valencia, Spain

Steering Committee Liaison

Zhang, Yanchun	Victoria University, Australia

Publicity Chair

Kirchberg, Markus	Institute for Infocomm Research, A*STAR, Singapore

Program Committee

Abramowicz, Witold	The Poznan University of Economics, Poznan, Poland
Altenhofen, Michael	SAP Research, Karlsruhe, Germany
Balke, Wolf-Tilo	University of Augsburg, Germany
Baresi, Luciano	Politecnico di Milano, Italy
Beeri, Catriel	The Hebrew University of Jerusalem, Israel
Benatallah, Boualem	University of New South Wales, Sydney, Australia
Bertossi, Leopoldo	Carleton University, Ottawa, Canada
Bidoit, Nicole	University of Paris-South, France
Bouguettaya, Athman	CSIRO ICT Centre, Canberra, Australia
Bry, Francois	Ludwig Maximilian University, Munich, Germany
Busse, Susanne	Berlin University of Technology, Germany
Casati, Fabio	University of Trento, Italy
Crescenzi, Valter	Roma Tre University, Rome, Italy
Cuzzocrea, Alfredo	University of Calabria, Rende, Italy
Damiani, Ernesto	University of Milan, Italy
De Antonellis, Valeria	Brescia University, Italy
De Troyer, Olga	Vrije Universiteit Brussel, Belgium
Delcambre, Lois	Portland State University, USA
Dobbie, Gill	University of Auckland, New Zealand
Dolog, Peter	Aalborg University, Denmark
Eckstein, Silke	Braunschweig University of Technology, Germany
Ehrich, Hans-Dieter	Braunschweig University of Technology, Germany
Embley, David	Brigham Young University, Provo, USA
Frasincar, Flavius	Erasmus University Rotterdam, The Netherlands
Garzotto, Franca	University of Milan, Italy
Geerts, Floris	University of Edinburgh, UK
Georgakopoulos	Dimitrios, Telcordia Technologies, Morristown, USA
Ghose, Aditya	University of Wollongong, Australia
Godart, Claude	University Henri Poincare, Nancy, France
Goh, Angela	Nanyang Technological University, Singapore
Gopalkrishnan, Vivekanand	Nanyang Technological University, Singapore
Gottlob, Georg	University of Oxford, UK
Gruhn, Volker	University of Leipzig, Germany
Guizzardi, Giancarlo	Federal University of Espírito Santo, Brazil
Han, Hyoil	Drexel University, Philadelphia, USA
Hartmann, Sven	Clausthal University of Technology, Germany
Hosking, John	University of Auckland, New Zealand
Houben, Geert-Jan	Vrije Universiteit Brussel, Belgium
Kanza, Yaron	Technion – Israel Institute of Technology, Haifa, Israel
Kappel, Gerti	Technical University of Vienna, Austria

Karlapalem, Kamalakar	University of Hyderabad, India
Kirchberg, Markus	Institute for Infocomm Research, A*STAR, Singapore
Klettke, Meike	University of Rostock, Germany
Koch, Nora	Ludwig Maximilian University, Munich, Germany
Lepape, Cecile	Pierre & Marie Curie University, Paris, France
Li, Qing	City University of Hong Kong, China
Liddle, Stephen	Newcastle University, UK
Lim, Ee-Peng	Singapore Management University, Singapore
Lin, Xuemin	University of New South Wales, Sydney, Australia
Ling, Tok Wang	National University of Singapore, Singapore
Link, Sebastian	Victoria University of Wellington, New Zealand
Liu, Chengfei	Swinburne University of Technology, Melbourne, Australia
Liu, Qing	CSIRO Tasmania ICT Centre, Hobart, Australia
Liu, Ying	Hong Kong Polytechnic University, China
Lu, Jianguo	University of Windsor, Canada
Ma, Hui	Victoria University of Wellington, New Zealand
Martens, Wim	Technical University of Dortmund, Germany
Medjahed, Brahim	The University of Michigan, Dearborn, USA
Meghini, Carlo	CNR ISTI Pisa, Italy
Mendes, Emilia	University of Auckland, New Zealand
Merialdo, Paolo	Roma Tre University, Rome, Italy
Nejdl, Wolfgang	Leibniz University Hannover, Germany
Ng, Wilfred	Hong Kong University of Science and Technology, China
Noack, Rene	Christian Albrechts University, Kiel, Germany
Norrie, Moira	ETH Zurich, Switzerland
Oliveira, José Palazzo M. de	Federal University of Rio Grande do Sul, Brazil
Pautasso, Cesare	University of Lugano, Switzerland
Peng, Zhiyong	Wuhan University, China
Pokorny, Jaroslav	Charles University, Prague, Czech Republic
Poulovassilis, Alexandra	University of London, UK
Purvis, Martin	University of Otago, Dunedin, New Zealand
Risch, Tore	Uppsala University, Sweden
Roantree, Mark	Dublin City University, Ireland
Röhm, Uwe	University of Sydney, Australia
Rolland, Colette	University of Paris1 – Pantheon-Sorbonne, France
Rosemann, Michael	Queensland University of Technology, Brisbane, Australia
Rossi, Gustavo	Universidad Nacional de La Plata, Argentina
Sadiq, Shazia	University of Queensland, Brisbane, Australia
Sato, Shin-ya	NTT Network Innovation Laboratories, Tokyo, Japan
Schneider, Michel	Blaise Pascal University, Aubiere, France

Schwabe, Daniel	Pontifical Catholic University of Rio de Janeiro, Brazil
Shen, Heng Tao	University of Queensland, Brisbane, Australia
Shui, Bill	National ICT Australia, Sydney, Australia
Spyratos, Nicolas	University of Paris-South, France
Srinivasa, Srinath	Indian Institute of Information Technology, Bangalore, India
Stumme, Gerd	University of Kassel, Germany
Sun, Aixin	Nanyang Technological University, Singapore
Tanaka, Katsumi	Kyoto University, Japan
Tanaka, Yuzuru	Hokkaido University, Japan
Taniar, David	Monash University, Melbourne, Australia
Theodoratos, Dimitri	New Jersey Institute of Technology, USA
Torlone, Riccardo	Roma Tre University, Rome, Italy
Vansummeren, Stijn	Hasselt University, Belgium
Wagner, Gerd	Cottbus University of Technology, Germany
Wang, Junhu	Griffith University, Brisbane, Australia
Wang, X. Sean	University of Vermont, Burlington, USA
Weber, Gerald	University of Auckland, New Zealand
Wong, Raymond	University of New South Wales, Sydney, Australia
Yu Xu, Jeffrey	Chinese University of Hong Kong, China
Zhang, Yanchun	Victoria University, Melbourne, Australia
Zhou, Xiaofang	University of Queensland, Brisbane, Australia

External Referees

Zhifeng Bao
Devis Bianchini
Henrik Björklund
Dario Colazzo
Hanbo Dai
Alan Fekete
Dominik Flejter
Sergio Flesca
Tim Furche
Lucantonio Ghionna
Adrian Giurca
Hai Huang
Hakim Hacid
Yanan Hao
Dat Hoang
Adrian Iacovelli
Adam Jatowt
Winly Jurnawan
George Koliadis
Woralak Kongdenfha
Andreas Kupfer
Cane Wing-ki Leung
Jianxin Li
Jiang Li
Philipp Liegl
Xumin Liu
Christoph Lofi
Giusy Di Lorenzo
Jiangang Ma
Zaki Malik
Brigitte Mathiak
Michele Melchiori
Ali Mesbah
Heiko Mueller
Satoshi Nakamura
Oana Nicolae
Hiroaki Ohshima
Satoshi Oyama

Ardian Poernomo
Andrea Pugliese
Thomas Richter
Michel de Rougemont
Seung Ryu
Deneckere Rébecca
Klaus Schulz
Martina Seidl
Carine Souveyet
Puay Siew Tan
Yannis Tzitzikas
Marlis Valentini
Fang Wei
Christoph Wieser
Manuel Wimmer
Vincent Wolff-Marting
Huayu Wu
Xiaoying Wu
Chuan Xiao
Liang Xu
Qi Yu
Xiaohui Zhao
George Zheng
Ying Zhou

Workshops

First International Workshop on Web Information Systems Engineering for Electronic Businesses and Governments (E-BAG 2008)
Link, Sebastian, Victoria University of Wellington, New Zealand
Ma, Hui, Victoria University of Wellington, New Zealand
Yang, Jian, Macquarie University, Sydney, Australia

Second International Workshop on Web Usability and Accessibility (IWWUA 2008)
Abrahão, Silvia, Valencia University of Technology, Spain
Cachero, Cristina, University of Alicante, Spain
Matera, Maristella, Politecnico di Milano, Italy

First International Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM&LCW 2008)
Kowalkiewicz, Marek, SAP Research Brisbane, Australia
Flejter, Dominik, Poznan University of Economics, Poland
Kaczmarek, Tomasz, Poznan University of Economics, Poland

Tutorials

Data Quality for the Global Enterprise
Sadiq, Shazia, The University of Queensland, Brisbane, Australia
Deng, Ke, The University of Queensland, Brisbane, Australia

Requirements Specification and Acceptance Testing of Collaborative Work Environments, Groupware, and CSCW Systems
Hausen, Hans-Ludwig, Fraunhofer, St. Augustin, Germany

Panel

Panel: Engineering Issues for the Web 2.0
Zhang, Yanchun, Victoria University, Melbourne, Australia (moderator)
Daniel, Florian University of Trento, Italy
Melia, Santiago University of Alicante, Spain
Tanaka, Katsumi University of Kyoto, Japan
Bouguettaya, Athman, CSIRO ICT Centre, Canberra, Australia
Nicklas, Daniela, Carl von Ossietzky Universität Oldenburg, Germany

Organized By

WISE Society
c/o Department of Computer Science
City University of Hong Kong
Tat Chee Avenue, Kowloon
Hong Kong, China

`http://www.wisesociety.org/`

Auckland University of Technology
New Zealand

`http://www.aut.ac.nz/`

Information Science Research Centre
20a Manapouri Crescent
Palmerston North 4410
New Zealand

`http://isrc.mucoms.org/`

Supported By

Institute for Infocomm Research (I^2R)
Agency for Science, Technology and Research
21 Heng Mui Keng Terrace
Singapore 119613
Singapore

http://www.i2r.a-star.edu.sg/

Christian Albrechts University, Kiel
Germany

http://www.uni-kiel.de/

Table of Contents

Keynotes

Grid Computing and Peer-to-Peer Systems

Web Mining

Rich Web User Interfaces

Semantic Web

Web Information Retrieval

Web Data Integration

Queries and Peer-to-Peer Systems

Web Services

Miscellaneous

Data Quality in Web Information Systems

Xiaofang Zhou, Shazia Sadiq, and Ke Deng

School of Information Technology and Electrical Engineering
The University of Queensland, Australia
{zxf,shazia,dengke}@itee.uq.edu.au

Abstract. The World Wide Web has brought a wave of revolutionary changes for people and organizations to generate, disseminate and use data. With unprecedented access to massive amount of data and powerful information gathering capabilities enabled by Web-based technologies, the traditional closed world assumption for database systems has been challenged. More and more data from the Web are used today as essential information sources, directly or indirectly, for all types of decision making purposes in not only just personal, but also many business and scientific applications. A user of such Web data, however, has to constantly rely on their own judgement on data quality, such as correctness, currency, consistency and completeness. This is an unreliable and often very difficult process, as the quality of this judgement itself often relies on the quality of other information obtained from the Web, and the relationship among the data used can be very complex and sometime hidden from the user.

While the issue of data quality is as old as data itself, it is now exposed at a much higher, broader and more critical level due to the scale, diversity and ubiquitousness of Web Information Systems. The intrinsic mismatch between the intended use and actual use of the data on the Web is a fundamental cause of poor data quality for Web-based applications. In this talk, we will introduce the notion of data quality, from its root in management information systems research to new issues and challenges in the context of large-scale Web Information Systems. After a brief introduction to organizational and architectural solutions to the data quality problem, this talk will focus on the current research activities and results on computational solutions form the database community in data profiling, record linking, conditional functional constraints, data provenance and data uncertainty. These technical solutions will be examined for their promises and limitations to the problem of data quality in Web Information Systems. Finally, we will discuss a list of open research problems.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, p. 1, 2008.

XML Storage and Processing on Mobile Devices

Raymond K. Wong

NICTA and University of New South Wales, NSW 2052, Australia
raymond.wong@nicta.com.au

Abstract. As XML database sizes grow, the amount of space used for storing the data and auxiliary data structures becomes a major factor in query and update performance. This is especially critical for devices with limited resources such as handheld computers, mobile phones or sensor networks. This presentation describes the requirements for efficiently storing and processing XML data on mobile devices. In particular, it summarizes our previous work on a compact XML storage scheme that supports all XPath navigational operations in near constant time. In addition to supporting efficient queries, the space requirement of the proposed scheme is within a constant factor of the information theoretic minimum, while insertions and deletions can be performed in near constant time as well. As a result, the proposed structure features a small memory footprint that increases cache locality, whilst still supporting standard APIs, such as DOM, and necessary database operations, such as queries and updates, efficiently. Finally, some applications using the proposed storage scheme are presented.

1 Introduction

The popularity of XML as a data representation language has produced a wealth of research on efficiently storing and querying tree structured data. As the amount of XML data available increases, it is becoming vital to be able to not only query and maintain this information quickly, but also store it in a compact manner. Therefore, it would be extremely useful to have a compact and yet efficient storage scheme for XML, i.e., a space-efficient representation of the data structure which also maintains low access and update costs for all of the desired primitive operations for data processing. The flexibility of XML makes finding a scheme which satisfies all these requirements at the same time extremely challenging.

When looking for a compact storage scheme for XML, there are several issues that need to be addressed. For example, it has to support fast operations, especially we are considering software applications that target people on the move. Moreover, if intensive compression methods are employed, they need to be optional and can be switched on or off due to low computation power of some mobile devices. In summary, the major issues include:

– *It must support fast navigational operations*: Many XML applications, such as collaborative document editing systems, depend upon efficient tree traversal, using a standard interface such as DOM. Halverson et al [8] demonstrated that a *combination* of navigational and structural join operators is most effective for evaluating

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 2–5, 2008.

queries. Hence, it is imperative that the storage scheme supports fast traversal of the XML tree, in all possible directions, preferably in constant time or near constant time. Previous work, such as that of Zhang et al [17], has addressed the issue of succinctly representing XML, but at the cost of linear time navigational operations, which is not acceptable for many practical applications.

- *It must support efficient insertions and deletions*: Several papers address the space issue by storing XML in compressed form [2, 11, 13, 15]. They also support path expression queries or fast navigational access but do not allow efficient update operations such as node insertion. This can be a critical concern in many database applications.
- *It must support efficient join operations*: Current query optimization techniques for XML such as work of Halverson et al [8], make heavy use of the structural join [1] or their variations, which relies on a constant time operator to determine the ancestor-descendant relationship between two nodes. Thus, any general XML storage scheme should also support such an operator in near constant time.
- *It should separate the topology, schema and text of the document*: All XML query languages select and filter results based on some combination of the topology, schema and text data of the document. To allow efficient scans over these parts of the document, it is natural to find a representation that partitions them into separate physical locations.
- *It should permit extra indexes*: Many applications may require addition specialized indexes to be built upon their data. Therefore, a general purpose database system is required to provide a storage representation, such that it is flexible enough to accommodate such need. More specifically, the storage scheme used by the database system must provide a simple, efficient and stable way of referencing its stored data items.

2 Brief Survey

To my best knowledge, Liefke and Suciu [11] proposed the first compressed XML scheme called XMill. Although XMill achieves a good compression ratio, its major drawback (which is the lack of support for query and update) hinders its broad application in database systems. Various approaches were proposed after XMill and they share similar benefits and drawbacks, e.g., XMLPPM [5].

Compared to XMill, XGrind [15] has a lower compression ratio but supports certain types of queries. XPRESS [13] uses reverse arithmetic encoding to encode tags using start/end regions. Both XGrind and XPRESS require top-down query evaluation, and do not support set-based query evaluation such as structural joins.

Buneman et al [2] separate the tree structure and its data. They then use bi-simulation to compress the documents that share the same sub-tree, however, they can only support node navigations in linear time. With a similar idea but different technique, Maneth et al [3, 12] also compress XML by calculating the minimal sharing graph equivalent to the minimal regular tree grammar. In order to provide tree navigations, a DOM proxy that maintains runtime traversal information is needed [3]. Since only the compression efficiency was reported in the paper, both query and navigation performance of their proposed scheme are unclear.

Most XML storage schemes, such as [7, 8, 9, 10], make use of interval and preorder/postorder labeling schemes to support constant time order lookup, but fail to address the issue of maintenance of these labels during updates. Recently, Silberstein et al [14] proposed a data structure to handle ordered XML which guarantees both update and lookup costs. Similarly, the L-Tree labeling scheme proposed by Chen et al [4] addressed the same problem and has the same time and space complexity as [14], however, they do not support persistent identifiers.

The succinct approach proposed by Zhang et al [17] targeted secondary storage, and used a balanced parentheses encoding for each block of data. Unfortunately, their summary and partition schemes support rank and select operations in linear time only. Their approach uses the Dewey encoding for node identifiers in their indexes. The drawbacks of the Dewey encoding are significant: updates to the labels require linear time, and the size of the labels is also linear to the size of the database in the worst case. Thus, the storage of the topology can require quadratic space in the worst case.

Finally, there are several proposals published recently, e.g. [6, 7]. [7] show that all XPath axes can be handled using a preorder/postorder labeling. Instead of maintaining these two labels (i.e., two integers), the scheme proposed by [16] requires less than 3 bits per node to process all XPath axes, which is an attractive alternative for applications that are both space and performance conscious.

Ferragina et. al.[6] first shred the XML tree into a table of two columns, then sort and compress the columns individually. It does not offer immediate capability of navigating or searching XML data unless an extra index is built. However, the extra index will degrade the overall storage size (i.e., the compression ratio). Furthermore, the times for disk access and decompression of local regional blocks have been omitted from their experiments. For most of work presented above, data updates have been disregarded.

References

1. Al-Khalifa, S., Jagadish, H.V., Koudas, N., Patel, J.M.: Structural joins: A primitive for efficient XML query pattern matching. In: Proceedings of the 18th International Conference on Data Engineering (ICDE), pp. 141–153. IEEE Computer Society, Los Alamitos (2002)
2. Buneman, P., Grohe, M., Koch, C.: Path Queries on Compressed XML. In: Proceedings of the 29th International Conference on Very Large Databases (VLDB), pp. 141–152. Morgan Kaufmann, San Francisco (2003)
3. Busatto, G., Lohrey, M., Maneth, S.: Efficient memory representation of XML documents. In: Bierman, G., Koch, C. (eds.) DBPL 2005. LNCS, vol. 3774, pp. 199–216. Springer, Heidelberg (2005)
4. Chen, Y., Mihaila, G.A., Bordawekar, R., Padmanabhan, S.: L-tree: A dynamic labeling structure for ordered XML data. In: Lindner, W., Mesiti, M., Türker, C., Tzitzikas, Y., Vakali, A.I. (eds.) EDBT 2004. LNCS, vol. 3268, pp. 209–218. Springer, Heidelberg (2004)
5. Cheney, J.: XMLPPM: XML-Conscious PPM Compression, `http://www.cs.cornell.edu/People/jcheney/xmlppm/xmlppm.html`
6. Ferragina, P., Luccio, F., Manzini, G., Muthukrishnan, S.: Compressing and searching XML data via two zips. In: WWW, pp. 751–760 (2006)
7. Grust, T., van Keulen, M., Teubner, J.: Accelerating xpath evaluation in any RDBMS. ACM Trans. Database Syst. 29, 91–131 (2004)

8. Halverson, A., Burger, J., Galanis, L., Kini, A., Krishnamurthy, R., Rao, A.N., Tian, F., Viglas, S., Wang, Y., Naughton, J.F., DeWitt, D.J.: Mixed Mode XML Query Processing. In: Proceedings of the 29th International Conference on Very Large Databases (VLDB), pp. 225–236. Morgan Kaufmann, San Francisco (2003)
9. Jagadish, H.V., Al-Khalifa, S., Chapman, A., Lakshmanan, L.V.S., Nierman, A., Paparizos, S., Patel, J.M., Srivastava, D., Wiwatwattana, N., Wu, Y., Yu, C.: TIMBER: A native XML database. VLDB Journal 11(4), 274–291 (2002)
10. Li, Q., Moon, B.: Indexing and querying XML data for regular path expressions. In: Proceedings of the 27th International Conference on Very Large Databases (VLDB), pp. 361–370. Morgan Kaufmann, San Francisco (2001)
11. Liefke, H., Suciu, D.: XMill: an efficient compressor for XML data. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, pp. 153–164. ACM Press, New York (2000)
12. Maneth, S., Busatto, G.: Tree transducers and tree compressions. In: Walukiewicz, I. (ed.) FOSSACS 2004. LNCS, vol. 2987, pp. 363–377. Springer, Heidelberg (2004)
13. Min, J.-K., Park, M.-J., Chung, C.-W.: XPRESS: A Queriable Compression for XML Data. In: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp. 122–133. ACM Press, New York (2003)
14. Silberstein, A., He, H., Yi, K., Yang, J.: BOXes: Efficient maintenance of order-based labeling for dynamic XML data. In: 21st International Conference on Data Engineering (ICDE), pp. 285–296 (2005)
15. Tolani, P.M., Haritsa, J.R.: XGRIND: A query-friendly XML compressor. In: Proceedings of the 18th International Conference on Data Engineering (ICDE), pp. 225–234. IEEE Computer Society, Los Alamitos (2002)
16. Wong, R., Lam, F., Shui, W.: Querying and maintaining a compact XML storage. In: WWW, pp. 1073–1082 (2007)
17. Zhang, N., Kacholia, V., Ozsu, M.T.: A Succinct Physical Storage Scheme for Efficient Evaluation of Path Queries in XML. In: Proceedings of the 20th International Conference on Data Engineering (ICDE), pp. 54–65. IEEE Computer Society, Los Alamitos (2004)

Approximate Information Filtering in Peer-to-Peer Networks

Christian Zimmer[1], Christos Tryfonopoulos[1], Klaus Berberich[1], Manolis Koubarakis[2], and Gerhard Weikum[1]

[1] Max-Planck-Institute for Informatics, Saarbrcken, Germany
[2] National and Kapodistrian University of Athens, Greece
{czimmer,trifon,kberberi,weikum}@mpi-inf.mpg.de, koubarak@di.uoa.gr

Abstract. Most approaches to information filtering taken so far have the underlying hypothesis of potentially delivering notifications from every information producer to subscribers. This exact publish/subscribe model creates an efficiency and scalability bottleneck, and might not even be desirable in certain applications. The work presented here puts forward MAPS, a novel approach to support approximate information filtering in a peer-to-peer environment. In MAPS a user subscribes to and monitors only carefully selected data sources, and receives notifications about interesting events from these sources only. This way scalability is enhanced by trading recall for lower message traffic. We define the protocols of a peer-to-peer architecture especially designed for approximate information filtering, and introduce new node selection strategies based on time series analysis techniques to improve data source selection. Our experimental evaluation shows that MAPS is scalable; it achieves high recall by monitoring only few data sources.[1]

1 Introduction

Much information of interest to humans is available today on the Web, making it extremely difficult to stay informed without sifting through enormous amounts of information. In such a dynamic setting, *information filtering (IF)*, also referred to as *publish/subscribe*, *continuous querying*, or *information push*, is equally important to one-time querying, since users are able to subscribe to information sources and be notified when documents of interest are published. This need for *content-based* push technologies is also stressed by the deployment of new tools such as Google Alert or the QSR system [1]. In an IF scenario, a user posts a *subscription* (or *continuous query*) to the system to receive *notifications* whenever certain events of interest take place (e.g., when a paper on distributed systems becomes available).

In this paper we put forward MAPS (*Minerva Approximate Publish/Subscribe*), a novel architecture to support content-based approximate information filtering in peer-to-peer (P2P) environments. While most information filtering approaches taken so far have the underlying hypothesis of potentially delivering notifications from every information producer, MAPS relaxes this assumption by monitoring only selected sources

[1] This work has been partly supported by the EU project AEOLUS and EVERGROW.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 6–19, 2008.

that are likely to publish documents relevant to the user's interests in the future. In MAPS, a user subscribes with a continuous query and monitors only the most interesting sources in the network. Only published documents from these sources are forwarded to him. The system is responsible for managing the user query, discovering new potential sources and moving queries to better or more promising sources. Since in an IF scenario the data is originally highly distributed residing on millions of sites (e.g., with people contributing to blogs), a P2P approach seems an ideal candidate for such a setting. However, exact pub/sub functionality has proven expensive for such distributed environments [2,3,4]. MAPS offers a natural solution to this problem, by avoiding document granularity dissemination as it is the main scalability bottleneck of other approaches.

As possible application scenarios for MAPS consider the case of news filtering (but with the emphasis on information quality rather than timeliness of delivery) or blog filtering where users subscribe to new posts. Not only do these settings pose scalability challenges, but they would also incur an information avalanche and thus cognitive overload to the subscribed users, if the users were alerted for each and every new document published at any source whenever this matched a submitted continuous query. Our approximate IF approach ranks sources, and delivers matches only from the best ones, by utilizing novel publisher selection strategies. Despite that the presented approach focuses on a P2P setting based on *Distributed Hash Tables* (DHT) [21,19], notice that our architecture can also be realized in other settings, like a single server monitoring a number of distributed sources, or a farm of servers in a data center providing an alerting service. In the light of the above, the contributions presented in this paper are threefold:

- We define a network-agnostic P2P architecture and its related protocols for supporting *approximate* IF functionality in a P2P environment. To the best of our knowledge this is the first approach that looks into the problem of approximate IF in such a setting.
- We show that traditional resource selection strategies are not sufficient in this setting, and devise a *novel* method to predict publishing behavior based on time series analysis of IR metrics. This technique allows us to improve recall, while monitoring only a small number of publishers.
- We present an extensive experimental study of our prediction mechanism in terms of recall, and evaluate our protocols in terms of message load in the network.

In previous work, we have compared exact and approximate information filtering in [5], applied approximate IR and IF to the digital library domain [6], and investigated different time series analysis methods [7]. The current paper extends the core ideas behind approximate IF by elaborating on the protocols, discussing in detail our prediction mechanisms, and presenting an extensive experimental evaluation of our approach.

The rest of the paper is organized as follows. Related work is discussed in Section 2. Section 3 presents the MAPS architecture and discusses the related protocols, while Section 4 introduces our node selection method. Experimental results are presented in Section 5, and Section 6 concludes this paper.

2 Related Work

Database research on continuous queries has its origins in [8] and systems like OpenCQ [9] and NiagaraCQ [10]. These papers offered centralized solutions to the problem of continuous query processing. More recently, continuous queries have been studied in depth in the context of monitoring and stream processing with various centralized [11,12] and distributed proposals [13,14,15]. The efforts to improve network efficiency and reduce delivery delays in content-based pub/sub systems lead to approaches like HYPER [16], where a hybrid architecture that exploits properties of subject-based pub/sub approaches is presented. Hermes [17] was one of the first proposals to use a Distributed Hash Table (DHT) for building a topic-based pub/sub system, while PeerCQ [13] utilized a DHT to build a content-based system for processing continuous queries. Finally, Meghdoot [18] utilized the CAN DHT [19] to support an attribute-value data model and offered new ideas for the processing of subscriptions with range predicates and load balancing.

Recently, several systems that employed an IR-based query language to support information filtering on top of structured overlay networks have been deployed. DHTrie [3] extended the Chord protocol [21] to achieve exact information filtering functionality and applied document-granularity dissemination to achieve the recall of a centralized system. In the same spirit, LibraRing [20] presented a framework to provide information retrieval and filtering services in two-tier digital library environments. Similarly, pFilter [2] used a hierarchical extension of the CAN DHT [19] to store user queries and relied on multi-cast trees to notify subscribers. In [4], the authors show how to implement a DHT-agnostic solution to support prefix and suffix operations over string attributes in a pub/sub environment.

Query placement, as implemented in exact information filtering approaches such as [2,3], is deterministic, and depends upon the terms contained in the query and the hash function provided by the DHT. These query placement protocols lead to filtering effectiveness of a centralized system. Compared to a centralized approach, [2,3] exhibit scalability, fault-tolerance, and load balancing at the expense of high message traffic at publication time. In MAPS, only the most promising nodes store a user query and are thus monitored. Publications are only matched against its local query database, since, for scalability reasons, no publication forwarding is used. Thus, in the case of approximate filtering, the recall achieved is lower than that of exact filtering, but document-granularity dissemination to the network is avoided.

3 The MAPS Protocols

3.1 The Directory Protocol

MAPS utilizes a conceptually global, but physically distributed directory, which can be layered on top of a Chord-style DHT [21], and manages aggregated information about each node's local knowledge in compact form, similarly to [22]. The DHT partitions the term space, such that every node is responsible for the statistics of a randomized subset of terms within the directory. To keep IR statistics up-to-date, each node distributes per-term summaries of its *local index* along with contact information to the directory. For

efficiency reasons, these messages can be piggy-backed to DHT maintenance messages with applying batching strategies.

To facilitate message sending, we will use the function `send(`msg, I`)` to send message msg to the node responsible for identifier I. This is similar to the Chord function `lookup(`I`)` [21], and costs $O(\log n)$ overlay hops for a network of n nodes. In MAPS, every publisher node uses POST messages to distribute per-term statistics.

Let us now examine how a publisher node P updates the global directory when $T = \{t_1, t_2, \ldots, t_k\}$ denotes the set of all terms contained in document publications of P occurring after the last update. For each term $t_i \in T$, P computes the maximum frequency of occurrence of term t_i within the documents contained in P's collection ($tf_{t_i}^{max}$), the number of documents in the document collection of P that t_i is contained in (df_{t_i}), and the size of the document collection cs. Having collected the statistics for term t_i, P creates message POST$(id(P), ip(P), tf_{t_i}^{max}, df_{t_i}, cs, t_i)$, where $id(P)$ is the identifier of node P and $ip(P)$ is the IP address of P. P then uses function `send()` to forward the message to the node D responsible for identifier $H(t_i)$ (i.e., the node responsible for maintaining statistics for term t_i by using the Chord hash function H mapping terms to identifiers). Once a node D receives a POST message, it stores the statistics for P in its local post database to keep them available on request for any node. Finally, notice that our architecture allows the usage of any type of P2P network (structured or unstructured), given that the necessary information (i.e., the per-node IR statistics) is made available through appropriate protocols.

3.2 The Subscription Protocol

The subscription protocol is responsible for selecting the publishers that will index a query. The node selection procedure utilizes the directory to discover and retrieve node statistics that will guide query indexing. Then, a ranking of the potential sources is performed and the query is sent to the *top-k* ranked publishers. The publishers storing the query are the only ones that will be monitored for new publications.

Let us assume that a subscriber node S wants to subscribe with a *multi-term query* q of the form $t_1 t_2 \ldots t_k$ with k distinct terms. To do so, S needs to determine which nodes in the network are promising candidates to satisfy the continuous query with appropriate documents published in the future. This source ranking can be decided once appropriate statistics about data sources are collected from the directory, and a ranking of these sources is calculated based on the node selection strategy described in Section 4.

To collect statistics about the data sources, S needs to contact all directory nodes responsible for the query terms. Thus, for each query term t_i, S computes $H(t_i)$, which is the identifier of the node responsible for storing statistics about other nodes that publish documents containing the term t_i. Subsequently, S creates message COLLECTSTATS$(id(S), ip(S), t_i)$, and uses the function `send()` to forward the message in $O(\log n)$ hops to the node responsible for identifier $H(t_i)$. Notice that the message contains $ip(S)$, so its recipient can directly contact S.

When a node D receives a COLLECTSTATS message asking for the statistics of term t_i, it searches its local post store to retrieve the node list L_i of all posts of the term. Subsequently, a message RETSTATS(L_i, t_i) is created by D and sent to S using its IP found in the COLLECTSTATS message. This collection of statistics is shown in step 1

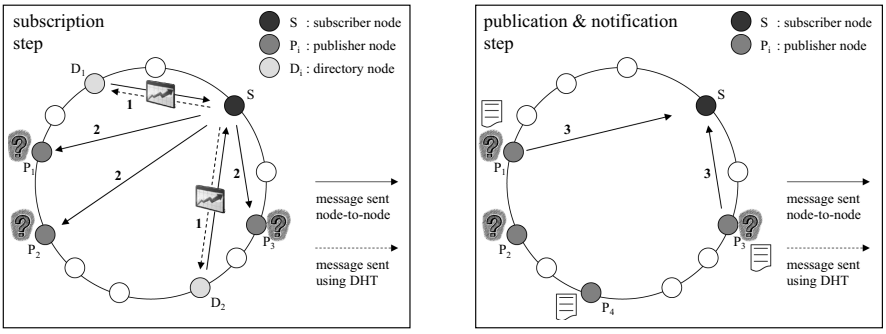


Fig. 1. Illustration of the MAPS protocols

of Figure 1, where S contacts directory nodes D_1 and D_2. Once S has collected all the node lists L_i for the terms contained in q, it utilizes an appropriate scoring function $score(N, q)$ to compute a node score with respect to q, for each one of the nodes N contained in L_i. Based on the score calculated for each node, a ranking of nodes is determined and the highest ranked nodes are candidates for storing q.

Once the nodes that will store q have been determined, S constructs message INDEXQ$(id(S), ip(S), q)$ and uses the IP addresses associated with the node to forward the message to the nodes that will store q. When a publisher P receives a message INDEXQ containing q, it stores q using a local query indexing mechanism such as [23,24]. This procedure is shown in step 2 of Figure 1, where S contacts publishers P_1, P_2 and P_3.

Nodes publishing documents relevant to q, but not indexing q, will not produce any notification, simply because they are not aware of q. Since only selected nodes are monitored for publications, the node ranking function becomes a critical component, which will determine the final recall achieved. This scoring function can be based on standard resource selection approaches from the IR literature (e.g. CORI [25]). However, as we show in Section 4.2, these approaches alone are not sufficient in an IF setting, since they where designed for retrieval scenarios, in contrast to the IF scenario considered here, and are aimed at identifying specialized authorities. Filtering and node selection are dynamic processes, therefore periodic query repositioning, based on user-set preferences, is necessary to adapt to changes in publisher's behavior. To reposition an already indexed query q, a subscriber re-executes the subscription protocol.

3.3 Publication and Notification Protocol

When a document d is published by P, it is matched against P's local query database to determine which subscribers should be notified. Then, for each subscriber S, P constructs a notification message NOTIFY$(id(P), ip(P), d)$ and sends it to S using the IP address associated with the stored query (shown in step 3 of Figure 1). Notice that nodes publishing documents relevant to a query q, but not storing it, will produce no notification (e.g., node P_4 in Figure 1).

4 Node Selection Strategy

To select which publishers should be monitored, the subscription protocol of Section 3.2 uses a scoring function to rank nodes. In our approach the subscriber computes a node score based on a combination of *resource selection* and *node behavior prediction* formulas as shown below:

$$score(P, q) = \alpha \cdot sel(P, q) + (1 - \alpha) \cdot pred(P, q) \quad (1)$$

In Equation 1, q is a query, P is a publisher node, and $sel(P, q)$ and $pred(P, q)$ are scoring functions based on resource selection and prediction methods respectively that assign a score to a node P with respect to a query q. Here, $score(P, q)$ is the scoring function that decides the final score. The tunable parameter α affects the balance between authorities (high $sel(P, q)$ scores) and nodes with potential to publish matching documents in the future (high $pred(P, q)$ scores). Based on these scores, a ranking of nodes is determined and q is forwarded to the highest ranked nodes. Notice that our node selection strategy is general and can also be used in centralized settings, where a server (instead of the distributed directory of Section 3.1) maintains the necessary statistics, and mediates the interaction between publishers and subscribers.

To show why an approach that relies only on resource selection is not sufficient, and give the intuition behind node behavior prediction, consider the following example. Assume a node P_1 that has specialized and become an authority in sports, but publishes no relevant documents any more. Another node P_2 is not specialized in sports, but is currently crawling a sports portal. Imagine a user who wants to stay informed about the upcoming 2008 Olympic Games, and subscribes with the continuous query *2008 Olympic Games*. If the ranking function solely relied on resource selection, node P_1 would always be chosen to index the user's query, which would be wrong given that node P_1 no longer publishes sports-related documents. On the other hand, to be assigned a high score by the ranking function, node P_2 would have to specialize in sports – a long procedure that is inapplicable in a IF setting which is by definition dynamic. The fact that resource selection alone is not sufficient is even more evident in the case of news items. News items have a short shelf-life, making them the worst candidate for slow-paced resource selection algorithms. The above example shows the need to make slow-paced selection algorithms more sensitive to the publication dynamics in the network. We employ node behavior prediction to cope with these dynamics. The main contribution of our work with respect to predicting node behavior, is to view the IR statistics as time series and use statistical analysis tools to model node behavior. Time series analysis accounts for the fact that the observations have some sort of internal structure (e.g., trend, seasonality etc.), and uses this fact to analyze older values and predict future ones.

4.1 Time Series Analysis

To predict node behavior, we consider time series of IR statistics, thus making a rich repository of techniques from time series analysis [26] applicable. These techniques

predict future time series values based on past observations and differ in (i) their assumptions about the internal structure (e.g., whether trends and seasonality can be observed) and (ii) their flexibility to put emphasis on more recent observations. Since the considered IR statistics exhibit trends, for instance, when nodes successively crawl sites that belong to different topics, or, gradually change their thematic focus, the employed time series prediction technique must be able to deal with trends. Further, in our scenario we would like to put emphasis on a node's recent behavior and thus assign higher weight to recent observations when making predictions about its future behavior. We choose *double exponential smoothing* (DES) as a prediction technique, since it can both deal with trends and put emphasis on more recent observations. Its explanation is included in detail in [7] in combination with optimized method to apply DES in the MAPS setting.

For completeness we mention that there is also triple exponential smoothing that, in addition, handles seasonality in the observed data. For an application with many long-lasting queries, one could use triple-exponential smoothing, so that seasonality is taken into account.

4.2 Node Behavior Prediction

The function $pred(P, q)$ returns a score for a node P that represents the likelihood of publishing documents relevant to query q in the future. Using the DES technique, two values are predicted. First, for all terms t in query q, we predict the value for $df_{P,t}$ (denoted as $\hat{df}_{P,t}$), and use the difference (denoted as $\delta(\hat{df}_{P,t})$) between the predicted and the last value obtained from the directory to calculate the score for P (function δ signifies difference). Value $\delta(\hat{df}_{P,t})$ reflects the number of relevant documents that P will publish in the next time-unit. Second, we predict $\delta(\hat{cs})$ as the difference in the collection size of node P reflecting the node's overall expected future publishing activity. We thus model two aspects of the node's behavior: (i) its potential to publish relevant documents in the future, and (ii) its overall expected future publishing activity. The time series of IR statistics that are needed as an input to our prediction mechanism are obtained using the distributed directory. The predicted behavior for node P is quantified as follows:

$$pred(P, q) = \sum_{t \in q} \log \left(\delta(\hat{df}_{P,t}) + \log \left(\delta(\hat{cs}_P) + 1 \right) + 1 \right) \tag{2}$$

In the above Equation 2, the publishing of relevant documents is more accented than the dampened publishing rate. If a node publishes no documents at all, or, to be exact, $\delta(\hat{cs})$ and $\delta(\hat{df})$ are 0, then the $pred(P, q)$ value is also 0. The addition of 1 in the $\log$ formulas yields positive predictions and avoids $\log 0$.

4.3 Resource Selection

The function $sel(P, q)$ returns a score for a node P and a query q, and is calculated using standard resource selection algorithms from the IR literature (see [27] for an overview), such as tf-idf based methods, CORI [25], or language models. Using $sel(P, q)$, we identify authorities specialized in a topic, which, as argued above, is not sufficient for

our IF setting. In our implementation we use an CORI-like approach well-known from P2P information retrieval [22].

5 Experimental Evaluation

5.1 Experimental Setup

To conduct each experiment described in the next sections the following steps are executed. Initially the network is set up and the underlying DHT is created. Subsequently, subscribers utilize the protocol described in Section 3.2 to subscribe to selected publishers. We will say that a publisher node is *monitored* with query q by a subscriber when it stores q in its local query database. Once queries are stored, the documents are published to the network and at certain intervals (called *rounds*) queries are repositioned. A repositioning round occurs every 30 document publications on average per peer. At the end of each round, message costs and recall for this round are calculated, and subscribers rank publishers using Equation 1 to reposition their queries accordingly. We consider the following system parameters:

- ρ**:** The percentage of top-ranked publishers. In experiments, ρ is the same for all subscribers and different system properties for a value of ρ up to 25% are investigated. It is clear that when $\rho = 100\%$ (i.e., all publishers are monitored) then recall is 1, and our approach degenerates to exact filtering.
- α**:** To control the influence of resource selection vs. node behavior prediction in our experiments, we vary the value of α in the node selection formula. A value of α close to 0 emphasizes node behavior prediction, while values close to 1 stress resource selection.

To investigate the effectiveness and efficiency of our approach, we model node publishing behavior through different publishing scenarios described in Section 5.3. Retrieval effectiveness of our approach is utilized by recall, while efficiency is measured using a benefit/cost ratio metric. Both are defined as follows:

- **Recall:** We measure recall by computing the *ratio* of the total number of notifications received by subscribers to the total number of published documents matching subscriptions. In experiments we consider the *average recall* computed over *all* rounds (i.e., for the complete experiment).
- **Benefit/Cost Ratio:** To evaluate the efficiency of our approach, we measure the total number of *subscription* and *notification* messages to calculate the benefit/cost ratio as the number of notifications per message sent. Notice that in our approach no publication messages are needed, since publications trigger only local node computations and are not disseminated as in exact matching approaches.

As explained in Section 3, the number of subscription messages depends on the number of query terms and monitored publishers. In addition, the subscription costs are proportional to the number of query repositionings, since for each repositioning the subscription protocol is re-executed. Finally, for each publication matching an indexed

query, a notification message is created and sent to the subscriber. In the experiments of Sections 5.3 and 5.4, the message cost needed to maintain the distributed directory information is not taken into account since our main goal is to focus on the filtering protocol costs. Typically, directory messages are included in DHT maintenance messages, thus they can be considered as part of the underlying routing infrastructure.

5.2 Experimental Data

The data collection contains over 2 million documents from a focused Web crawl categorized in one of ten categories: *Music*, *Finance*, *Arts*, *Sports*, *Natural Science*, *Health*, *Movies*, *Travel*, *Politics*, and *Nature*. The overall number of corpus documents is $2,052,712$. The smallest category consists of $67,374$ documents, the largest category of $325,377$ documents. The number of distinct terms after stemming amounts to $593,876$.

In all experiments, the network consists of $1,000$ nodes containing 300 documents each in their initial local collection. Each peer hosts 15% random documents, 10% not categorized documents, and 75% documents from a single category, resulting in 100 nodes specializing in each category. Using the document collection, we construct 30 continuous queries containing two, three or four query terms. Each of the query terms selected is a strong representative of a document category (i.e., a frequent term in documents of one category and infrequent in documents of the other categories). Example queries are *music instrument*, *museum modern art*, or *space model research study*.

5.3 Different Publishing Scenarios

To measure MAPS's efficiency in terms of recall and message cost under various settings, we consider four scenarios representing different publishing behaviors. The overall number of published documents is constant in all scenarios ($300K$ documents) therefore the maximum number of notifications concerning the 30 active queries is also constant ($146,319$ notifications), allowing us to compare across different scenarios. The following figures show experimental results with average recall and benefit/cost ratio for different publishing scenarios and different α and ρ values. A baseline approach (called *rand*) that implements a random node selection method is included for comparison purposes.

Consistent Publishing. The first publishing scenario targets the performance of our approach when nodes' interests remain unchanged over time. Figure 2 shows that the average recall and the benefit/cost ratio do not depend on the ranking method used, and our approach presents the same performance for all values of α. This can be explained as follows. Publishers that are consistently publishing documents from one category have built up an expertise in this category and node selection techniques are able to detect this and monitor the authorities for each topic. Similarly, publication prediction observes this trend for consistent behavior and chooses to monitor the most specialized nodes. Compared to the baseline approach of random selection, our approach achieves up to 7 times a higher average recall (for $\rho = 10\%$). Finally, the best value for the benefit/cost ratio is when $\rho = 10\%$.

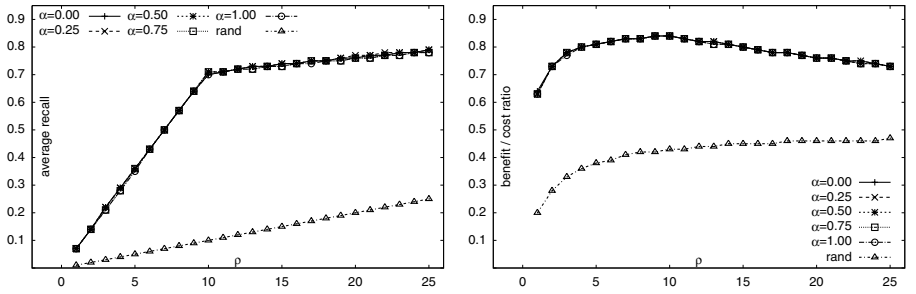


Fig. 2. Average recall and benefit/cost ratio for *Consist* scenario

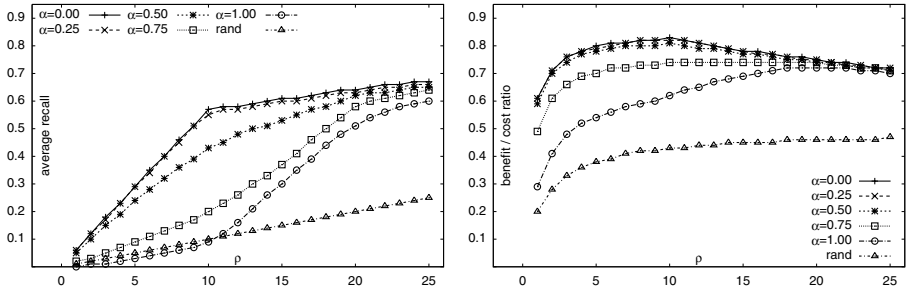


Fig. 3. Average recall and benefit/cost ratio for *CatChg* scenario

Category Change. Since users may publish documents from different topics, we use this scenario to simulate the changes in a publisher's content. In the *CatChg* scenario, a node initially publishes documents from one category, and switches to a different category at some point in time. Figures 3 illustrates the performance of our approach in this scenario for different values for α and ρ. The most important observation from this figure is the performance of the prediction method in comparison to resource selection. In some cases (e.g., when $\rho = 10\%$) not only publication prediction achieves more that 6 times better average recall than resource selection, but also resource selection is only marginally better than *rand* (e.g., when monitoring 15% of publishers). In general, both average recall and benefit cost/ratio improve as α reaches 0 and prediction is stressed. This abrupt changes in the publishers' content cannot be captured by the resource selection method, which favors topic authorities. On the other hand, publication prediction detects the publishers' topic change from changes in the IR statistics and adapts the scoring function to monitor nodes publishing documents relevant to subscribed queries.

Publishing Breaks. The *Break* scenario models the behavior of nodes as they log in and out of the network. We assume that some publisher is active and publishes documents for some rounds, and then logs out of the network, publishing no documents any more. This procedure is continued in intervals, modeling, e.g., a user using the publication service at home, and switching it off every day in the office. Our ranking mechanism

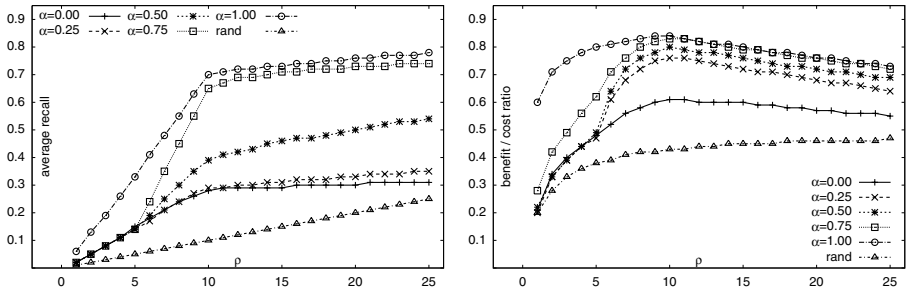


Fig. 4. Average recall and benefit/cost ratio for *Break* scenario

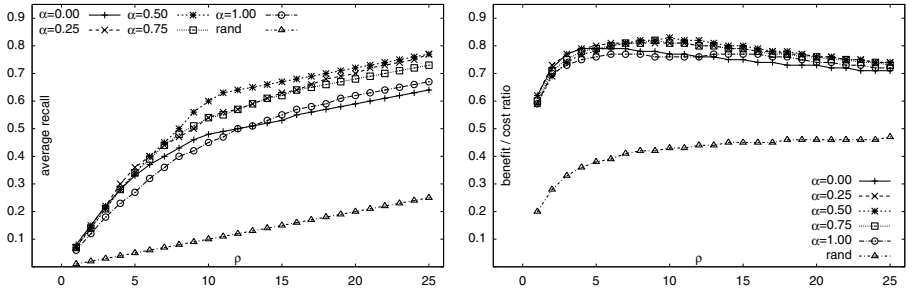


Fig. 5. Average recall and benefit/cost ratio for *TmpChg* scenario

should adapt to these inactivity periods, and distinguish between nodes not publishing documents any more and nodes making temporary pauses.

Figure 4 demonstrates that both average recall and benefit/cost ratio improve when resource selection is emphasized (i.e., when α is close to 1), since pauses in the publishing mislead the prediction formula to foresee that, in the future, no relevant publications will occur. For this reason, nodes with inactivity periods are ranked lower resulting in miss of relevant documents. On the other hand, resource selection accommodates less dynamics, so temporary breaks remain undetected and the topic authorities continue to be monitored since the ranking procedure is not affected. Consequently, selecting a ranking method that favors prediction leads to poor recall and low benefit/cost ratio, that are comparable to those of *rand*.

Temporary Changes. The last scenario we investigated (*TmpChg*), targets temporary changes in a node's published content. This scenario models users utilizing the service e.g., for both their work and hobbies, or users that temporarily change their publishing topic due to an unexpected or highly interesting event (earthquake, world cup finals, etc.). Here, a publisher makes available documents about one topic for a number of rounds, and then temporarily publishes documents about a different topic. In the next rounds, the publisher reverts between publishing documents out of these categories, to stress the behavior of nodes being interested in a topic but occasionally publish documents covering other topics.

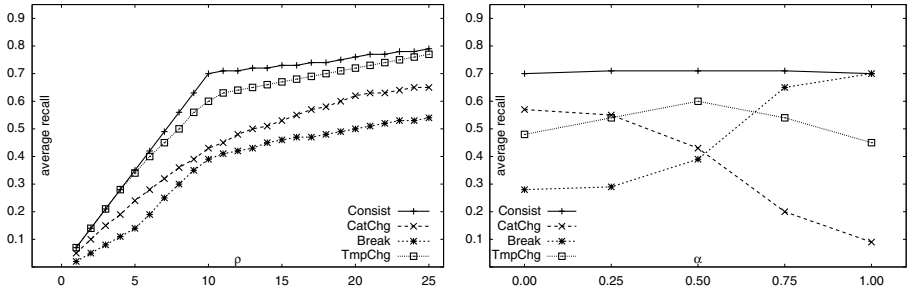


Fig. 6. Average recall across scenarios for $\alpha = 0.5$ and $\rho = 10\%$

In this scenario, MAPS presents the highest average recall values when equally utilizing resource selection and prediction methods ($\alpha = 0.5$), as suggested by Figure 5. This happens because *TmpChg* can be considered as a scenario lying between an abrupt category change (*CatChg*) and publishing documents about a specific topic with small breaks (*Break*). Thus, the combination of publication prediction and resource selection used by subscribers, aids in identifying these publication patterns in publisher's behaviors and thus selecting the nodes publishing more relevant documents. Finally, an interesting observation emerging from this figure is that almost all combinations of ranking methods perform similarly both in terms of average recall and benefit/cost ratio. This is due to the effectiveness of the ranking methods, that cause the dampening of the subscription messages by the high number of notification messages created. Compared to our baseline random node selection, all methods show an increase of as much as 600% for average recall and 200% for benefit/cost ratio.

5.4 Comparison across Scenarios

In this section, we change our experimental viewpoint, select some baseline values for α and ρ, and compare the average recall and the benefit/cost *across scenarios*.

Average Recall Analysis. Figure 6 illustrates the average recall values achieved for the various publishing scenarios. When $\alpha = 0.5$ and ρ value increases up to 25% of monitored publishers (leftmost figure), we see that *Consist* achieves the highest average recall, since, as explained in Section 5.3, it is not affected by the choice of α. The rest of the scenarios achieve lower average recall with the *TmpChg* scenario being the most promising. For the rest of the scenarios the choice of $\alpha = 0.5$ is a compromise that leads to a satisfactory average recall level. When ρ is set to 10% and the emphasis of the ranking method moves from publication prediction ($\alpha = 0$) to resource selection ($\alpha = 1$), average recall remains relatively unaffected for *Consist* publication scenarios (second figure). In contrast, *CatChg* and *Break* are influenced by the ranking method, and demonstrate a significant change in their behavior and average recall achieved. The *TmpChg* scenario reaches the highest average recall levels when both publication prediction and resource selection are equally weighted with $\alpha = 0.5$.

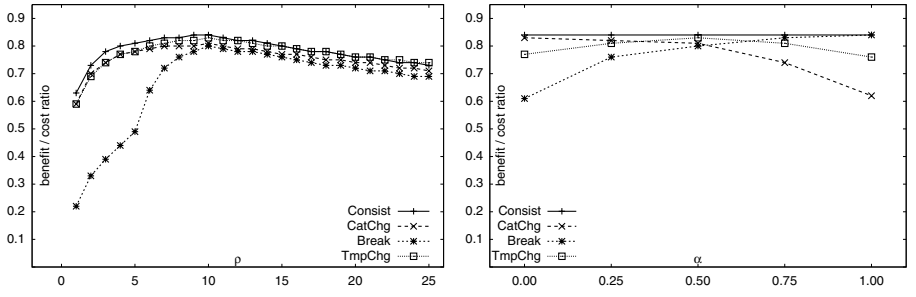


Fig. 7. Benefit / cost ratio across scenarios for $\alpha = 0.5$ and $\rho = 10\%$

Message Costs Analysis. The benefit/cost ratio for the different publishing scenarios is shown in Figure 7. Here, the value of ρ increases to demonstrate the benefit/cost ratio for a constant $\alpha = 0.5$ (third figure) and the dependency of the benefit/cost ratio parameter on the ranking method is illustrated as a function of α for a constant ρ of 10% (rightmost figure).

The most important observation is that independently of the ranking method used, in all scenarios, the highest value for the benefit/cost ratio is achieved when monitoring 10% of the publisher nodes. At this value, our approach needs around 1.2 messages per notification generated (since the number of notifications/message is around 0.8 as shown in the graphs). Obviously, the best possible benefit/cost ratio is 1, since at least one message (the notification message) is needed at publication time to inform a subscriber about a matching document. This means that we generate an average of 0.2 extra subscription messages per notification sent.

We observe that in the *Consist* scenario, a change in the ranking method has no effect on the value of the benefit/cost ratio. Nevertheless, the *Break* and *CatChg* scenarios perform differently such that the benefit/cost ratio increases for the case of resource selection and publication prediction respectively. The *TmpChg* scenario differs from all other scenarios because, for the same reason as in Section 5.4, the highest benefit/cost ratio is achieved when combining both resource selection as node prediction scores.

6 Conclusions

We have presented the MAPS system architecture, discussed the associated protocols, and introduced a novel node selection technique based on time series analysis to support approximate IF in a P2P setting. Our experimental evaluation showed the efficiency and effectiveness of our approach in many diverse scenarios.

References

1. Yang, B., Jeh, G.: Retroactive Answering of Search Queries. In: WWW (2006)
2. Tang, C., Xu, Z.: pFilter: Global Information Filtering and Dissemination Using Structured Overlay Networks. In: FTDCS (2003)

3. Tryfonopoulos, C., Idreos, S., Koubarakis, M.: Publish/Subscribe Functionality in IR Environments using Structured Overlay Networks. In: SIGIR (2005)
4. Aekaterinidis, I., Triantafillou, P.: PastryStrings: A Comprehensive Content-Based Publish/Subscribe DHT Network. In: ICDCS (2006)
5. Tryfonopoulos, C., Zimmer, C., Weikum, G., Koubarakis, M.: Architectural Alternatives for Information Filtering in Structured Overlays. Internet Computing (2007)
6. Zimmer, C., Tryfonopoulos, C., Weikum, G.: MinervaDL: An Architecture for Information Retrieval and Filtering in Distributed Digital Libraries. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 148–160. Springer, Heidelberg (2007)
7. Zimmer, C., Tryfonopoulos, C., Berberich, K., Weikum, G., Koubarakis, M.: Node Behavior Prediction for LargeScale Approximate Information Filtering. In: LSDS-IR (2007)
8. Terry, D., Goldberg, D., Nichols, D., Oki, B.: Continuous Queries over Append-Only Databases. In: SIGMOD (1992)
9. Liu, L., Pu, C., Tang, W.: Continual Queries for Internet Scale Event-Driven Information Delivery. In: TKDE 2000 (2000)
10. Chen, J., DeWitt, D.J., Tian, F., Wang, Y.: NiagaraCQ: A Scalable Continuous Query System for Internet Databases. In: SIGMOD (2000)
11. Madden, S., Shah, M.A., Hellerstein, J.M., Raman, V.: Continuously Adaptive Continuous Queries over Streams. In: SIGMOD 2002 (2002)
12. Chandrasekaran, S., Franklin, M.J.: PSoup: A System for Streaming Queries over Streaming Data. VLDB Journal (2003)
13. Gedik, B., Liu, L.: PeerCQ: A Decentralized and Self-Configuring Peer-to-Peer Information Monitoring System. In: ICDCS (2003)
14. Ahmad, Y., Çetintemel, U.: Networked Query Processing for Distributed Stream-Based Applications. In: VLDB (2004)
15. Jain, A., Hellerstein, J.M., Ratnasamy, S., Wetherall, D.: A Wakeup Call for Internet Monitoring Systems: The Case for Distributed Triggers. HotNets (2004)
16. Zhang, R., Hu, Y.C.: HYPER: A Hybrid Approach to Efficient Content-Based Publish/Subscribe. In: ICDCS (2005)
17. Pietzuch, P.R., Bacon, J.: Hermes: A Distributed Event-Based Middleware Architecture. In: DEBS (2002)
18. Gupta, A., Sahin, O.D., Agrawal, D., Abbadi, A.E.: Meghdoot: Content-Based Publish/Subscribe over P2P Networks. In: Jacobsen, H.-A. (ed.) Middleware 2004. LNCS, vol. 3231, pp. 254–273. Springer, Heidelberg (2004)
19. Ratnasamy, S., Francis, P., Handley, M., Karp, R.M., Shenker, S.: A Scalable Content-Addressable Network. In: SIGCOMM (2001)
20. Tryfonopoulos, C., Idreos, S., Koubarakis, M.: LibraRing: An Architecture for Distributed Digital Libraries Based on DHTs. In: Rauber, A., Christodoulakis, S., Tjoa, A.M. (eds.) ECDL 2005. LNCS, vol. 3652, pp. 25–36. Springer, Heidelberg (2005)
21. Stoica, I., Morris, R., Karger, D.R., Kaashoek, M.F., Balakrishnan, H.: Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In: SIGCOMM (2001)
22. Bender, M., Michel, S., Triantafillou, P., Weikum, G., Zimmer, C.: Improving Collection Selection with Overlap Awareness in P2P Search Engines. In: SIGIR (2005)
23. Tryfonopoulos, C., Koubarakis, M., Drougas, Y.: Filtering Algorithms for Information Retrieval Models with Named Attributes and Proximity Operators. In: SIGIR (2004)
24. Yan, T.W., Garcia-Molina, H.: The SIFT Information Dissemination System. In: TODS (1999)
25. Callan, J.: Distributed Information Retrieval. Kluwer Academic Publishers, Dordrecht (2000)
26. Chatfield, C.: The Analysis of Time Series - An Introduction. CRC Press, Boca Raton (2004)
27. Nottelmann, H., Fuhr, N.: Evaluating Different Methods of Estimating Retrieval Quality for Resource Selection. In: SIGIR (2003)

Achieving Effective Multi-term Queries for Fast DHT Information Retrieval*

Quanqing Xu[1], Heng Tao Shen[2], Yafei Dai[1], Bin Cui[1], and Xiaofang Zhou[2]

[1] Department of Computer Science and Technology, Peking University, 100871 Beijing, China
{xqq,dyf}@net.pku.edu.cn, bin.cui@pku.edu.cn
[2] School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, QLD, Australia
{shenht,zxf}@itee.uq.edu.au

Abstract. Distributed Hash Tables (DHTs) are well-suited for exact match lookups using unique identifiers, but do not directly support multi-term queries. Related research of query expansion has shown that adding new terms to a query via ad hoc feedback improves the retrieval effectiveness of such query. In the paper, we propose an effective multi-term query processing algorithm for information retrieval in DHT systems. Given the significance of first term in a multi-term query, the query is sent to the peers containing the first term. To enhance the query effectiveness, we design two query expansion mechanisms and an implicit relevance feedback approach based on users' behaviors. Additionally, we record the query log and the expansion terms for each query which can accelerate the future queries and improve the query accuracy. Experimental results show that our query methods yield substantial improvements in retrieval effectiveness in the following three aspects: recall, precision at 10 standard recall levels and precision histograms.

1 Introduction

Recently, Peer-to-Peer (P2P) information retrieval (IR) has gained tremendous interest from the research community since it is a foundation upon which P2P-based large-scale digital library systems [1, 2] and web search engines (ODISSEA [3], MinervaLight [4], etc) rest. More and more considerations have been concerned with the question whether P2P techniques can and should also be adopted for ordinary IR. As a primary design goal, most DHT-based P2P systems have achieved efficient key lookup, typically at $O(\log N)$ complexity. However, the inherent exact matching in DHT lookup circumscribes its functionality. For instance, multi-term query, which is defined to find the entire query terms over the underlying P2P network, is difficult to achieve via DHT lookup directly. To answer multi-term queries, DHT keyword search systems must transmit inverted lists over the network to perform a join. Several techniques [3, 5, 6, 7] have been proposed to reduce this cost.

* This research has been supported by the National Natural Science foundation of China under Grant No.60673183, National Grand Fundamental Research 973 program of China under Grant No.2004CB318204 and Australian research grant (ARC DP0773483).

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 20–35, 2008.

Prior research [8] shows that 85% of the queries posted at web search engines have 3 or less query terms, which usually can not adequately express information required for retrieving the relevant documents. Web search engines widely uses automatic query expansion mechanisms to solve this problem. Naturally, P2P networks, especially based on DHT, should adopt automatic query expansion mechanisms to facilitate information retrieval since the DHT infrastructure is a good one for P2P information retrieval [3, 5, 4]. On the other hand, since the average length of Gnutella queries is 3.54 terms and 83% queries have no more than 5 terms [9], while the average length of the queries in TREC-3 (query #151-200) set is 19.08 terms per query on average [10] for traditional text retrieval task, multi-term queries have to be supported for information retrieval in DHT networks. For instance, query terms are sent to many nodes because of the inherent exact matching in DHT lookup.In the proposed approaches (Mercury [11], PHT [12], GChord [13]), multiple DHTs are utilized, one per term, where each DHT is responsible for the value space of the associated term. Thus, for a multi-term query, a join operation is necessary to arrive at the final response for the query. They are not scalable nor efficient solutions.

In this paper, we propose an effective query method based on query expansion mechanisms, users' behaviors and search history to gracefully support multi-term queries in the DHT networks. We firstly design two query expansion mechanisms and an implicit user relevance feedback approach based on users' behaviors. The first term of a multi-term query q is the most significant for information retrieval, which is maintained by a peer p. The query is routed via p to all the peers containing the first term. For each query, its log is recorded in peer p for accelerating the same query in the future. For the queries whose multi-term is the subset of q, its expansion terms are employed to improve the query accuracy. The main contributions of this paper are fourfold:

- For a multi-term query, query terms are interrelated and even compose a sentence. But the first term is primary, which is maintained by a peer p in DHT networks. The query is guided by p. Since the query terms generally have high probability to be in a single sentence, we propose two simple and affective query expansion mechanisms from terms of the same sentence of returned documents for the DHT IR systems.
- To the best of our knowledge, it is the first time that an implicit relevance feedback approach based on users' browse and download behaviors was presented in DHT networks. It can change documents rank position to obtain better expansion terms for a query according to users' implicit feedback, since a peer maintained the first term can collect all the documents about query terms and return the query results.
- We design a novel multi-term query approach based on query expansion, users' browse and download behaviors and search history for fast DHT information retrieval.
- An extensive performance study has been conducted to verify the effectiveness and efficiency of our method.

The remainder of this paper is organized as follows: Section 2 formulates the problem and reviews the related work. We present two query expansion mechanisms in Section 3. Section 4 describes implicit relevance feedback based on users' browse and

download behaviors. Section 5 presents an effective query method for fast DHT information retrieval. Section 6 reports the experimental results. Section 7 concludes this paper.

2 Preliminaries

2.1 Problem Formulation

Because DHT offers performance guarantees in self-healing and self-organizing capabilities, network scalability and routing efficiency, even in the high network churn rates due to frequent peer joining and leaving, DHT is employed as the underlying system architecture for P2P information retrieval. Let a system have totally N peers in the DHT networks, each peer maintains a randomized set of terms which includes one or more terms since DHT can automatically partition the term space. Each peer is guaranteed within $O(\log N)$ complexity in the routing of the DHT network. For a multi-term query, intersection of index lists is absolutely indispensable, but the intersection is often large, while search engines or digital libraries based on P2P usually present only the most highly ranked documents. So, a large quantity of precious bandwidth is wasted. To reduce the bandwidth consumption [5, 6], Bloom Filter (BF) is employed as a probabilistic hash based scheme that represents a set of keys with minimal cost. In essence, a multi-term query method for DHT IR is a partition by keyword method [6], which reads as shown in Figure 1.

Procedure 1. QueryPartitionByKeyword(P_{ori}, q)
Input: P_{ori} is the query originator
q is the query text of P_{ori}
Output: the valid pairs of peers and documents
1. $R = \emptyset$;
2. Send a query text q to the DHT networks;
// P_i is a node that stores the i^{th} term of q
3. Route q to P_1 and get the index list L_1 of t_1;
4. $R = L_1$;
5. **for each** $t_i \in \{q - t_1\}$
6. P_1 routes t_i and sends $BF(R)$ to P_i;
7. P_i computes $R = BF(R) \bigcap L_i$;
8. P_i returns R to P_1;
9. P_1 computes the scores of q;
10. P_1 ranks returned results;
11. return the ranked results to P_{ori};

Fig. 1. A Universal Multi-term Query Approach for DHT Information Retrieval

From Figure 1, we can conclude that the universal approach is too slow for a multi-term query since it uses an iterative method, not a parallel one. Secondly, the P2P networks are so dynamic that a universal approach is sometimes unavailable since P_i maintaining the i^{th} term abruptly left the networks. Lastly, to some extent, the Bloom Filter technique is employed to reduce the bandwidth consumption, but it still requires many rounds of transmission of the query and $BF(L_i)$, whose cost is still not trivial.

2.2 Related Work

[5] presented a symmetrically distributed peer-to-peer search engine based on a DHT. Since multiple-term queries dominate the search workload, optimizing them is important for end-user performance. So, [5] focused on minimizing network traffic for

multiple-term queries. A technique developed in the database community [5] is adopted to perform the join more efficiently. This technique transmits the Bloom Filter of inverted lists instead of the inverted lists themselves. Fagin's algorithm [3] is adopted to compute the top-k results without transmitting the entire inverted lists. This algorithm transmits the inverted lists incrementally and terminates early if sufficient results have already been obtained.

Several techniques [6] were combined to reduce the cost of a distributed join, including caching, Bloom Filter, document clustering, etc. The KSS [7] system pre-computes and stores results for all possible queries consisting of up to a certain number of terms. Unfortunately, the number of possible queries grows exponentially with the number of terms in the thesaurus. Caching of query result sets as the performance improvement for an existing distributed index is utilized by a few P2P indexing methods. [14] used caching to improve the search efficiency while processing range queries. [1] proposed a log-based query algorithm in hybrid P2P networks.

In order to further reduce the chance of missing relevant documents, [15] proposed automatic query expansion to alleviate the degradation of quality of search results due to the selective replication. Query expansion scheme automatically identifies additional terms relevant to a query and also searches nodes responsible for those terms. Automatic query expansion improves precision, particularly when documents are published under very few top terms. By analyzing various techniques query expansion in P2P IR and applying them to structure building and searching, [16] found that query expansion is likely to enhance recall in these distributed settings: when working with compact and incomplete peer descriptions, query expansion using co-occurrence data can improve recall by about 10%. It has been shown in [17] that query expansion can greatly improve distributed IR.

3 Two Query Expansion Mechanisms

A query expansion mechanism extracts the most informative terms from the top-returned documents as query expansion terms. In the IR systems, a user often inputs several query terms, which are interrelated and even compose a complete sentence. e.g., when a user wants to know what the significance of word order is in a search engine, he/she inputs a query terms including "*significance word order search engine*". So, query expansion terms should firstly be extracted from the same sentence with query terms. Then we give the following two expansion mechanisms. To estimate the relatedness or independence of other correlated terms with given query terms, each sentence of a document is viewed as a multi-set.

3.1 Term Co-occurrence Expansion Mechanism

The frequency of a term is approximately proportional to the reverse of its position in the list of all the terms ranked by term frequencies in a collection of documents, as Zipf's Law goes. It is not difficult for us to infer that the number of terms that occur only once in the collection is roughly half the size of the collections vocabulary [1]. Therefore if we discard those terms that occur only once, we could decrease computational cost greatly. And hence, each multi-set does not include those single-occurrence

terms. If two terms are correlated (or constitute a phrase-like structure), then they are expected to appear simultaneously in many documents. Given the presence of one of the terms in a sentence, the chance of the other occurring within the same sentence is likely to be relatively high. On the contrary, if two terms deal with independent concepts the occurrences of the terms should not be strongly correlated. So, the following Formulas are given to measure the mutual dependence of two terms:

$$Cooccurrence(X,Y) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} cooccurrence(x,y) \tag{1}$$

$$Occurrence(Y) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} occurrence(y) \tag{2}$$

where n and m are respectively the number of returned documents and sentences of a returned document. X is a query term, while Y is a candidate term for expansion. For two terms Y and Z, if $Cooccurrence(X,Y)$ is larger than $Cooccurrence(X,Z)$, Y is more likely to become a candidate one for expansion than Z; if $Cooccurrence(X,Y)$ is equal to $Cooccurrence(X,Z)$ and $Occurrence(Y)$ is less than $Occurrence(Z)$, Y is more likely to become a candidate one than Z.

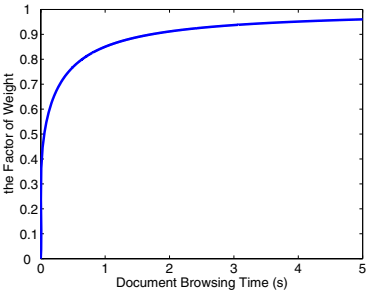
3.2 Sentence-Grained Mutual Information Expansion Mechanism

For two random variables, the mutual information is a quantity that measures the mutual dependence of them in probability and information theory. Mutual information is also a criterion commonly used in statistical language modeling of term associations and related applications [18]. In this paper, we design sentence-grained mutual information as follows, which measures the correlation degree of a query term *X* and a candidate term *Y* for expansion. Formally, the mutual information of two different terms (discrete random variables) *X* and *Y* can be defined as shown

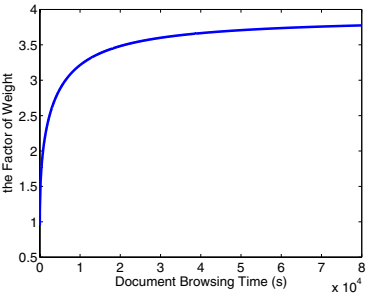
$$I(X,Y) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \tag{3}$$

where n and m respectively the number of returned documents and sentences of a returned document, $p(x,y)$ is the joint probability distribution function of *X* and *Y*, and $p(x)$ and $p(y)$ are the marginal probability distribution functions of *X* and *Y* respectively. The greater $I(X,Y)$ is, the more likely *Y* becomes a candidate query expansion term of *X*. *X* and *Y* are irrelevant if $I(X,Y)$ has a natural value of zero.

Let a query $q = \{t_0, t_1, \ldots, t_{n-1}\}$, expansion terms of t_i are extracted according to *Co-occu* and *Sg-MI* expansion mechanisms. For q viewed as a document, different term is set to have different frequency, e.g., $\{t_0, t_1, \ldots, t_{n-1}\}$ is a an arithmetic progression, where the difference of any two successive members is a constant Δf, i.e., $t_0 = t_{n-1} + (n-1)\times\Delta f$ ($\Delta f > 0$). So, the weight of each term may be calculated. For the query q, its expansion terms consist of those of each term t of q according to the ranked results of the product of $t's$ weight and the score of $t's$ expansion term in *Co-occu* or *Sg-MI* expansion mechanism.



(a) $0 - t_1$ ($t_1 = 5s$)



(b) $t_2 - \infty (t_2 = 8s, c = 3.9)$

Fig. 2. The Impact of the Weight Factor

4 Implicit Relevance Feedback Based on Users' Behaviors

In the DHT IR systems, the usual way to obtain a user's profile would be to ask, for each returned document, whether the user found it relevant, and to acquire the user's profile from the answers. However, it is difficult for us to give this kind of explicit feedback. Instead, relevance can be inferred from implicit feedback derived from document reading time, or by monitoring other behaviors of the user, such as downloading and printing.

Relevance feedback is the most popular query reformulation strategy. In a relevance feedback cycle, the user profile is presented with the retrieved documents, and then the user marks those which are relevant after checking them. In fact, only the top $n (n = 10, \ldots, 30)$ ranked documents are considered as relevant documents and need to be examined. For the top n ranked documents or the documents after them, when the user browses a document, there are three time slices: from 0 to t_1, the document is implicitly viewed as the irrelevant one with the query terms, and then its position is lowered in the top n documents; from t_1 to t_2, its position is not changed because it can not be determined whether it is relevant or irrelevant; larger than t_2, its position is promoted. So we design a function based on browse time to change a document's position. It reads as shown in Formula 4.

$$f(t) = \begin{cases} \frac{1}{\log_{10}(10 + \frac{1}{t/t_1})}, & 0 < t \leq t_1 \\ 1, & t_1 < t \leq t_2 \\ \frac{c}{\log_{10}(10 + \frac{10^c}{t/t_2})}, & t_2 < t \end{cases} \tag{4}$$

When the user decides to download the document, its weight should be promoted to a maximum multiple c. Figure 2(a) shows that the shorter document browse time is, the smaller the factor of weight is. From $0s$ to near to $1s$, an impressive promotion of the factor is produced, while the factor is steadily promoted to be close to 1 from $1s$ to $5s$. From $8s$ to ∞, there is a similar law with the time slice of $0s - 5s$. If a document is only read and not downloaded, the factor of weight only comes near to c for the document browse time of ∞ (Figure 2(b)). So, Formula 4 satisfies the real P2P systems.

Since the collection make-up is unknown to the most users, it is difficult for them to formulate queries which are well designed for retrieval purposes. As observed with web search engines, the users of information retrieval systems, especially DHT IR systems, might obviously need to spend a great deal of time reformulating their queries to accomplish effective retrieval. For the readability and understanding of this paper, the standard Rocchio reads as shown:$\boldsymbol{q_m} = \alpha \boldsymbol{q_0} + \beta \frac{1}{|D_r|} \sum_{\boldsymbol{d_j} \in D_r} \boldsymbol{d_j} - \gamma \frac{1}{|D_{nr}|} \sum_{\boldsymbol{d_j} \in D_{nr}} \boldsymbol{d_j}$, where $\boldsymbol{q_m}$ is a modified query vector; $\boldsymbol{q_0}$ is an original query vector; α, β, γ: weights (hand-chosen or set empirically), α/β is set to 16 [15] when a small number f ($f = 10$) of best matching documents are retrieved, while α/β is set to 21 when there are 30 retrieved documents for the current query in our experiment. Negative term weights are commonly ignored, so γ is set to 0; D_r is set of known relevant document vectors; D_{nr} is set of known irrelevant document vectors. New query moves toward relevant documents and away from irrelevant documents.

5 An Effective Query Method for Fast DHT Information Retrieval

5.1 Peer Selection Policy

In the traditional IR model: Vector Space Model [19] (VSM), the similarity between two vectors is generally measured as the cosine of the angle between them, where their weights can be calculated by the TFIDF rule [19]. In unstructured P2P networks, however, it is impossible for us to calculate the global IDF values since the calculation of them involves counting the document frequency of terms. But it is easy for us to calculate the global IDF values since the same term is maintained by the same peer in DHT networks. So, an effective formula of calculating them reads as follows: $IDF_{term} = \log(N/DF_{term})$, where N is a maximum unsigned integer.

We present a peer selection policy that combines the average term weight w_t^{avg} with the maximum term weight w_t^{max} that can be calculated by the TFIDF rule, where the w_t^{avg} for term t of a peer p is the average term weight containing t, and the w_t^{max} value for term t of p is the weight of t in the document d that has the maximum weight of t among all documents at p. The collection score s_p of p with regard to a query q's first term is calculated: $s_p = \mu \times w_t^{max} + (1 - \mu) \times w_t^{avg}$, where the parameter $\mu \in [0, 1]$ is utilized to emphasize the importance of w_t^{max} versus w_t^{avg}. For all the relevant peers, the scores s_p are calculated and sorted in descending order for obtaining the final peer ranking.

5.2 Query Log Updating Policy

To help deal with this problem, we first give a definition of a query log.

Definition 1. *A query log is a five-tuple*

$$Record = [\ ID_{ori}, Q, E, (R(P_i, D_j), \dots), TS\ ]$$

where ID_{ori} is a query originator. Q is a set of query terms. E is a set of expansion terms. $R(P_i, D_j)$ is a pair, which represents the D_j document in node P_i, while D_j is commonly represented by an unique ID, which is the MD5 or SHA-1 value of a

document. There are top n ranked pairs in a query log. TS is a timestamp of this query log. A query log is stored by a node, which indexes the first term of Q.

If the update conditions for a new query log data item d are fulfilled and the query log size S_{ql} exceeds its limit, a replacement policy is used for a query log update policy. Depending on the policy, a certain ranking value $R_d(t_i)$ may include the number $Q_q(t_i)$ of query requests at the current peer for a query q during the time interval $t_i =_{def} [(i-1)\Delta t, i\Delta t]$. If a query log is inserted into the query logs during the time interval t_i, the query q with the minimal ranking value $R_q(t_{i-1})$ is replaced. The measurement values $R_q(t_i)$ are exploited in the subsequent time interval t_{i+1} to decide which query log is a candidate for being replaced. Another condition for inserting a data item d into the query logs has to be fulfilled in order to ensure that d does not replace a more useful data, which is done by checking if $Q_q(t_i) > Q_r(t_i)$ for the data r with the minimal replacement value $R_r(t_{i-1})$. So, we present the following definition of a **fresh query log**:

Definition 2. ***Fresh Query Log:*** *A query log data item d is fresh, iff $Q_d(t_{i-1}) > Q_{th}$ and $Q_d(t_i) > Q_r(t_i)$ for the data r with the minimal replacement value $R_r(t_{i-1})$, where Q_{th} is a given threshold.*

On the one hand, the standard of kicking a data item out should not be directly related with the last access time for it, like in the least recently used (LRU). On the other hand, a decision to replace a data item from the query logs is made by the number of references to it. It would be reasonable to remove data items that are old and accessed infrequently first. The number $Q_d(t_i)$ of a data item requests during time interval t_i is another measure indicating its popularity. The more requests are seen at a peer, the more popular it is. Hence, we define the freshness of a data item d as shown in Formula 5, where t_c and t_a are respectively the current and last access time, λ is the *freshness factor*.

$$F_d(t_i) = \frac{1}{\lg(10 + t_c - t_a)} + \sum_{j=0}^{i} \lambda^{i-j} \times Q_d(t_j) \tag{5}$$

Since the data d is not replaced during t_0, we can include historical values for the subsequent time intervals. This avoids too fast reacting on very frequent changes of the data requests and smooths the variation over time. The parameter ν is a *damping factor*. We propose the following ranking value of query log replacement for the data d.

$$R_d(t_i) = \begin{cases} F_d(t_0), & \text{if } i = 0 \\ \frac{F_d(t_i) + R_d(t_{i-1})^{\nu}}{2}, & \text{if } i > 0 \end{cases} \tag{6}$$

The foundational idea for inserting a query q into the query logs is that the number of query requests exceeds a given threshold Q_{th} at time $t \in t_i$, i.e. $Q_q(t_{i-1}) > Q_{th}$. If a query log is inserted during the time interval t_i, it is not replaced during this period. The idea behind this is that we assume that the query log is only inserted into the cache, because this increases the system performance during the current time interval t_i. Hence, we present the following query log update algorithm as shown in Figure 3.

Algorithm 1. UpdateQueryLog(d, C_p)
Input: d is an arriving data item
C_p is the cache of a peer p
Output: C_p
1. **if** $d \in C_p$
2. Update $d's$ access time and frequency;
3. **else**
4. **if** $sizeof(d + C_p) \leq Maximum$
5. Cache data item d in C_p ;
6. **else**
7. **while** $sizeof(d + C_p) > Maximum$
8. **if** d is a **Fresh Cache** //using Definition 2
9. Delete cache r with minimal $R_r(t_{i-1})$;
10. **else**
11. return;
12. Cache data item d in C_p;

Fig. 3. Update query log algorithm

Algorithm 2. QueryInDHT(P_{ori}, q)
Input: P_{ori} is the query originator
q is the query text of P_{ori}
Output: the valid pairs of peers and documents
1. Send a query text q to the DHT networks;
// P_i is a node that stores the first term of q
2. **if**($QueryLog_{P_i} == \phi$)
3. Get top n peers S including $q's$ first term;
//RPs is the pairs of $R(P_j, D_k)$
4. RPs = QueryDocuments(P_{ori}, q, S);
5. RPs = Re-rank RPs using Formula 4;
6. Get expansion from RPs and log it on P_i;
7. return RPs;
8. **else**
9. $Match = 0$;//0: no match
10. **for each** query log $ql \in QL$
11. $tmpM$ = Match q with query text of ql;
12. **if**($tmpM > Match$)
13. $Match = tmpM$;
14. **if**($Match == 2$) //2: perfect match
15. **if**($OL_{R(P_j,D_k)} \geq OL_{th}$)
16. return the valid pairs of $R(P_j, D_k)$;
17. **else** //route q based on its query text
18. Add the expansion terms to q, goto 3;
19. **else if**($Match == 1$) //1: sub-match
20. Add expansion terms to q, goto 3;
21. **else**
22. View the query q as a new query, goto 3;

Fig. 4. An Effective Query Approach for Fast DHT Information Retrieval

5.3 Query Method

In the IR systems, query terms in a different order cause different search results [20]. The significance of key words is decreased from left to right in a search engine. e.g., there are two queries: "*significance order word search engine*" and "*search engine word order significance*", which are submitted to the Google search engine[1]. Their query terms are exactly identical, but the order of them is different. The top 10 retrieval results that are returned by Google are completely different[2]. In this paper, the first term of query terms is viewed as the primary key, which fells on a peer in DHT networks. On the other hand, the first term is not indexed in the DHT networks because there is very big difference between query terms and indexed ones since synonymical and polysemous terms exist in great numbers. This is so called "*dictionary problem*" [21], which restricts applications of P2P systems, especially DHT. If the first term is not indexed, it is replaced by the second term, and so on. Based on query expansion, users' browse and download behaviors and search history, we propose an effective query approach for DHT information retrieval as shown in Figure 4.

[1] http://www.google.com
[2] This test was conducted on 2007-01-29.

A user sends a query text including one or more terms. The first term is viewed as the primary key, which is used to locate a node P_i in DHT networks (line 1). The standard method is used and queries are not expanded (line 2 - 7). The top n promising peers are determined with peer selection policy (line 3). The procedure of getting the relevant documents from S reads as shown in Figure 5 (line 4). The expansion from RPs is logged on P_i (line 6). Afterward, based on the user's behaviors: browse time and download whether or not (implicit relevance feedback) and top n documents (pseudo relevance feedback), query expansion terms are extracted (line 6). The query log is recorded, which is sent to the node P_i that saves the indexes of that primary key (line 6). When the query log size exceeds the available capacity, old query logs are discarded to make room for new query logs, where the replacement policy follows the Figure 3 rule (line 6). Let QL be a set of query logs (line 9). Let $OL_{R(P_j,D_k)}$ be the number of online $R(P_j, D_k)$, OL_{th} be a given threshold (line 14).

Algorithm 3. QueryDocuments(P_{ori}, q, S)
Input: P_{ori} is the query originator
q is the query text of P_{ori}
S is a set of peers including $q's$ first term
Output: the valid pairs of peers and documents
1. **for each** peer $p \in S$
2. Use VSM to search relevant documents D; //similarity value (SV)
3. Get top n query SVs with heap sort;
4. Send them to P_{ori};
5. Use merge sort in P_{ori} to get n^{th} SV;
6. Use n^{th} value to filter irrelevant documents and get the relevant documents;
7. return the pairs of $R(P_j, D_k)$;

Fig. 5. The Algorithm to Get the Relevant Peer-Document Pairs

Let's give an example: Figure 6 shows the procedure of DHT IR of this paper. Peer E issues a query q "*significance order word search engine*" to the DHT network. The query q is routed to A, which maintains the index of "*significance*" (step 1). A forwards q since it has the information of top n peers, which have the term "*significance*" (step 2). The algorithm: QueryDocuments in Figure 5 is used to get the relevant peer-document pairs, which are returned to E (step 3). Based on E's behaviors and Formula 4, the returned documents are re-ranked. According to the re-ranked documents, two expansion mechanisms are used to get a record of query log as shown in Definition 1 (step 4). The record is sent to A, which is used to accelerate future retrieval for the same query in the DHT network (step 5).

5.4 Cost Analysis

We present a simple but reasonable cost analysis for the proposed multi-term query approach that serves as a feasibility check with respect to scalability. We only focus on communication costs, as local-processing and storage costs are relatively trivial in our setting. For each query, the communication cost contains the costs of step 2, 3 and 5 in Figure 6. It is: $[|S| \times len(Q)] + [|S| \times n \times 4] + [20 + len(Q) + len(E) + \sum_{p \in S_s}(20 + |D_p|) + 4]$, where a query similarity value and time stamp are respectively $float$ and $long$, a peer ID is represented by SHA-1 (20 bytes), S is a set of peers including $Q's$ first term, Q and E are respectively sets of query and expansion terms, D_p is a set of relevant documents in a peer p.

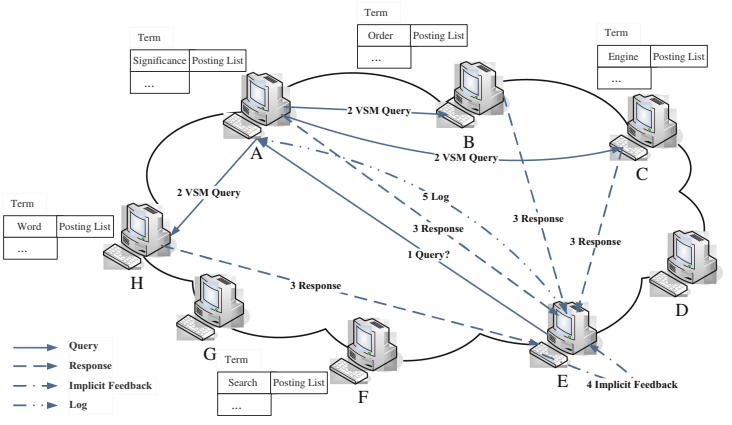


Fig. 6. An Example of DHT Information Retrieval

We assume that there are $N = 10^6$ peers and $Q_t = 5$ terms that have 7 terms averagely for each query, each peer issues a query per 5 minutes. For a multi-term query, we further suppose that the peer that is responsible for the first term decides the 100 most promising peers by using peer selection policy. Let the number of retrieval results and expansion terms be 30 (n=30) respectively. For $|D_p|$, it is one at worst. So the communication cost is: $[1,00 \times 7 \times 5] + [1,00 \times 30 \times 4] + [20 + 7 \times 5 + 7 \times 30 + (20 + 20 \times 30) + 4] = 16,389 bytes$ per query. With $N = 10^6$ peers each with a query per 5 minutes, this results in $N \times \frac{1}{60 \times 5} \times 16,389 \approx 4.97 \times 10^7$ bytes per second, less than 100 MBytes/s for the entire P2P network. Based on this coarse analysis, we can conclude that the approach does not bring any critical performance challenges and seems very feasible also from a scalability standpoint.

6 Experimental Settings and Results

6.1 Data Sets and Experimental Settings

There has been no standard data for evaluating the performance of information retrieval in DHT networks, so we use the Chinese Web Test collection (CWT200g) [22], which is a large scale Chinese web page. 149,287 web pages are randomly selected from CWT200g. There are 218,155 words in our dictionary (Dict.dat: 1,381,623 bytes/1.4M without compression), which is released in DHT systems. To simulate the P2P network environment, we adopted the live peers' online information log of Maze[3] from 2006-12-17 to 2006-12-31.

6.2 Evaluation Methodology

We randomly selected key words as queries test set from the "*title*" field of the selected test document collection. It is expensive for us to obtain relevance judgments for so

[3] A P2P file sharing system with millions of registered users, http://maze.pku.edu.cn.

Table 1. Parameter and settings

Parameter	Default	Description
Query number	10,000	# of queries sent by peers
Peer number	2,048	# of peers
Expansion size	30	# of expansion terms of a query
Query size	1∼3	# of terms of original queries
N	30	# of top n returned documents
t_1	5s	The first time slice threshold
t_2	8s	The second time slice threshold
c	3.9	Weight maximum multiple
α/β	21	he ratio of weight factor
Sentence delimiter:exclamation mark,full stop,interrogation		

Table 2. Queries for test

No.	Query text
1	Study abroad
2	Talent information
3	Employment
4	System maintenance
5	Drama
6	Air Line
7	Sports star
8	Education theory development
9	Scientific research institution
10	Computer malfunction

many automatically-generated queries. Instead, we used the retrieval results that are achieved by Google Desktop Search[4] from a single large collection as the baseline, and measured how well the P2P network could reproduce this baseline. The single large collection is the subset of the CWT200g used to define peer contents in the P2P network, and accuracy was measured over the top n documents retrieved for each query. Although this methodology is not perfect, it is reasonable because distributed retrieval systems are not yet better than the "*single collection*" baseline. Accuracy is measured with forms of set-based $precision$ ($P = \frac{|A \cap B|}{|A|}$), $recall$ ($R = \frac{|A \cap B|}{|B|}$) for a query q posted by a peer, A is the set of the documents returned by retrieval in the P2P network, B is the set of top n ranked documents returned by retrieval using the single collection. $P(q)$ describes how much irrelevant material the user may have to look through to find the relevant material. $R(q)$ captures the fraction of relevant documents a retrieval algorithm can identify and present to the user.

6.3 Experiments

In our experiments, we evaluated four combined query expansion models:

- Standard: As a first model, query terms are not expanded.
- Cooccs: It works very similarly to the Standard model, with the only difference that queries are expanded with terms that often co-occur with query terms in the training corpus. A likelihood ratio measure is used for computing the significance of co-occurrences. Similarities were computed [16], i.e. direct keyword matches are considered four times as valuable as matches arrived at by query expansion.
- Co-occu: In this case, we use Co-occu query expansion mechanism, please refer to 3.1.
- Sg-MI: Sg-MI is used as the query expansion method, as described in 3.2.

For Co-occu and Sg-MI, based on the Standard strategy, they expand queries "*on the fly*" by implicit and pseudo feedback: top n ($n = 30$) terms occurring in relevant

[4] http://desktop.google.com/

documents are added to the query with term co-occurrence expansion mechanism as these documents are found on the peers. Similarity between queries and documents is obtained by using VSM.

Experiment 1: Recall Relevance feedback is most useful for increasing recall in situations where recall is important. We selected a moment of every day at random and then let each of them search for all of randomly selected test queries. Cumulative recall is recorded for each query text the query made. Macro averaging is then used, i.e. recall values are first computed for each query instance separately and then averaged over all instances. Comparison of query recall is shown in Table 3 and Figure 7.

Table 3. Comparison of query recall

Times	# of Queries	# of peers visited	# of online peers[a]	# of peers publishing query terms	Recall			
					Standard	Cooccs (impr. %)	Co-occu (impr. %)	Sg-MI (impr. %)
Sun Dec 17 20:03:49 2006	67	701	716	1836	40.22	42.37(+5.35)	46.97(+16.78)	47.85(+18.97)
Mon Dec 18 01:19:23 2006	52	456	473	1815	24.51	26.95(+9.96)	31.43(+28.23)	31.27(+27.58)
Tue Dec 19 12:03:45 2006	83	565	576	1796	31.43	33.91(+7.89)	37.15(+18.20)	36.56(+16.32)
Wed Dec 20 10:13:44 2006	78	449	456	1896	24.34	27.21(+11.79)	31.15(+27.98)	31.15(+27.98)
Thu Dec 21 16:23:47 2006	45	626	645	1743	38.01	40.69(+7.05)	44.96(+18.28)	44.50(+17.07)
Fri Dec 22 09:59:24 2006	91	447	454	2014	21.23	24.18(+13.90)	27.59(+29.96)	26.63(+25.44)
Sun Dec 24 18:59:27 2006	56	737	757	2005	35.78	38.94(+8.83)	42.95(+20.04)	44.09(+23.23)
Mon Dec 25 07:43:43 2006	103	169	170	2030	9.21	12.44(+35.07)	16.17(+75.57)	17.51(+90.12)
Tue Dec 26 13:03:45 2006	73	674	694	2021	33.69	37.54(+11.43)	41.58(+23.42)	40.69(+20.78)
Wed Dec 27 12:59:25 2006	94	666	686	2019	32.20	36.19(+12.39)	40.99(+27.30)	41.15(+27.80)
Thu Dec 28 17:19:26 2006	57	688	707	1991	36.28	40.13(+10.61)	43.54(+20.01)	45.47(+25.33)
Fri Dec 29 15:19:26 2006	82	691	713	1994	35.58	39.98(+12.37)	43.52(+22.32)	43.85(+23.24)
Sat Dec 30 08:19:24 2006	69	197	199	2011	10.70	15.34(+43.36)	19.39(+81.21)	20.95(+95.79)
Sun Dec 31 21:03:50 2006	112	807	827	2013	42.58	47.40(+11.32)	51.17(+20.17)	50.85(+19.42)
Total averaged	75.86	562.36	576.64	1941.71	29.7	33.09(+14.38)	37.04(+30.68)	37.32(+32.79)

[a] It includes that of peers visited that provide sharing service and that of active peers that open the Maze client. The MazeSrv process is closed in Maze system and can not provide sharing service.

As Table 3 shows, three algorithms based on query expansion improve recall compared with Standard method. Co-occu attains maximum recall improvement 81.21%, average recall improvement 30.68%. Sg-MI attains maximum recall improvement 95.79%, average recall improvement 32.79%, while Cooccs only attains maximum recall improvement 43.36%, average recall improvement 14.38%. Compared with Cooccs, Co-occu and Sg-MI are not based on document-grained, but based on sentence-grained. To some degree query terms may convey a meaning, which form a sentence. Therefore Co-occu and Sg-MI greatly improve recall compared with Cooccs.

Experiment 2: Precision at 10 Standard Recall Levels. The log-based query algorithm is implemented. Based on it, standard, Cooccs, Co-occu and Sg-MI are compared in this experiment. To evaluate the retrieval performance of the four algorithms over all test queries, we average the precision values at each recall level as follows:

$$\overline{P}(r) = \frac{1}{N_q} \sum_{i=1}^{N_q} P_i(r) \tag{7}$$

where $\overline{P}(r)$ is the average precision at the recall level r, N_q is the number of queries used, and $P_i(r)$ is the precision at recall level r for the i-th query.

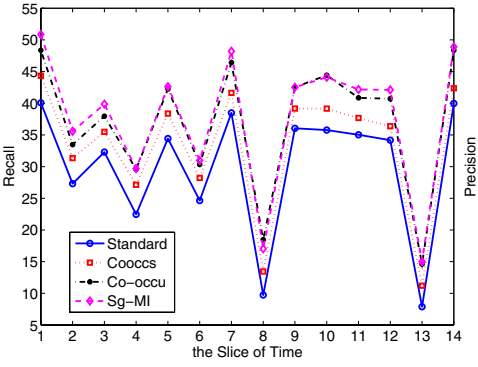


Fig. 7. Comparison of query recall

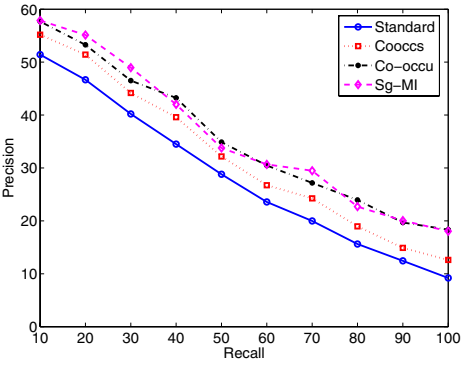


Fig. 8. Average recall vs. precision figures

As Table 4 and Figure 8 show, three algorithms based on query expansion achieve better results than Standard method. Co-occu attains maximum precision improvement 99.13%, average precision improvement 36.43%. Sg-MI attains maximum precision improvement 95.98%, average precision improvement 37.14%, while Cooccs only attains maximum precision improvement 37.02%, average precision improvement 16.67%. With the accumulation of query logs, advantages of query expansion are reflected so that precision is improved. The accumulation of query logs implies the accumulation of correlation judgment of query terms and query expansion terms, which is built by users.

Table 4. Comparison of precision at 10 recall levels

Recall	Standard	Cooccs (↑ %)	Co-occu (↑ %)	Sg-MI (↑ %)
10	51.42	55.18 (+7.31)	57.68 (+12.17)	57.86 (+12.52)
20	46.66	51.42 (+10.20)	53.27 (+14.17)	55.12 (+18.13)
30	40.21	44.17 (+9.85)	46.51 (+15.67)	48.95 (+21.74)
40	34.50	39.59 (+14.75)	43.24 (+25.33)	42.01 (+21.77)
50	28.81	32.17 (+11.66)	34.89 (+21.10)	33.78 (+17.25)
60	23.57	26.74 (+13.45)	30.47 (+29.27)	30.67 (+30.12)
70	19.98	24.26 (+21.42)	27.17 (+35.99)	29.47 (+47.50)
80	15.62	18.96 (+21.38)	23.95 (+53.33)	22.69 (+45.26)
90	12.45	14.90 (+19.68)	19.69 (+58.15)	20.06 (+61.12)
100	9.21	12.62 (+37.02)	18.34 (+99.13)	18.05 (+95.98)
Avg.	28.24	32.00 (+16.67)	35.52 (+36.43)	35.87 (+37.14)

Table 5. Evaluation of feedback

No.	Appr. I	Appr. II	Diff.
1	27.76	29.21	1.45
2	40.22	47.95	7.73
3	30.13	41.04	10.91
4	41.1	49.77	8.67
5	40.92	43.81	2.89
6	29.92	30.57	0.65
7	37.73	40.31	2.58
8	33.3	36.89	3.59
9	40.71	42.17	1.45
10	41.12	43.89	2.78
Avg.	36.29	40.56	4.27

Experiment 3: Precision Histograms. The R-precision metrics for several queries can be used to compare the effectiveness of two retrieval approaches, which can be defined as follows:

$$RP_{II/I} = RP_{II}(i) - RP_{I}(i) \tag{8}$$

where $RP_{II}(i)$ and $RP_{I}(i)$ are the R-precision values of the retrieval approaches II and I for the i-th query. The purpose of this experiment is to investigate the effectiveness of

implicit relevance feedback based on users' browse and download behaviors. Among the four methods, the effectiveness of Sg-MI is the best one. So we selected Sg-MI as the basic evaluation method. Sg-MI without and with implicit relevance feedback are respectively called I and II. For the queries in the table 2, we randomly selected 5 moments to send them and recorded our browse and download behaviors, and then 10 different precisions for each query are achieved when recall ranges from 10% to 100%. Finally, the average R-precision values of the two retrieval algorithms for the i-th query are achieved as shown in Table 5.

As Table 5 and Figure 9 illustrate, II performs better for ten queries than I. Compared with I, II attains maximum R-precision improvement 10.91% and average R-precision improvement 4.27%. The longer a document is browsed, the more important it is. If it is downloaded, it is very relevant to a query, which is sent by a user. The process includes the users evaluation on relevance of the query and the document. So II is superior to I.

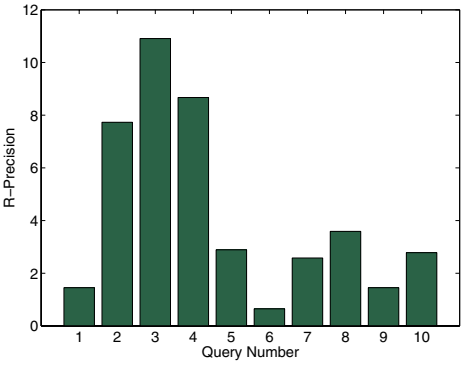


Fig. 9. Precision histogram for test queries

7 Conclusions and Future Works

In this paper, we propose an effective query method based on query expansion mechanisms, users' behaviors and search history to gracefully support multi-term query in the DHT networks. We firstly design two query expansion mechanisms and an implicit user relevance feedback approach based on users' browse and download behaviors. To support multi-term queries for DHT IR, a peer p that maintains the first term of a multi-term query q plays an important role to direct the query to other peers included the first term. For each query, its log is recorded in peer p for accelerating the same query in the future. For the subset of q, its expansion terms are employed to improve the query accuracy. From recall, precision at 10 standard recall levels and precision histograms, we conducted 3 experiments, which show that it greatly improves effectiveness compared with previous methods.

We believe that possible directions to future work include some research topics such as preventing query drift for Co-occu and Sg-MI methods in DHT systems, reducing the storage cost of node's query logs because of hot terms and etc. Retrieval based on documents

content (e.g. web search and digital libraries) is a hot research topic in DHT system, which is worth of further theoretical research, especially in dynamic distributed environment.

References

1. Lu, J., Callan, J.P.: Content-based retrieval in hybrid peer-to-peer networks. In: CIKM, pp. 199–206 (2003)
2. Lu, J., Callan, J.P.: User modeling for full-text federated search in peer-to-peer networks. In: SIGIR, pp. 332–339 (2006)
3. Suel, T., Mathur, C., Wu, J., Zhang, J., Delis, A., Kharrazi, M., Long, X., Shanmugasunderam, K.: Odissea: A peer-to-peer architecture for scalable web search and information retrieval. In: WebDB 2003 (June 2003)
4. Bender, M., Michel, S., Parreira, J.X., Crecelius, T.: P2p web search: Make it light, make it fly (demo). In: CIDR, pp. 164–168 (2007)
5. Reynolds, P., Vahdat, A.: Efficient peer-to-peer keyword searching. In: Endler, M., Schmidt, D.C. (eds.) Middleware 2003. LNCS, vol. 2672. Springer, Heidelberg (2003)
6. Li, J., Loo, B.T., Hellerstein, J., Kaashoek, F., Karger, D.R., Morris, R.: On the feasibility of peer-to-peer web indexing and search. In: IPTPS 2003 (February 2003)
7. Gnawali, O.D.: A keyword set search system for peer-to-peer networks. Master's thesis, Massachusetts Institute of Technology (June 2002)
8. Jansen, B.J., Spink, A., Saracevic, T.: Real life, real users, and real needs: A study and analysis of user queries on the web. Information Processing and Management 36, 207–227 (2000)
9. Chen, H., Jin, H., Liu, Y., Ni, L.M.: Difficulty-aware hybrid search in peer-to-peer networks. In: ICPP, p. 6 (2007)
10. Kwok, K.L.: A new method of weighting query terms for ad-hoc retrieval. In: SIGIR, pp. 187–195 (1996)
11. Bharambe, A.R., Agrawal, M., Seshan, S.: Mercury: supporting scalable multi-attribute range queries. In: SIGCOMM, pp. 353–366 (2004)
12. Ramabhadran, S., Hellerstein, J.M., Ratnasamy, S., Shenker, S.: Prefix hash tree: An indexing data structure over distributed hash tables (2004)
13. Zhou, M., Zhang, R., Qian, W., Zhou, A.: Gchord: Indexing for multiattribute query in p2p system with low maintenance cost. In: Kotagiri, R., Radha Krishna, P., Mohania, M., Nantajeewarawat, E. (eds.) DASFAA 2007. LNCS, vol. 4443, pp. 55–66. Springer, Heidelberg (2007)
14. Sahin, O.D., Gupta, A., Agrawal, D., Abbadi, A.E.: A peer-to-peer framework for caching range queries. In: ICDE, pp. 165–176 (2004)
15. Tang, C., Dwarkadas, S.: Hybrid global-local indexing for efficient peer-to-peer information retrieval. In: NSDI (2004)
16. Witschel, H.F., Böhme, T.: Evaluating profiling and query expansion methods for p2p information retrieval. In: P2PIR (2005)
17. Xu, J., Callan, J.: Effective retrieval with distributed collections. In: Proc. of SIGIR 1998, pp. 112–120 (1998)
18. Churn, K.W., Hanks, P.: Word association norms, mutual information and lexicography. In: Proceedings of ACL 27, Vancouver, Canada, pp. 76–83 (1989)
19. Salton, G., Wang, A., Yang, C.: A vector space model for information retrieval. Journal of the American Society for Information Science 18, 613–620 (1975)
20. Bausch, P., Calishain, T., Dornfest, R.: Google Hacks, 3rd edn., pp. 101–105. O'Reilly Media, Inc., Sebastopol (2006)
21. Furnas, G.W., Landauer, T.K., Gomez, L.M., Dumais, S.T.: The vocabulary problem in human-system communication. Communications of the ACM 30, 964–971 (1987)
22. Chinese web information retrieval forum, `http://www.cwirf.org`

Efficiently Handling Dynamics in Distributed Link Based Authority Analysis

Josiane Xavier Parreira[1], Sebastian Michel[2], and Gerhard Weikum[1]

[1] Max-Planck Institute for Informatics
Saarbrücken, Germany
{jparreir,weikum}@mpi-inf.mpg.de
[2] Ecole Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
sebastian.michel@epfl.ch

Abstract. Link based authority analysis is an important tool for ranking resources in social networks and other graphs. Previous work have presented $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$, a decentralized algorithm for computing PageRank scores. The algorithm is designed to work in distributed systems, such as peer-to-peer (P2P) networks. However, the dynamics of the P2P networks, one if its main characteristics, is currently not handled by the algorithm. This paper shows how to adapt $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ to work under network churn. First, we present a distributed algorithm that estimates the number of distinct documents in the network, which is needed in the local computation of the PageRank scores. We then present a method that enables each peer to detect other peers leave and to update its view of the network. We show that the number of stored items in the network can be efficiently estimated, with little overhead on the network traffic. Second, we present an extension of the original $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ algorithms that can cope with network and content dynamics. We show by a comprehensive performance analysis the practical usability of our approach. The proposed estimators together with the changes in the core $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ components allow for a fast and authority score computation even under heavy churn. We believe that this is the last missing step toward the application of distributed PageRank measures in real-life large-scale applications.

1 Introduction

The peer-to-peer (P2P) approach facilitates the sharing of huge amounts of data in a distributed and self-organizing way. These characteristics offer enormous potential benefit for search capabilities powerful in terms of scalability, efficiency, and resilience to failures and dynamics. Additionally, a P2P search engine can potentially benefit from the intellectual input (e.g., bookmarks, query logs, click streams, etc.) of a large user community participating in the data sharing network. Finally, but perhaps even more importantly, a P2P search engine can also facilitate pluralism in informing users about Internet content, which is crucial in order to preclude the formation of information-resource monopolies and the biased visibility of content from economically powerful sources.

A conceivable, very intriguing application of P2P computing is Web search. The functionality would include search for names and simple attributes of files, but also

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 36–49, 2008.

Google-style keyword or even richer XML-oriented search capabilities. It is important to point out that Web search is not simply keyword filtering, but involves relevance assessment and ranking search results. We envision an architecture where each peer can compile its data at its discretion, according to the user's personal interests and data production activities (e.g., publications, blogs, news gathered from different feeds, Web pages collected by a thematically focused crawl). Queries can be executed locally on the small-to-medium personalized corpus, but they can also be forwarded to other, appropriately selected, peers for additional or better search results.

In previous works [25,26], we have presented the J_P^X algorithm for decentralized computation of global PageRank scores in a P2P network. It works by combining local PageRank computation at peers and exchange of messages in the network. The authority scores obtained with J_P^X are proved to converge to the global PageRank scores that one would obtain by running the PageRank algorithm in the union of all contents in the network.

However the algorithm currently does not handle one of the main characteristics of P2P networks, namely their dynamic nature. Peers are constantly joining and leaving the network, meaning that the fully content is not always available. Moreover, peers might change what they store, for instance a user can become interested in a different topic and start to store information about this new topic instead. This has a big impact on the computation of authority scores, since the endorsement links might as well change.

In this work we propose methods to adapt the J_P^X algorithm to work under dynamics. The dynamics considered here can be of one of the two types: network dynamics and content dynamics. Network dynamics refers to changes on the peer population since nodes are continuously joining and leaving the system. Content dynamics refers to changes on the what is stored by the peers.

This paper is organized as follows. Section 3 briefly reviews the basic principles and properties of J_P^X. Section 4 shows how global statistics can be gathered using small statistical sketches that are piggy-backed onto the existing communication. These statistics are of fundamental importance to adjust the algorithm to work under dynamics. Section 5 addresses exactly these countermeasures to avoid inaccurate PageRank estimations when the underlying data changes due to node failures/departures or changing data. Section 6 presents the experimental results. Section 7 concludes the paper and gives an overview on ongoing and future work.

2 Related Work

Link-based authority ranking has received great attention in the literature. Good surveys of the many improvements and variations are given in [9,20,7,5].

The basic idea of PageRank [8] is that if page p has a link to page q then the author of p is implicitly endorsing q, i.e., giving some importance to page q. How much p contributes to the importance of q is proportional to the importance of p itself.

This recursive definition of importance is captured by the stationary distribution of a Markov chain that describes a random walk over the graph, where we start at an arbitrary page and in each step choose a random outgoing edge from the current page. To ensure the ergodicity of the chain, random jumps among pages are allowed, with smaller probability.

With the advent of P2P networks [1,31,27,28] a lot of research has been dedicated to distributed link analysis techniques has been growing.

In [32] Wang and DeWitt presented a distributed search engine framework, in which the authority score of each page is computed by performing the PageRank algorithm at the Web server that is the responsible host for the page, based only on the intra-server links. They also assign authority scores to each server in the network, based on the inter-server links, and then approximate global PageRank values by combining local page authority scores and server authority values. Wu and Aberer [33] pursue a similar approach based on a layered Markov model. Both of these approaches are in turn closely related to the work by Haveliwala et al. [17] that postulates a block structure of the link matrix and exploits this structure for faster convergence of the global PageRank computation.

Other techniques [19,11] for approximating PageRank-style authority scores with partial knowledge of the global graph use state-aggregation technique from the stationary analysis of large Markov chains. A storage-efficient approach to computing authority scores is the OPIC algorithm developed by Abiteboul et al. [2]. This method avoids having the entire link graph in one site, which, albeit sparse, is very large and usually exceeds the available main memory size. The above mentioned approaches however, are not focused on P2P networks, therefore the issue of dynamics is not addressed.

In [29], Sankaralingam et al. presented a P2P algorithm in which the PageRank computation is performed at the network level, with peers constantly updating the scores of their local pages and sending these updated values through the network. Shi et al. [30] also compute PR at the network level, but they reduce the communication among peers by distributing the pages among the peers according to some load-sharing function.

Counting the number of distinct elements in a multiset has been a well studied problem, in particular in the context of database systems [10,15,12]. The recent work by Ntarmos et al [23] considers hash sketched based counting of distinct items leveraging a distributed hash table (DHT). Our own prior work [4] considers the estimation of global document frequency in a P2P Web search engine layered on top of which is DHT. In the area of unstructured networks, which we consider in our current work, there have been a lot of research on the so called gossiping algorithms [16,3,18]. The main idea is to let peers perform random meetings as a background process along with the actual application. Peers constantly sent values to the others peers and updated according to the messages received from the other participants. These approaches are in particular well suited for the application in $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$, since $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ relies anyway on random peer meetings. In this paper, we consider a gossip based algorithm based on hash sketches [15] to estimate the number of distinct documents in the system. This differs from standard gossip-based aggregation approaches, which usually focus on computing values such as *max*, *avg*, and *count*.

3 JXP Basics

$\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ [26] is an algorithm to compute global authority scores in a decentralized manner. In [26] the authors give a mathematical proof that the $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ scores converge to the global

PageRank authority scores, i.e., the scores that would be obtained by a PageRank computation on a hypothetically centralized combined Web graph over all peers.

Running at each peer, J_P^X combines standard PageRank computations on the local portion of the Web graph with condensed knowledge on the rest of the network, which is continuously being refined by meetings with other peers. The knowledge about the non-local partition of the Web graph is collapsed into a *single* dedicated node that is added to the local Web graph, the so-called *world node*. It conceptually represents all non-local documents of the Web graph.

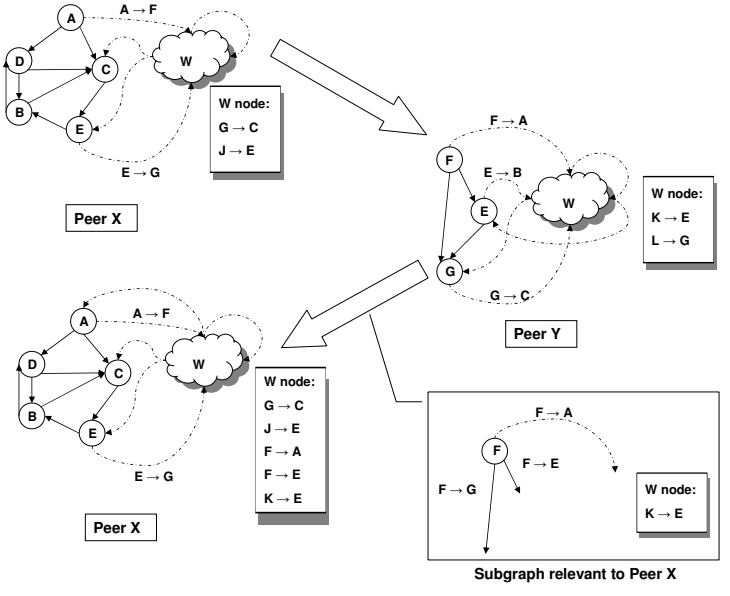


Fig. 1. Exchanging local knowledge

This is an application of the state-lumping techniques used in the analysis of large Markov models. All documents of the local Web graph that point to non-local documents will create an edge to the world node (cf. Figure 1).

Meetings with other peers in the network are used to exchange local knowledge and to improve the local approximation of global authority scores, as illustrated in Figure 1. As a peer learns about non-local documents pointing at a local document, a corresponding edge from the world node to that local document is inserted into the local Web graph[1]. Each peer locally maintains a list of scores for external documents that point to a local document. The weight of an edge from the world node to a local document reflects the authority score mass that is transferred from the non-local document; if this edge already exists, its weight is updated with the maximum of both scores. The world node contains an additional self-loop link, representing links within non-local pages. The J_P^X authority score of the world node itself reflects the J_P^X score mass of

[1] Note that such a meeting does *not* increase the number of nodes of a peer's local Web graph.

all non-local pages. Locally, each peer recomputes its local $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ scores by a standard PageRank power iteration on the local Web graph augmented by the world node.

The $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ algorithm is scalable, as the PageRank power iteration computation is always performed on small local graphs, regardless of the number of peers in the network. The local storage requirements at each peer are independent from the number of remote peers they have previously met and the size of the remote (or even the complete) Web graph, i.e., the size of the local Web graph *only* reflects the local crawl. The autonomy of peers is fully preserved by the asynchronous nature of communication and computation.

Current limitations of the algorithm is that it assumes that (i) the global size of the graph is known, and (ii) peers and their contents are static through all the computation. In [25] we addressed (i), showing that a wrong estimation of global size causes only a rescaling of the $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ scores, while the ranking order is preserved. For convergence to the true PageRank scores however, the correct graph size is needed. In case of peer dynamics only, i.e., the Web graph is fixed and peers are constantly leaving and eventually joining the network again, the convergence guarantees given in [26] still hold, with the difference that the convergence is slowed down, given that some peers are not accessible for a certain period. Dealing with content dynamics, i.e., documents being added to the network or becoming unavailable, gives a more realistic model, and is the main contribution of this current work. More details can be found in Section 5.

4 Estimating the Global Number Documents

As mentioned earlier, convergence to the true PageRank values requires the knowledge of the total number of documents in the network. In this section we propose a method for computing such value in a dynamic P2P network.

Instead of a single value, peers initialize a hash sketch [15] that represents the set of local pages. During a meeting, peers exchange the hash sketches and the local copy is updated by taking the union of both sketches (local and from the peer met). What we seek is to have the hash sketches at all peers to be the same and equal to the sketch that represents the union of all local sets. The size of the network, can then be estimated at each peer, with error bounds given by the original hash sketch work, which we will briefly discuss in the following section. Thereafter, we discuss how this gossiping algorithm can be applied to dynamic settings using a sliding window approach.

4.1 Hash Sketches

Hash sketches were first proposed by Flajolet and Martin in [15] to probabilistically estimate the cardinality of a multi set S. Hash sketches rely on the existence of a pseudo-uniform hash function $h() : S \rightarrow [0, 1, \ldots, 2^L)$. Durand and Flajolet presented a similar algorithm in [12] (*super-LogLog counting*) which reduced the space complexity and relaxed the required statistical properties of the hash function.

Briefly, hash sketches work as follows. Let $\rho(y) : [0, 2^L) \rightarrow [0, L)$ be the position of the least significant (leftmost) 1-bit in the binary representation of y; that is,

$$\rho(y) = \min_{k \geq 0} bit(y, k) \neq 0,\ y > 0$$

and $\rho(0) = L$. $bit(y, k)$ denotes the k-th bit in the binary representation of y (bit-position 0 corresponds to the least significant bit). In order to estimate the number n of distinct elements in a multi set S we apply $\rho(h(d))$ to all $d \in S$ and record the least-significant 1-bits in a bitmap vector $B[0 \ldots L-1]$. Since $h()$ distributes values uniformly over $[0, 2^L)$, it follows that

$$P(\rho(h(d)) = k) = 2^{-k-1}$$

Thus, when counting elements in an n-item multi set, $B[0]$ will be set to 1 approximately $\frac{n}{2}$ times, $B[1]$ approximately $\frac{n}{4}$ times, etc. Then, the quantity $R(S) = max_{d \in S}\rho(d)$ provides an estimation of the value of $\log_2 n$. The authors in [15,12] present analysis and techniques to bound from above the error introduced.Techniques which provably reduce the statistical estimation error typically rely on employing multiple bitmaps for each hash sketch, instead of only one. The overall estimation then is an averaging over the individual estimations produced using each bitmap.

Hash sketches offer duplicate elimination "for free", or in other words, they allow counting distinct elements in multi sets. Estimating the number of distinct elements (e.g., documents) of the *union* of an arbitrary number of multi sets (e.g., distributed and autonomous collections) - each represented by a hash sketch synopsis - is easy by design: a simple bit-wise OR-operation over all synopses yields a hash sketch for the combined collection that instantly allows us to estimate the number of distinct documents of the combined collection.

4.2 Estimating Global Counts Using Hash Sketches

In the task of estimating global counts using hash sketches, peers can benefit from the counts of all the other peers, due to the duplicate aware counting. To make the analysis tractable, lets assume for now that all peers perform their meetings in a synchronized way, i.e. after some amount of time, all peers have performed the same number of meetings. Consider one particular peer that is about to perform its m^{th} meeting. Obviously, it has already performed $m-1$ meetings in the past, as well as the peer it will meet in its m^{th} meeting. In total, both peers now double the amount of meetings they are aware of (recorded in hash sketches). We denote $C(m)$ the number of meetings a peer is aware of after the m^{th} meeting. It is easy to see that we can also write $C(m) = 2^{(m-1)}$, i.e., the number of meetings a peer is aware of grows exponentially with the number of meetings the peer has performed. From a single peer's point of view, after having performed m meetings the situation is identical with having had $C(m)$ meetings where peers do not share information about their previous meetings.

Charikar et al [10] consider the problem of estimating the number of distinct values in a column of a table. The difference to our scenario is that in a database table, the number of tuples is known, whereas in a truly distributed large scale system, the total

number of peers is unknown. In addition, we know only how many peers or documents we have seen so far, and not the frequency of observation. In practice, all we have is an estimate of distinct values given the sampling using meetings and the exchanged hash sketches, thus we cannot directly apply the estimators from [10]. The estimation of the number of distinct items, however, in a multi set is a well studied problem (cf., e.g., [21]). We will now briefly discuss on how many distinct meetings will in expectation be among the $C(m)$ meetings.

Applying the results from [21], given $C(m)$ samples of a multi set that contains n distinct elements, the expected number of elements is

$$E[distinctItems] = n(1 - e^{-C(m)/n})$$

According to [21] it can be derived that

$$\frac{C(m)}{n} = ln(\frac{n}{n - E[distinctItems]})$$

and this expression can be used to get an estimator $\hat{n}$ of the total number of distinct elements n. Then, the variance of the estimator is given by

$$\sigma^2_{\hat{n}} = \frac{n}{e^{C(m)/n} - (1 + \frac{C(m)}{n})}$$

Hence, to reach an negligible error even for really huge n, we need only few rounds of peer meeting since $C(m)$ grows exponentially.

How can we now estimate the number of documents we did not see during the estimation process. Note that this is not the same as the error in estimating the number peers (distinct meetings) since the data is not evenly spread across the peers with some peers being small and some peers being extremely large. Due to the fast estimation process, the case that few peers are not observed after some time by the majority of peers becomes very unlikely.

In practice we do not know the value of $C(m)$ since peers meet asynchronously and the online time of peers largely varies. In addition, we are of course not aware of n, the total distinct number of elements (peers or documents) in the system. We have only an estimate given by the hash sketch based sampling. The reasoning presented above shows, however, that few iterations are needed to get to a meaningful hash sketch. That does of course not include any reasoning about the quality of the estimated obtained by the hash sketches which is given in the original work by Flajolet et al and is thus nicely orthogonal to our goals.

The number of documents changes here as well as in the case when documents are leaving the system, numbers can increase or decrease. The former case can be easily handled by the estimator introduced above. The latter case requires some further improvements. Since one can easily add items to a hash sketch but one cannot remove items from such a sketch, we employ the usage of a time sliding window over multiple hash sketches. We let each peer keep an array of k hash sketches, ordered by time, the k^{th} hash sketch is considered to be the "oldest" one. After τ time steps we remove the

oldest sketch and insert an empty one at array position 1. At any time, the current estimate of distinct items is the estimate derived from the hash sketch created by forming the union of all k sketches. Obviously, newly observed items will be inserted into the sketch at position 1.

5 Handling Dynamics

Recalling the previous $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ meeting procedure, a peer selects another peer for a meeting and contacts this peer. The contacted peer then returns the information that is relevant to the peer initiator. This information consists of a list of all external pages known to the contacted peer (local + world node) that contain links to local pages at the peer initiator. Due to possible overlaps and the asynchronous nature of the algorithm, different peers might provide different score values for the same page. In these cases, the highest score is kept, since the correctness proof of the algorithm shows that scores are, at any time during the computation, upper-bounded by the true PageRank scores, i.e., the scores to which the $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ scores converge to. Therefore, keeping the highest values provides a speed up in convergence. In addition, local pages with links to pages outside the local graph do not need to know the exact location of those, since links to non-local documents are represented as links to the world node. With dynamics, however, three new events come into play, and the algorithm needs to detect them: pages can be added, modified, or deleted.

5.1 The New World Node

So far we actually did not consider the problem of invalid information kept in the World node in case of peers and/or documents leaving the system. One idea would be to keep for each document in the world node that points to a local page a list of peers that had reported a score for that particular document. The number of data to keep track at (bookkeeping) should be constant or growing sub linearly, e.g. in the order of $log(N)$ like in the case of N peers in a *Distributed Hash Tables* (DHTs) [1,31,27,28], where N is the number of peers in the network. Keeping track of all peers that store a particular document is out of question, since it would require massive amount of storage caused by overly popular pages, like for instance, `google.com` or `cnn.com`.

Instead of using a sketch based representation to keep track about the peers that have reported scores for a particular page, we opt for storing the last χ peers that reported a score, i.e., we store for each document a list of pairs (peerId, score) for the last χ scores seen for the page, along with the corresponding peer. The parameter χ can depend on the storage capacity of each peer, but we envision χ to be in order of $O(log\ N)$. This limitation to a certain length is reasonable, since the probability that all peers inside a per-document list leave is small, (analog to the size of "finger tables" in DHTs). Even if a peer removes a particular page and that page is actually in the system, it will be rediscovered, due to the basic $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ performance. Hence, the actual choice of χ is not overly crucial for the performance of $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$.

In addition to remembering external pages with outgoing edges to the local graph, the world node now needs also to keep track of external pages that are *pointed by* local

pages. This way we can correctly reconstruct both links from and to the world node. Here we also apply the approach of keeping a list of limited size containing the last χ peers met that contained the page, but no score is needed, since they do not directly influence the local scores.

5.2 JXP Meetings Adapted

In the previous version of $\mathrm{J}^{\mathrm{X}}_{\mathrm{P}}$, the meetings are of fundamental importance for the effectiveness and correctness of the algorithm. With dynamics, its role becomes even more crucial: it is through the meetings that peers will be able to detect the changes in the network. As stated before, a change can be of one of the three types: pages can be added, modified, or deleted.

Page addition is a trivial problem, since the algorithm is already designed to discover non-local documents. For speeding up convergence, a peer also sends information about pages currently in its world node to the meeting initiator (like in the previous version) . With scores lists instead of single scores, a decision has to be made about what to send for those pages. For keeping message cost small, our solution is to send a single (peerId, score) pair per page, where the score is computed by averaging all scores current known for the page. With the limit on the size of the lists, and a fair amount of meetings performed, old scores will gradually be replaced by updated, better scores, and the average is then expected to converge to the correct score of the page. For the peerId, we can simply choose the most recent peer met for that page, since chances are higher that this peer will remain for a longer period in the network.

Page deletion might occur when peers that reported information for the page have left the network or changed their contents. Whenever one of the two happens, the reference for that peer is removed from the world node. If the list of peers for a document becomes empty, it is assumed that the page no longer exist, and therefore must be removed from the world node.

It could also happen that a page had its contents modified, so it could still be reached but the new information given for that page contradicts previous information. By comparing what is already stored on the world node against what is being reported by the other peer we can detect such events. Changes on the score are not considered, since peers are constantly updating this information. What is checked is whether the outgoing edges have been modified. If so, the page is initially removed from the world node and re-added with the new information.

What is left to describe is how to detect when a peer has left the system. In P2P networks, it is very common that peers temporally leave the network and return to it a short later. In such situations, we would rather leave the world node unchanged and wait until the peer returns. Therefore, a single failed attempt to contact a peer sometimes might not be an good indication that the peer has left the network indefinitely. Instead, we keep a counter of number of consecutive failed attempts made to contact a peer, and only if this number is above a certain threshold, that can be tuned according to the network behavior, we declare that the peer no longer alive, and its references should be removed. During a successful attempt this counter is reset.

6 Experiments

6.1 Experimental Setup

J_P^X peers are implemented in Java 1.6, the peers' data collections (i.e., their local graphs) obtained by performing independent crawls on the *eu-2005* dataset, available under `http://law.dsi.unimi.it/`, and accessible using the WebGraph framework [6], available under `http://webgraph.dsi.unimi.it/`. The dataset was obtained in 2005 by crawling parts of the .eu domain, and contains 862,664 documents with 19,235,140 links. For a meeting, a peer contacts a randomly chosen peer in the network, and asks for its current local knowledge.

For evaluating the performance we compare the authority scores given by the J_P^X algorithm against the true PageRank scores of pages in the complete collection. Since, in the J_P^X approach, the pages are distributed among the peers and for the true PageRank computation the complete graph is needed, in order to compare the two approaches we construct a total ranking from the distributed scores by essentially merging the score lists from all peers. Since we are trying to evaluate the performance of J_P^X under network churn, the evaluation becomes more complicated, since the baseline, i.e., the PageRank scores of all pages currently available in the system, is not static anymore. Hence, for every change in the network, we consider the union of all pages currently in the system, and compute the baseline scores. For each of these points, we let each active peer also report its current view of the global state. We ignore at runtime how well J_P^X performs and let peers run completely independent, however, each peer reports after each meeting its current view on the global state. After the run we reconstruct at any point back in time what documents have been indexed. Then for some points in time we run the PageRank method for these documents and compare with what peers reported.

The total top-k ranking given by the J_P^X algorithm and the ranking given by traditional, centralized PageRank are compared using the scaled footrule distance [14,13], the weights contributions of elements based on the size of the rankings they are present in. More formally, given the local ranking τ and the global ranking σ, $F(\sigma,\tau) = \sum_{i\in\tau} |\sigma(i)/|\sigma| - \tau(i)/|\tau||$, where $\sigma(i)$ and $\tau(i)$ are the positions of the page i in the respective rankings. The measure is normalized by dividing it by $|\tau|/2$. We also use a *linear score error* measure, which is defined as the average of the absolute difference between the J_P^X score and the global PR score over the top-k pages in the centralized PR ranking. In addition, we report on the cosine similarity between the two vectors and the L1-norm of the vector containing the J_P^X scores.

To model peer behavior, we use previous works [22,24] that have derived mathematical models that closely represent the dynamics observed in P2P networks. More specifically, peer joins are expected to follow a Poisson distribution, i.e., the probability that n peers join the network on the next time interval can be written as $P_\lambda(n) = \frac{\lambda^n}{n!}e^{-\lambda}$, where λ is the average number of peers joining the network per time interval. Peer leaves, in turn, follow an exponential distribution. Given the average number of drop outs in one time interval (μ), the probability that a peer leaves the network after x time intervals is $F(x) = 1 - e^{-\mu x}$. Note that the interval between two consecutive Poisson events also follows an exponential distribution. In the following experiments, we used

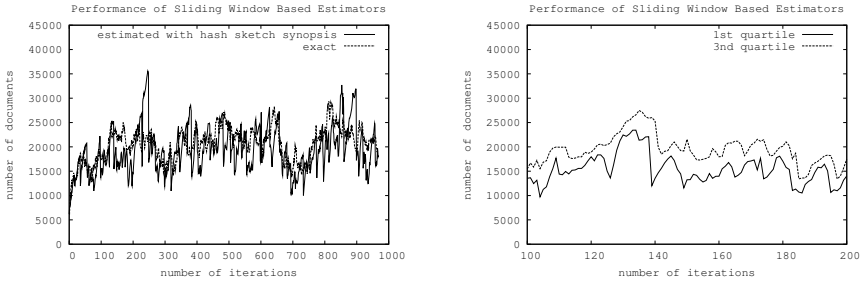


Fig. 2. Hash sketch based estimation of the number of documents under network churn

these models to generate peer dynamics. For the content dynamics, we randomly choose a percentage of the peers and replace their local graphs by performing new crawls.

6.2 Experimental Results

The experimental evaluation consists of two parts. First we report on the performance of the estimator presented in Section 4. Second, we present results on the performance of $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ under dynamics, which is the main focus of this paper.

Figure 2 (left) shows a quality of the document estimator compared to the exact values, i.e., the number of documents currently in the system. For this experiment, we simulated random peer meetings within a system of 50 peers. Each peer randomly draws from a pool of $150,000$ documents between 250 and 1000 distinct documents. Peers are either active or inactive, according to the above mentioned exponential distributions that models the peer behavior. Each peer maintains only 4 hash sketches with 2^{10} bitmaps each, resulting in a negligible storage consumption of $32KByte$. After 2 meetings, each peer shifts the sliding window over the hash sketches by one position, i.e., each hash sketch is valid only for 2 meetings. As shown in Figure 2 (left), the estimation accurately follows the exact values, with major drastic fluctuations being smoothed out. To get a deeper insight about the usability of our estimator inside $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$, we also report on the distribution of count estimates, as presented in Figure 2 (right). The variation between the first and the third quartile is remarkably small, indicating that peers nearly agree on one particular value, which is important for the performance of $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$. Note that both figures shows one particular, representative run, and that it is not smoothed over multiple runs or multiple parameter choices.

For the experiments with the adapted $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ we increased the size of the network to 1000 peers. Overlaps among local graphs are allowed, and the collection of all peers holds in total around $100,000$ documents. Peer and content dynamics are introduced in the system always after a certain number of meetings has occurred in the network. We considered both successful and unsuccessful meetings for the counter. We then varied the parameters of the peer churn and content dynamics models, to simulate different degrees of dynamics.

We present results for two scenarios: *Moderate Churn*, with join and leave rates of $100/0.1$, and a change of the contents of 1% of peers; and *Heavy Churn*, with join and

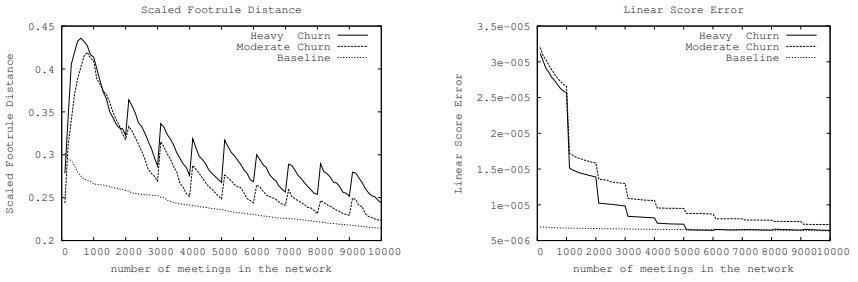


Fig. 3. Scaled Footrule distance (left) and Linear Score Error (right)

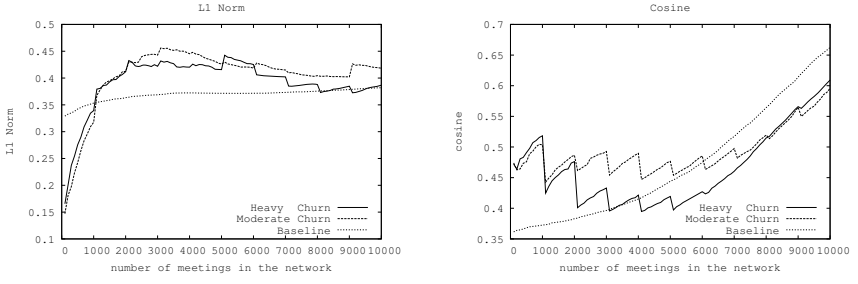


Fig. 4. L1-norm (left) and Cosine (right)

leave rates of $200/0.1$, and a change of the contents of 5% of peers. For a better understanding of the impact of dynamics the following results were obtaining without the use of our document estimator, and peers were artificially told about the correct size of the network. Figures 3 and 4 show the results obtained, where the baseline simulates the case where there is no dynamics. Note that the actual values of the linear score error are in general not meaningful: since scores correspond to stationary probability, they are expected to sum up to one, so if there is an increase of the number of pages in the network, the scores drop, which explain the behavior of the curve. However, the key insight obtained here is that the error decreases even under dynamics. The other three accuracy measures show a very nice performance of $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ under churn, in particular the L1-norm nicely follows the baseline, even though the underlying network (data) is not stable.

7 Conclusion

In this paper we have addressed the problem of computing distributed PageRank authority scores with particular emphasis on the key requirements to address the challenges of highly dynamic systems. We have identified the main shortcomings of our $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$ method, and presented means to extend the algorithm to cope with network dynamics. We have presented an estimator based on hash sketches and sliding windows to count the number of distinct documents in a dynamic network, which is one of the basic input parameters of $\mathrm{J}_{\mathrm{P}}^{\mathrm{X}}$. Secondly, but perhaps even more importantly, we have presented several

modifications to the original J_P^X data structures that allow for accurate PageRank computation even under high churn, while keeping storage requirements and message costs low. As future work we can think of ways of automatically detecting the dynamic behavior that would allow peers to adjust their local settings, for an even better performance.

References

1. Aberer, K.: P-grid: A self-organizing access structure for p2p information systems. In: Batini, C., Giunchiglia, F., Giorgini, P., Mecella, M. (eds.) CoopIS 2001. LNCS, vol. 2172, pp. 179–194. Springer, Heidelberg (2001)
2. Abiteboul, S., Preda, M., Cobena, G.: Adaptive on-line page importance computation. In: WWW Conference, pp. 280–290. ACM Press, New York (2003)
3. Bawa, M., Gionis, A., Garcia-Molina, H., Motwani, R.: The price of validity in dynamic networks. J. Comput. Syst. Sci. 73(3), 245–264 (2007)
4. Bender, M., Michel, S., Triantafillou, P., Weikum, G.: Global document frequency estimation in peer-to-peer web search. In: WebDB (2006)
5. Berkhin, P.: A survey on pagerank computing. Internet Mathematics 2(1), 73–120 (2005)
6. Boldi, P., Vigna, S.: The webgraph framework i: compression techniques. In: WWW, pp. 595–602 (2004)
7. Borodin, A., Roberts, G.O., Rosenthal, J.S., Tsaparas, P.: Link analysis ranking: algorithms, theory, and experiments. ACM TOIT 5(1), 231–297 (2005)
8. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: WWW7, pp. 107–117 (1998)
9. Chakrabarti, S.: Mining the Web: Discovering Knowledge from Hypertext Data. Morgan-Kauffman, San Francisco (2002)
10. Charikar, M., Chaudhuri, S., Motwani, R., Narasayya, V.R.: Towards estimation error guarantees for distinct values. In: PODS, pp. 268–279 (2000)
11. Chien, S., Dwork, C., Kumar, R., Simon, D.R., Sivakumar, D.: Link evolution: Analysis and algorithm. Internet Mathematics 1(3), 277–304 (2004)
12. Durand, M., Flajolet, P.: Loglog counting of large cardinalities (extended abstract). In: Di Battista, G., Zwick, U. (eds.) ESA 2003. LNCS, vol. 2832, pp. 605–617. Springer, Heidelberg (2003)
13. Dwork, C., Kumar, S.R., Naor, M., Sivakumar, D.: Rank aggregation methods for the web. In: WWW, pp. 613–622 (2001)
14. Fagin, R., Kumar, R., Sivakumar, D.: Comparing top k lists. In: SIAM Discrete Algorithms (2003)
15. Flajolet, P., Martin, G.N.: Probabilistic counting algorithms for data base applications. J. Comput. Syst. Sci. 31(2), 182–209 (1985)
16. Jelasity, M., Montresor, A., Babaoglu, Ö.: Gossip-based aggregation in large dynamic networks. ACM Trans. Comput. Syst. 23(3), 219–252 (2005)
17. Kamvar, S., Haveliwala, T., Manning, C., Golub, G.: Exploiting the block structure of the web for computing pagerank. Technical report, Stanford University (2003)
18. Kempe, D., Dobra, A., Gehrke, J.: Gossip-based computation of aggregate information. In: FOCS, Washington, DC, USA, p. 482. IEEE Computer Society, Los Alamitos (2003)
19. Langville, A., Meyer, C.: Updating the stationary vector of an irreducible markov chain with an eye on google's pagerank. In: SIMAX (2005)
20. Langville, A.N., Meyer, C.D.: Deeper inside pagerank. Internet Mathematics 1(3), 335–400 (2004)

21. Lewontin, R., Prout, T.: Estimation of the number of different classes in a population. Biometrics 12(2), 211–233 (1956)
22. Liben-Nowell, D., Balakrishnan, H., Karger, D.R.: Analysis of the evolution of peer-to-peer systems. In: PODC, pp. 233–242 (2002)
23. Ntarmos, N., Triantafillou, P., Weikum, G.: Counting at large: Efficient cardinality estimation in internet-scale data networks. In: ICDE, p. 40 (2006)
24. Pandurangan, G., Raghavan, P., Upfal, E.: Building low-diameter p2p networks. In: FOCS, pp. 492–499 (2001)
25. Parreira, J.X., Castillo, C., Donato, D., Michel, S., Weikum, G.: The juxtaposed approximate pagerank method for robust pagerank approximation in a peer-to-peer web search network. VLDB J. 17(2), 291–313 (2008)
26. Parreira, J.X., Donato, D., Michel, S., Weikum, G.: Efficient and decentralized pagerank approximation in a peer-to-peer web search network. In: VLDB, pp. 415–426 (2006)
27. Ratnasamy, S., Francis, P., Handley, M., Karp, R.M., Shenker, S.: A scalable content-addressable network. In: SIGCOMM, pp. 161–172 (2001)
28. Rowstron, A.I.T., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: IFIP/ACM Middleware, pp. 329–350 (2001)
29. Sankaralingam, K., Yalamanchi, M., Sethumadhavan, S., Browne, J.C.: Pagerank computation and keyword search on distributed systems and p2p networks. J. Grid Comput. 1(3), 291–307 (2003)
30. Shi, S., Yu, J., Yang, G., Wang, D.: Distributed page ranking in structured p2p networks. In: ICPP (2003)
31. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: SIGCOMM, NY, USA, pp. 149–160. ACM Press, New York (2001)
32. Wang, Y., DeWitt, D.J.: Computing pagerank in a distributed internet search system. In: VLDB (2004)
33. Wu, J., Aberer, K.: Using a Layered Markov Model for Distributed Web Ranking Computation. In: ICDCS (2005)

Online Outlier Detection Based on Relative Neighbourhood Dissimilarity

Hoang Vu Nguyen, Vivekanand Gopalkrishnan, and Praneeth Namburi

Nanyang Technological University, 50 Nanyang Avenue, Singapore
ng0001vu@ntu.edu.sg, asvivek@ntu.edu.sg, pran0001@ntu.edu.sg

Abstract. Outlier detection has many practical applications, especially in domains that have scope for abnormal behavior, such as fraud detection, network intrusion detection, medical diagnosis, etc. In this paper, we present a technique for detecting outliers and learning from data in multi-dimensional streams. Since the concept in such streaming data may drift, learning approaches should be online and should adapt quickly. Our technique adapts to new incoming data points, and incrementally maintains the models it builds in order to overcome the effect of concept drift. Through various experimental results on real data sets, our approach is shown to be effective in detecting outliers in data streams as well as in maintaining model accuracy.

1 Introduction

A large part of the web is dynamic, and several applications like customer profiling, fraud detection, event detection, etc., need to learn from such changing data. However, in order to improve the learning process, it is important to accurately and quickly identify outliers in all such cases. Techniques for outlier detection can be broadly classified into supervised and unsupervised approaches. Unsupervised outlier detection can be further classified as distance-based [3, 11] and density-based [4]. However, the accuracy of all these approaches is often reduced due to the phenomenon of *concept drift* [16]. Besides concept drift, model building methods on data streams also have other problems. As mentioned in [9], incremental clustering methods share a common property which is also a drawback: they are *order-dependent*. An approach is *order-independent* if it generates the same result regardless of the order in which data is presented, otherwise it is said to be *order-dependent*. In this work, we aim to detect outliers in a concept drift environment, and design an online outlier detection technique to specifically handle the following issues:

Adaptation to Concept Drift. Concept drift causes the underlying model to become outdated, since new concepts appear in data. The technique therefore should be able to detect these changes and cope with the current trend in data.

Memory Constraints. In a streaming environment, data grow with time and it is impossible to store them. The designed technique should contain mechanisms to extract and store only relevant characteristics of data for learning.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 50–61, 2008.

In order to address this problem, we present an *incremental* online outlier detection approach based on *Relative Neighbour-hood Dissimilarity* (ReND) of data points. This *ReND* framework enables detection and learning from outliers in multi-dimensional data streams. We prove through empirical results that exploiting the information contained in outliers contributes to the process of knowledge discovery. The rest of this paper is organized as follows. Related work is analysed and compared with our *ReND* framework in the next section. Algorithms for our framework are proposed and analyzed in Section 3, and empirical comparisons with other current-best approaches are presented in Section 4. Finally, the paper is summarized in Section 5 with directions for future work.

2 Background and Related Work

Methods of learning concept drift can be classified into two types: *incremental (online) learning* [6] and *batch learning* [10]. Online learning methods are desired by the nature of the problem being addressed here. Recent work describes and evaluates VFDT [6], an *anytime system* that builds decision trees using constant memory and constant time per example to overcome concept drift in data streams. Upon detecting a concept change, VFDT builds a new prediction model from scratch. Our system, on the other hand, maintains a set of clusters implicitly capturing information from historical data and avoids scratch update.

Many supervised approaches to outlier mining first learn a model over a sample already labeled as exception or not [7] and then evaluate a given new input as *normal* or *outlier* depending on how well it fits the model. The main drawbacks of supervised techniques include the requirement of labeled data, and the limited ability in discovering new types of abnormal events. These techniques usually do not address the problem of outdated labels which may occur due to concept drift, and therefore, may not be suitable for the problem in question.

Among existing unsupervised methods, some well-known ones are distance-based and density-based. The concept of distance-based outlier was first introduced by Knorr and Ng [11] and recently Angiulli et al. [3] proposed a new definition of outliers based on the distance of data points to their corresponding k nearest neighbours. Breunig et al. [4] introduce the notion of Local Outlier Factor (LOF) that measures the degree that an object is an outlier with respect to the density of the local neighbourhood. Generally, these approaches can only be applied on static databases and cannot be used to deal with data streams where the possibility of changing the detection model is very high as new data arrives.

StreamEvent [2] updates the detection model using outliers by proposing a method for distinguishing between *primary events* and *secondary events*. However, it does not propose a scheme for maintaining the model of the system. More specifically, *StreamEvent* can be used for detecting outliers but it cannot be used for answering the question *"what is the current classification of normal data points?"*. The *ReND* framework, on the other hand, is applicable *both* for detecting outliers and for handling concept drift. Because *ReND* also organizes

data into clusters representing the classification of normal data points, it can also be used for answering queries about normal data. Recent techniques proposed by Otey et al. [13] and Subramaniam et al. [15] also address the problem of detecting outliers in a data stream environment. Both these approaches are window-based. An inherent problem associated with window-based approaches is that the knowledge from historical data is ignored if the window size is not large enough. Furthermore, the importance of data in learning process does not necessarily depend on the order in which it arrives at the model. However, if the window size is too large to capture the history knowledge, the corresponding method becomes space-consuming. Finding a way to capture the knowledge from the past data in the constraint of limited memory is obviously a much better approach. Online clustering methods [5, 8] aim to maintain the clusters for data streams. These techniques are equipped with mechanisms for updating the clusters whenever new data points arrive. However, they are not mainly designed for detecting outliers. Therefore, the problem they try to tackle is different from ours. Our proposed technique detects outliers as well as maintains the model accuracy.

3 The ReND Framework

Since the *ReND* framework continuously monitors streams, we have multiple data points arriving at any specific time instant. However, to simplify explanation in this work, we assume that only one data point arrives at every instant. This may be achieved, without losing generality, by a simple *discretization* of the points in time and monitoring the stream in ticks. As in other supervised approaches, we use training audit trails as sources of expert experiences to provide a first view of the data streams. We then exploit the audit trails for constructing behavior characteristics of the training datasets, i.e., the initial clusters.

3.1 Definitions

Consider an d-dimensional data stream. At a specific instant of time, the *ReND* framework maintains a set of n clusters $C^T = \{C_1, C_2, \ldots, C_n\}$ that represents knowledge captured from the historical data, from where they may be initially built. A cluster $C_i \in C^T$ $(1 \leq i \leq n)$ contains the following information:

- A solving set $SolvSet(C_i)$ of size L, which contains L data points - $p_1^{C_i}, p_2^{C_i}, \ldots, p_L^{C_i}$ sorted in the descending order of their time stamps. Each data point $p_j^{C_i} \in SolvSet(C_i)$ contains a list of k nearest neighbours and the distances to these neighbours (sorted in ascending order of distance to $p_j^{C_i}$).
- Support $S(C_i)$, which is the number of data points that belong to C_i.
- Centroid $Cent(C_i)$, which is the center of C_i.

The set of k nearest neighbours of a data point p (excluding itself) in a cluster $C \in C^T$ is denoted as kNN_p.

Definition 1. ***Cumulative Neighbourhood Distance*** *denotes the dissimilarity of a point p with respect to its k nearest neighbours, and is defined as the total distance from p to its k nearest neighbours in C. Therefore,*
$Dis_p = \sum_{m \in kNN_p} D(p, m)$

Dis_p captures the degree of dissimilarity of p to its nearest neighbours. As a consequence, the lower Dis_p is, the more similar p is to its neighbours, i.e., higher is the probability that p belongs to same cluster as its neighbours, and vice versa. The mean, μ_p and standard deviation, σ_p of neighbourhood weight density of a data point p that belongs to cluster C are defined as $\mu_p = \frac{\sum_{m \in kNN_p} Dis_m}{k}$ and $\sigma_p = \sqrt{\frac{\sum_{m \in kNN_p} (Dis_m - \mu_p)^2}{k}}$. Intuitively, μ_p and σ_p represent the local distribution of data point p's neighbourhood density in terms of Cumulative Neighbourhood Distance.

Definition 2. *The* ***Relative Outlier Score*** *of a data point p, in terms of the dissimilarity with respect to its k nearest neighbours in a cluster, is defined as* $ROS_p = 1 - \frac{Dis_p}{\mu_p}$

Definition 3. *A data point p of a cluster* $C \in C^T$ *is considered as an* ***Outlier*** *if the absolute value of* ROS_p *is greater than three times the normalized standard deviation of neighbourhood weight density of p. Hence, we have* $ROS_p| > 3\sigma_p/\mu_p$

Here we assume that distribution of points in the cluster is Gaussian. The mechanism for detecting outliers in the $ReND$ framework is deviation-based. The outlier score of a data point p is compared with its prospective neighbours, and the outlier flagging decision is based on the assumed local distribution of p's Cumulative Neighbourhood Distance. This criterion eliminates the dependence on other external parameters (e.g., the number of outliers that should be flagged). We will prove experimentally that this new notion of outlier score and the corresponding flagging mechanism are more efficient and intuitive than LOF [4] for detecting outliers in static databases (c.f. Section 4). Through various tests on streaming data, they are shown to be also applicable in a streaming environment.

3.2 Adapting New Incoming Data Points

For each new incoming data point p, $ReND$ first checks whether p is an outlier or a normal data point based on the available clusters in C^T. In case p is classified as a normal data point, its corresponding cluster will be updated, otherwise, it is stored for future model reconstruction (if any) that is caused by *concept drift*. Consider a new data point p. Assume at the time p arrives, there are n clusters C_i $(1 \leq i \leq n)$ in C^T, and a set C^O of temporary clusters constructed from outliers using a simple incremental clustering technique. The *Monitor_Stream* (Algorithm 1) then processes p, and classifies it to cluster C if the distance from p to C's centroid is the smallest among the available clusters in C^T, i.e. $D(p, Cent(C)) = min_{C_i \in C^T} D(p, Cent(C_i))$. If more than one cluster satisfies

Algorithm 1. MONITOR_STREAM

Input: p, C^T, C^O
Set candidate signatures $CanClusters \Leftarrow \emptyset$
foreach $C_i \in \mathcal{C}^T$ **do**
  if $D(p, Cent(C_i)) = min_{C_j \in C^T} D(p, Cent(C_j))$ **then**
    $CanClusters \Leftarrow CanClusters \cup C_i$
Select C_{near} such that $S(C_{near}) = max_{C_i \in CanClusters} S(C_i)$
Compute Dis_p and ROS_p w.r.t to C_{near}
if $|ROS_p| > 3\sigma_p/\mu_p$ **then**
  Flag p as outlier; Incrementally cluster C^O with p
  if τ *percentages data points during period* M *are outliers* **then**
    call RECONSTRUCT_MODEL
else
  Set $Cent(C_{near}) \Leftarrow \frac{S(C_{near}) \cdot Cent(C_{near}) + p}{S(C_{near})+1}$; Set $S(C_{near}) \Leftarrow S(C_{near}) + 1$
  Set $A \Leftarrow SolvSet(C_{near}) \cup p$; Form $G_1, \ldots, G_{L+1}$ L-subsets of A
  foreach $G \in A$ **do**
    Compute $Cent(G)$
  Select $G_{min} \in A$ s.t. $D(Cent(G_{min}), Cent(C_{near})) =$
  $\min(D(Cent(G), Cent(C_{near}))), \forall G \in A$
  Set $SolvSet(C_{near}) \Leftarrow G_{min}$
  foreach $p_o \in SolvSet(C_{near})$ **do**
    Re-compute kNN_{p_o}

this condition, we choose the cluster with largest support, say C_{near}. The k nearest neighbours of p are then identified from the solving set $SolvSet(C_{near})$. If $|ROS_p| > 3\sigma_p/\mu_p$ (c.f. Definition 3), then we flag p as outlier, and then apply simple incremental clustering for C^O in which each cluster's radius is less than or equal to the maximum radius of the n clusters in C^T. This is done in order to estimate the total number of clusters before carrying out the model reconstruction (if any). Due to memory limitations, each outlier is only kept for M ticks, i.e. its timeout is M, after which it is discarded. If the outlier flagging condition is $false$, we accept p as member of C_{near} which is then updated. To reflect the *concept drift*, we first compute new centroid and update the support, i.e. $Cent(C_{near}) = \frac{S(C_{near}) \cdot Cent(C_{near}) + p}{S(C_{near})+1}$ and $S(C_{near}) = S(C_{near}) + 1$. We then combine p into $SolvSet(C_{near})$. For each L-subset in $SolvSet(C_{near})$, we compute its corresponding centroid. After this process, we choose the subset whose centroid is nearest to $Cent(C_{near})$ to be the new solving set and assign it to $SolvSet(C_{near})$. We then re-calculate the list of nearest neighbours for each data point in $SolvSet(C_{near})$.

3.3 Model Reconstruction

If the current model is unable to accurately identify the characteristics of new data points, we perform model reconstruction (Algorithm 2). This is triggered by checking if the condition in Definition 3 is *true* for a user-defined τ percentage data points during period M. Since this is an offline step, the current model is

Algorithm 2. RECONSTRUCT_MODEL

Input: p, C^T, C^O, Partitional clustering technique T_C
1 Prune C^T ; Compute number of new clusters n_C
2 Apply T_C with number of clusters $K = n_C$
3 **foreach** *created cluster* C_{new} **do**
4 ⊢ Call $Create\_Solver(C_{new}, 0.2 \cdot S(C_{new}), k)$
5 Assign the final clusters to C^T ; Set $C^O \Leftarrow \emptyset$

still kept for detection, and is subsequently replaced by the new model. This step consists of two phases: re-clustering and cluster construction.

Re-Clustering: Let the number of temporary clusters in C^O be n_o. Let us denote $\mu_C = \frac{\sum_{i=1}^{n} S(C_i)}{n}$ and $\sigma_C = \sqrt{\frac{\sum_{i=1}^{n} [S(C_i) - \mu_C]^2}{n}}$, which are the mean and standard deviation of cluster support in C^T, respectively. We then prune clusters from C^T by removing each cluster $C_i \in C^T$ if $|S(C_i) - \mu_C| > 3\sigma_C$. Assume that after this process, there are n' clusters left. The set of data points in these n' clusters and n_o temporary clusters will be combined for the clustering process. To identify the number of clusters for this clustering process, we apply the method proposed by Can et al. [5]. We perform a grid *discretization* of the data, where each attribute of the data is divided into R *equiwidth* ranges in which $R = 2$ if $(n' + n_o) \leq d$ and $\lceil (n' + n_o)/d \rceil$ otherwise. Therefore, in total we have $d \cdot R$ ranges of all dimensions. We denote them as r_i $(1 \leq i \leq d \cdot R)$. For each data point p, we form a vector of length $d \cdot R$. If the projection of p on the i^{th} dimension lies in range r_j (r_j is a range of the i^{th} dimension), then the j^{th} position of p's vector has value 1, and 0 otherwise. We then combine all the created vectors to form a matrix Z of size $N \cdot (d \cdot R)$, where N is the total number of data points left. The total number of clusters, say n_C, is then identified from matrix Z. According to [5], we have $n_C \leq d \cdot R$, i.e., $n_C = O(n' + n_o)$. Therefore, $n_C = O(n + M)$, since $n_o = O(M)$ and $n' \leq n$. Since in practice, M is chosen to be $\leq 2L$ and $\geq L$, we can conclude that $n_C = O(n + L)$. A partitional clustering algorithm, e.g. K-means, is then employed with K chosen as the estimated n_C. After the clustering process, we obtain K clusters C'_i $(1 \leq i \leq K)$, from where clusters with low support are pruned (applying the same pruning procedure as shown above). If any cluster C'_i has less than $2L$ data points, we sample without replacement from the current members of C'_i.

Cluster Construction: With these created clusters, we apply *Create_Solver* (Algorithm 3), to obtain the solving set for each cluster as in [3]. This algorithm takes three inputs: cluster (C), number of data points (n) whose Cumulative Neighbourhood Distances are largest in C, and number of nearest neighbours (k). n is chosen to be 20% of the total size of C. $Insert(S, s, p)$ inserts a data point p into a reverse-sorted list S if the Cumulative Neighbourhood Distance of p is larger than that of the s^{th} item. After the above process, the size of $SolvSet(C)$ is maintained at L, by sampling (with or without replacement as

Algorithm 3. CREATE_SOLVER

```
                                    Input: C, n, k
1   Set SolvSet(C) ⇐ ∅, Top ⇐ ∅, CandSet ⇐ k random data points of C, c ⇐ 0
2                                   while CandSet ≠ ∅ do
3       Set SolvSet(C) ⇐ SolvSet(C) ∪ CandSet, C ⇐ C\CandSet
4                                   foreach p ∈ CandSet do
5       foreach q ∈ CandSet do
6                                       if p ≠ q then
7       Update(kNN_p, q), Update(kNN_q, p)
8                                   Set NextSet ⇐ ∅
9                                   foreach p ∈ C do
10      foreach q ∈ CandSet do
11                                      Set dis ⇐ max(Dis_p, Dis_q)
12                                      if dis ≥ c and p ≠ q then
13      Update(kNN_p, q), Update(kNN_q, p)
14                                  Insert(NextSet, k, p)
15                                  foreach p ∈ CandSet do
16      Insert(Top, n, p)
17                                  if |Top| = n then
18      Set c ⇐ Min(Top)
19                                  Set X ⇐ ∅
20                                  foreach p ∈ NextSet do
21                                  if Dis_p ≥ c and |X| < m/2 then
22      Set X ⇐ X ∪ p
23                                  Set Y ⇐ ∅
24                                  foreach p ∈ (C\NextSet) do
25                                  if Dis_p ≥ c and |Y| < m/2 then
26      Set Y ⇐ Y ∪ p
27                                  Set CandSet ⇐ X ∪ Y
```

needed) from itself. Finally, we assign the set of obtained clusters to C^T, and empty C^O.

3.4 Time Complexity and Comments on Parameters Used

In this section, we present a brief discussion about time complexity of the *ReND* framework. The following analysis is based on n, the number of clusters of C^T, which may be changed upon reconstruction. We also discuss inherent drawbacks in dealing with data streams.

Time complexity of Monitor_Stream: The operations of *Monitor_Stream* involve a) choosing a cluster nearest to the new data point p, b) computing p's k nearest neighbours, c) computing Cumulative Neighbourhood Distance and *ROS* for p, and d) either performing incremental clustering if p is identified as an outlier, or updating the nearest cluster. Therefore, the time complexity of *Monitor_Stream* can be stated as $O(d \cdot n) + O(k \cdot d \cdot L) + (O(d \cdot L) + O(k \cdot d)) + max(O(d \cdot M), O(d \cdot L^2)) = O(d \cdot n + k \cdot d \cdot L + d \cdot L^2))$. This simplifies to $O(d \cdot (n + L^2))$, since practically, we have $L \leq M \leq 2L$ and $L/2 \leq k \leq L$. Also,

since the number of clusters n is negligible compared to the number of points L in the solving set, the online processing time is quadratic with respect to L. However, since the size of the solving set, L is user-controlled and fixed during the outlier detection process, we can achieve very high efficiency (c.f. Table 3).

Time complexity of Reconstruct_Model: The process of reconstructing the model consists of a) estimating n_C, b) clustering with $K = n_C$ clusters, and c) constructing the solving set for each cluster. Consequently, the time complexity of $Reconstruct\_Model = O((n \cdot L + M) \cdot (n + M)) + O(d \cdot (n \cdot L + M) \cdot (n + M)) + O(d \cdot (n + M) \cdot L^2)$, which simplifies to $O(d \cdot (n + L) \cdot L^2)$. As before, since L is fixed and n is small compared to L, the reconstruction complexity is determined by L, and we observe that the process is very efficient.

Inherent Drawbacks in Dealing with Data Streams: Data streams, by nature are problematic for incremental clustering methods which are *order-dependent* [9]. The $ReND$ framework organizes normal data concepts into clusters such that each cluster represents a concept, and assigns each new incoming data point to an appropriate cluster (concept). Even though it is affected by *order-dependence*, $ReND$ reduces its effect by *batching* data through the use of time interval M before carrying out any major updates of the system model. Thus, major changes will not occur until we get enough information indicating that the current model is outdated. Batching reduces the effect of data order on the model accuracy by reducing unnecessary updates while processing the streams.

4 Empirical Results and Analysis

In order to evaluate the approaches, we need datasets with normal points and outliers. We used publicly available datasets from the UCI repository [1] for this purpose, and demonstrate the:

- Efficiency of our new notion of outlier score (ROS) in detecting outliers in *static* databases, compared with LOF [4] and feature bagging [12].
- *Online* detection power of the $ReND$ framework in comparison with $StreamEvent$ [2], accuracy and effectiveness of $ReND$'s *reconstruction* mechanism, in comparison with $StreamKM$ [8], a representative online technique.

The datasets are presented in Table 1. The first two are obtained from the web: Syskill & Webert Web Page Ratings (SysWeb) and Anonymous Microsoft Web Data (MicWeb). SysWeb contains data in four categories, with 50 most informative words extracted for each category. A boolean feature vector representing each record in a category is formed by indicating whether or not a particular informative word is present in that record. A similar procedure has been introduced in [14]. Since all categories have non-overlapping set of features, to unify them, we extend the number of features of each category to 200, assigning 0 to the extended features. By doing so, we have a dataset with 200 Boolean attributes and 4 classes. For MicWeb, we merge the training and testing sets

together, and then cluster the complete set into 5 classes. The other datasets have been popularly used by the outlier detection community, and for brevity, the reader is kindly referred to [12] for their detailed descriptions. We also note that for the following experiment, K-means clustering is used as the clustering technique for the $ReND$ framework during the model reconstruction process.

Table 1. Experiment setting $\mathcal{E}_1$ and AUC values

Dataset	Outlier class v/s. Normal	Static Detection			Online Detection	
		ROS	*LOF*	Feature Bagging	*ReND*	*StreamEvent*
SysWeb	each class v/s. rest	**0.774**	0.631	0.712	**0.891**	0.878
MicWeb	each class v/s. rest	**0.817**	0.742	0.783	**0.952**	0.944
Ann-thyroid	class 1 v/s. 3	**0.920**	0.869	0.869	**0.984**	0.950
Ann-thyroid	class 2 v/s. 3	**0.769**	0.761	**0.769**	**0.931**	0.890
Iris	each class v/s. rest	0.990	**1.000**	0.990	**0.990**	**0.990**
Letter	each class v/s. rest	**0.849**	0.816	0.820	**0.887**	0.850
Lymphograhpy	merged class 2 & 4 v/s. rest	**0.980**	0.924	0.967	**0.981**	0.976
Satimage	smallest class v/s. rest	**0.701**	0.510	0.546	**0.738**	0.667
Segment	each class v/s. rest	**0.883**	0.820	0.845	**0.945**	0.890
Shuttle	class 2, 3, 5, 6, 7 v/s. 1	**0.851**	0.825	0.839	**0.998**	0.961

Offline Detection Power: This experiment verifies the effectiveness of our outlier score, ROS for detecting outliers. Since there is no clear intuition of what is an *outlier* in these datasets, a class conversion procedure as used in [7, 12] was deployed to convert the data classes to binary sets (*normal* or *outlier*). The average AUC of all tests is then computed, by first constructing the ROC (Receiver Operating Characteristic) curve [3] for each dataset, and then choosing the corresponding area under the ROC curve as AUC. Larger the value of AUC, better is the detection quality. Here, we compare ROS against LOF [4] and feature bagging [12], which have been used to detect outliers in static data. Experimental settings and the corresponding results (AUC under *StaticDetection*) are shown in Table 1. We observe that ROS performs better than other techniques, except on the Iris dataset where LOF performs slightly better than ROS. This implies that ROS is competitive in detecting outliers in *static* datasets.

Online Detection Power: In order to evaluate the utility of ROS in data streams, we took approximately 10% data points from each *normal* class and combined them with those from *outlier* classes for testing. This simulates the streaming effect, by introducing outliers gradually (and not suddenly) in the detection phase. Since this experiment aims only to test the online detection power (and not throughput), we set the timeout M to infinity here. Also, the solving set, L, is chosen to be 30% of the maximum cluster size and the value of k is varied from $L/2$ to L. Our approach is compared against *StreamEvent* [2], which is also designed for detecting deviations online, and their average AUC results are shown in Table 1 (under *OnlineDetection*). These results show that the $ReND$ framework performs better than *StreamEvent* in all cases, except on the Iris dataset, where both approaches are similar.

Table 2. Experiment setting $\mathcal{E}_2$: False Alarm Rate and SSQ statistics

Datasets	**Model** (data points)	**Test** (data points)	**False Alarm Rate**		**SSQ**	
			ReND	*StreamEvent*	*ReND*	*StreamKM*
SysWeb	Two arbitrary classes	Remaining classes	0.077	0.29	189.11	320.52
MicWeb	Two arbitrary classes	Remaining classes	0.12	0.34	18573.22	30226.14
Iris	Two arbitrary classes	Remaining classes	0.174	0.42	30.46	52.58
Letter	Two arbitrary classes	Remaining class	0.133	0.23	3552.43	5015.25
Segment	Six arbitrary classes	Remaining classes	0.086	0.27	9180.80	11614.01
Shuttle	Class 4 and 5	Class 1 (part)	0.054	0.35	52852.60	67564.55

Construction Power of ReND: This experiment illustrates the importance of capturing both normal and abnormal data in the learning phase, and shows the ability of our method in constructing new clusters. Here we split the datasets differently from the previous case, because we would like to observe adaptability of the model to frequently changing concepts. For example, for the Shuttle dataset, since the total supports of class 2, 3, 6 and 7 are negligible compared to class 4 and 5, we only use class 4 and 5 to construct the initial model. For each round of tests, we choose randomly 20-30% data points of class 1. For $ReND$, we fixed the value of L to be 30% of the maximum cluster size and varied the values of M (within the range $[L, 2L]$) and k (within the range $[L/2, L]$) to obtain different values of *false alarm rate*. $ReND$ is compared with $StreamEvent$ using the average false alarm rate for each update process. This setup procedure and results are shown in Table 2. Since $StreamEvent$ does not capture normal data, it is unable to detect changes in the concept, and hence performs poorly compared to our method. This particularly happens because $StreamEvent$ continuously executes its $LearnStream$ method, causing a significant delay in processing the stream, and leading to a higher false alarm rate.

Quality of Created Clusters: Now we evaluate the quality of clusters created during the detection process, using the same setting as in the previous experiment. Since online clustering techniques also address cluster maintenance, we compare against a recent representative - $StreamKM$ [8]. $StreamKM$ processes a stream in chunks, and maintains a set of K centroids to characterize the previous chunks by recursive clustering. Since K-means clustering technique, is employed in both $StreamKM$ and the $ReND$ framework, we choose the criterion function [9] as SSQ which is the total sum of squared distances from each data point to its assigned cluster's centroid. This metric reflects the quality of all resultant clusters in a specific dataset, and smaller the value of SSQ, better is the clustering algorithm. Preprocessing for our approach is similar to the previous experiment. For $StreamKM$, during the training phase, we extract K centroids, where K is chosen to be the number of classes of each testing dataset, e.g. $K = 3$ for the Iris dataset. $StreamKM$ then iterates to find the best centroids. During the testing phase, we divide the testing dataset into S_D/S_C chunks where S_D is the size of the dataset and S_C is the size of a chunk. We ran $StreamKM$

Table 3. Processing times for $ReND$ and $StreamKM$ and $\max(n_r)$

Datasets	t_p (ms)	t_C (ms)	n_r
SysWeb	1.36	14.51	infinity
MicWeb	1.72	17.68	infinity
Iris	0.14	1.44	infinity
Letter	0.71	0.30	1490
Segment	0.54	8.89	infinity
Shuttle	0.83	0.10	1223

with three different values of chunk size (5, 8 and 10). Small chunk sizes were chosen since we wanted to compare the throughput of both methods in a fast streaming environment. SSQ for both approaches is taken as the average of all corresponding runs, and is shown in Table 2. We observe that $StreamKM$ performs worse than the $ReND$ framework in all cases. The result may be better if the chunk size is increased, however, that would significantly reduce throughput of the technique. So we conclude that although the $ReND$ framework processes streams in an incremental manner, its resultant cluster quality is still high.

Throughput Comparison: Let the time taken by the $ReND$ framework to process a data point be t_p (milliseconds), and let t_C (milliseconds) be the time taken by $StreamKM$ to process a chunk of size S_C. Table 3 shows the average values of t_p and t_C from the above experiments. Let us also denote the number of data points arriving per millisecond into the system as n_r. The throughput of the $ReND$ framework is $1/t_p$. Since $StreamKM$ first waits for a chunk of data to arrive and then processes the chunk, the total time taken by $StreamKM$ to process a chunk includes waiting time and chunk processing time, i.e., $StreamKM$'s processing time $= S_C/n_r$ (waiting time) $+ t_C$ (processing time). So the throughput of $StreamKM = S_C/(S_C/n_r + t_C)$. We can deduce that the throughput of the $ReND$ framework is greater than that of $StreamKM$ iff $t_p - 1/n_r < t_C/S_C$. From the values of t_p and t_C obtained by the experiments, we can derive the maximum n_r that satisfies the above inequality. This tells us the maximum number of data points that can arrive in the system such that the throughput of the $ReND$ framework is better than that of $StreamKM$. From Table 3, we observe that this value is quite large (it reaches infinity in some cases). Hence we conclude that for practical scenario, the $ReND$ framework is very efficient, and can be applied for fast data streams with high quality.

5 Conclusions

This work contributes to outlier detection research by proposing a new notion of outlier score - ROS and a model that handles changes in concepts using the outliers themselves. Our detection mechanism also eliminates the dependence on other input parameters. Experimentally, we have proven that ROS outperforms LOF [4] for outlier detection in static databases. We also developed a framework that adapts to drifting concepts in multi-dimensional data streams. This

ReND framework can learn from new data and carries out necessary updates to maintain the model accuracy. Experimentally, we have also shown that *ReND* is very suitable for monitoring fast and high dimensional streams. We're currently developing a sampling-based optimization approach to determine the optimal values of nearest neighbours and timeout.

In other current work, we are examining the problem of carrying out online reconstruction. This requires us to maintain a certain balance between adaptation speed and model accuracy. In particular, we try to extract necessary information from data and perform model reconstruction partly based on the data obtained, i.e., reducing delay in the knowledge discovery process.

References

[1] UCI machine learning repository, `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[2] Aggarwal, C.C.: On abnormality detection in spuriously populated data streams. In: SDM (2005)

[3] Angiulli, F., Basta, S., Pizzuti, C.: Distance-based detection and prediction of outliers. IEEE Transactions on Knowledge and Data Engineering 18(2), 145–160 (2006)

[4] Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J.: LOF: Identifying density-based local outliers. In: SIGMOD Conference, pp. 93–104 (2000)

[5] Can, F.: Incremental clustering for dynamic information processing. ACM Transactions on Information Systems 11(2), 143–164 (1993)

[6] Domingos, P., Hulten, G.: Mining high-speed data streams. In: KDD, pp. 71–80 (2000)

[7] Fawcett, T., Provost, F.J.: Adaptive fraud detection. Data Mining and Knowledge Discovery 1(3), 291–316 (1997)

[8] Guha, S., Meyerson, A., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams: Theory and practice. IEEE Transactions on Knowledge and Data Engineering 15(3), 515–528 (2003)

[9] Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys 31(3), 264–323 (1999)

[10] Klinkenberg, R.: Learning drifting concepts: Example selection vs. example weighting. Intelligent Data Analysis 8(3), 281–300 (2004)

[11] Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: VLDB, pp. 392–403 (1998)

[12] Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: KDD, pp. 157–166 (2005)

[13] Otey, M.E., Ghoting, A., Parthasarathy, S.: Fast distributed outlier detection in mixed-attribute data sets. Data Mining and Knowledge Discovery 12(2-3), 203–228 (2006)

[14] Pazzani, M., Muramatsu, J., Billsus, D.: Syskill and webert: Identifying interesting web sites. In: AAAI, pp. 54–61 (1996)

[15] Subramaniam, S., Palpanas, T., Papadopoulos, D., Kalogeraki, V., Gunopulos, D.: Online outlier detection in sensor data using non-parametric models. In: VLDB, pp. 187–198 (2006)

[16] Tsymbal, A.: The problem of concept drift: Definitions and related work. Technical Report TCD-CS-2004-15, Department of Computer Science, Trinity College Dublin, Ireland (2004)

Behavioural Targeting in On-Line Advertising: An Empirical Study

Joanna Jaworska[1] and Marcin Sydow[2]

[1] Gemius SA, Wołoska 7, 02-675 Warszawa, Poland
joanna.jaworska@gemius.pl
[2] Web Mining Lab,
Polish-Japanese Institute of Information Technology,
Koszykowa 86, 02-008, Warszawa, Poland*
msyd@pjwstk.edu.pl

Abstract. On-line behavioural targeting is a dynamically evolving area of web mining concerning the applications of data analysis of on-line users' behaviour and machine learning in optimising web on-line advertising and constitutes a problem of high importance and complexity.

The paper reports on experimental work concerning testing various benchmark machine-learning algorithms and attribute preprocessing techniques in the context of behavioural targeting.

Our final goal is to build a system which automatically learns and subsequently decides which on-line advertisements to present to a user visiting a web page, based on his previous recorded behaviour, in order to maximise the revenue of the ad-network and the web site hosting the ad, and, at the same time, to minimise the user's annoyance caused by potentially inappropriate advertisements.

We present a general adaptive model which makes it possible to test various machine-learning algorithms in a plug-in mode and its implementation.

We also report our experimental work concerning comparison of the performance of various machine-learning algorithms and data preprocessing techniques on a real dataset.

The performance of our experiments is evaluated by some objective metrics such as the *click-through rate* (CTR) or related.

Our experimental results clearly indicate that the presented adaptive system can significantly increase the CTR metric by 40% for some settings.

All the experiments are performed on a real industrial dataset concerning on-line ads emitted in the Polish Web during a 1-month period in 2007.

Up to the authors' best knowledge this is the first and largest evaluation made on this kind of real industrial data in Poland, at the time of writing.

Keywords: behavioural targeting, on-line advertising, machine learning, experimentation, click-through rate, user profile.

* The authors are listed in the alphabetical order.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 62–76, 2008.

1 Introduction

The Web today is economically driven by on-line advertising. The total revenues on the Internet advertising in USA amounts to an impressive figure of over $21B in 2007[1] with the incredible annual growth rate being above 20% for the recent few years. There are many forms of the Internet advertising but the top-3 largest shares accumulate about 70% of the total revenues and belong to the following categories:[2]

- *search-based advertising* (40%)
- *display ads* (22%)
- *classifieds* (17%)

In all of the above forms of on-line advertising the revenues of the advertisers significantly depend on the degree of match between the presented ad and the context in which a user is visiting the web page. The better match the higher probability that the user is interested in the ad and subsequently undertakes some actions (as clicking the ad or even making a transaction) which are the short-term goals of the ad campaign. The problem of such a matching is called *targeting* and is a well known concept in marketing. In one of the simpler forms, the users could be divided into some groups with regard to some measured or known attributes (like web usage profile, age, etc.) and each user group is presented with different group of ads, which is called *consumer segmentation* [5].

In this paper, we interchangeably use the terms *advertisement* (or just *ad*, for short) and *creative*.

1.1 Behavioural Targeting

The goal of a behavioural targeting system is to automatically decide which creatives are most relevant to be presented to a web user visiting a web page based on the previous recorded on-line behaviour of that user. For example, a user who is exceptionally frequently visiting sport-related web pages could be presented more sport-oriented ads than others, etc. Behavioural targeting is recently in the centre of interest in on-line advertising (e.g. [12]) and has a great potential in improving the performance of ad-serving systems. There are also some recent ideas about the potential usage of the user data collected by Internet service providers to better fit the ads presented to the users [4].

Technically, when a user visits a web page with a set of potential advertisement *placements*, an ad-serving system has to decide which out of many creatives and on which placements to present to that user. Such a decision can be based on, among others, various factors concerning the previous web behaviour of that user. The exemplary factors of interest could be:

- what pages or page categories have been previously visited by the user (page-category based user profile)

[1] IAB annual report: `http://www.iab.net/insights_research/`

[2] In the other recent years the proportions were similar, according to the previous IAB annual reports.

- which creatives have been previously clicked by the user or users that are similar in some way (ad-based user profile)
- what are the temporal patterns of that user
- what placements are usually clicked by that user, etc.

The list is not exhaustive and one should notice, that the 'behavioural' factors listed above can be easily combined with many other, so called 'contextual' factors such as the content of a web page, link structure of the visited site, current popularity of some keywords present in the URL or page title, etc.

1.2 Privacy Issues

Besides its great practical potential in on-line advertising, behavioural targeting also raises some important issues concerning e.g. preserving user privacy [9]. Namely, all the large-scale search engines, which are the major players in on-line advertising, collect lots of data concerning on-line users in order to better serve them, for example, to better match ads to particular users. However, such data could be also potentially used to break users' privacy which constitutes a very important issue [13].

Quite recently, a popular social-networking portal Facebook[3] introduced new features which make it possible for the advertisers to connect the social-based information collected by the company running the portal to be used for behavioural targeting on third-party sites which raised a heated dispute concerning preserving users' privacy [1].

The experiments reported in this paper, do not constitute any serious threat on users' privacy, since the users are represented by cookies and the only information concerning them which is analysed concerns the distribution of the visited page categories (see the sections 2 and 3).

1.3 Contributions

The contributions of this paper are as follows:

- we present an experimental framework for testing and evaluating various settings concerning behavioural targeting
- we propose a general adaptive behavioural targeting model and its exemplary instantiation which is based purely on a simple category-based user profiles
- we report experimental results which demonstrate that the model is generally successful in practice, despite its current simplicity
- a preliminary comparison of a couple of classification algorithms and attribute-preprocessing techniques is made and reported
- the evaluation is made on unique, large industrial datasets. Up to our best knowledge this is one of the first reported evaluations made on real datasets of this kind and scale
- we propose various extensions to the model and settings to be studied in further work

[3] `www.facebook.com`

1.4 Outline of the Paper

In section 2, we propose a general model for behavioural targeting. We describe the experimental design, framework and the datasets in the section 3, as well as the techniques used to transform the data, the machine-learning algorithms used in the experiments and also the evaluation metrics. In section 4, we present our experimental results and discussion. In section 5, we report some previous work related to ours. In section 6, we conclude and propose a plan for potential future work and experiments. In the last section 6, we list the members of the Gemius company team, who significantly contributed to this paper by designing the model and implementing the experiments.

2 The General Model

We take the machine-learning approach to the behavioural targeting, i.e. we consider the system which can automatically *learn* how to best match ads to a user given her previous on-line behaviour.

Each user is identified by a cookie and represented by a set of attributes U. In our current setting, we limit the user attributes merely to her *page-category profile*. More precisely, we consider 13 different web page categories as follows: (1) business, finance, law, (2) immovable, construction, (3) education, (4) information, publicity, media, (5) culture and entertainment, (6) automobiles, (7) new technologies, (8) job, (9) social issues, (10) sport, (11) life style, (12) touristic, (13) other. In our experiments, we took into account and classified about 400 web sites. In most cases the classification was at the domain level and only in 3 cases we classified at the sub-domain level, due to the sizes of the sites. The classification of the web sites was done by a team of human experts at Gemius.

Each user is represented by 13 attributes corresponding to the categories mentioned above. In the raw form, each attribute simply counts the number of times a user visited a page in the corresponding category during the profile estimation period. The user profiles were aggregated during the two months preceding the month in which the ads used in our experiments were emitted.

After each visit of a user to a web page, the element of the user profile U that corresponds to the category of the visited web site was increased by 1.

Profiles were built for all users that visited some chosen web sites in the given three-month period. A number of 24,684,183 profiles was gathered.

Although the ads chosen in the experiments were displayed to part of them we built profiles for all users in order to have data ready for future experiments. An average user visits only 2 or 3 categories of web sites so that the user profiles are quite sparse.

A couple of transformations of the user profile attributes were considered in our experiments (see subsection 3.2 for more details).

In the initial phase of the ad campaign, for some creative c, the system collects the training examples in order to learn the relationship between the user's profile U and her response to the presented ad (clicked or not).

In the current stage of our research, for simplicity, we focus only on the 'behavioural' aspect of the user-ad matching problem so that we do not take into account any other dimensions of information such as the placement of the ad or temporal attributes nor any kind of the contextual information such as the contents of the page.

As the outcome of the learning phase, for a given creative c, the system produces a model which can be represented by a function $f_c(U) = p \in [0, 1]$, where U denotes a user profile. The value $f_c(U)$ can be interpreted as the potential relevance of the ad c presented to the user described by the profile U. After the training phase is completed, the learnt function f_c can be applied to support the decision whether to present the ad c to a user visiting the page and characterised by her profile U. We assume, that the ad is *presented* to the user whenever $f_c(U) > \theta_c$, for some threshold θ_c which can be tuned experimentally.

The model is created for the purposes of experiments based on real historical data. We use this model to analyse the impressions of separate ads. Obviously, in real industrial setting, usually a *set* of ads should be *ranked* in order to chose the subset of them to be presented to the visitor. Thus, our current model is merely a simplification, since it considers only a *single* ad at each visit. This simplification is partially due to the limited dataset we had at disposal for our research at the time of performing the experiments.

In our opinion, the presented model, though being a simplification of a real model, as stated above, is still very valuable for research purposes as it makes it possible to test various:

- attributes characterising a user
- machine-learning algorithms used in the training phase
- data preprocessing techniques applied to the attributes, etc.

As we suspect, the selection and preprocessing techniques of the attributes representing the user profile as well as the choice of the learning algorithm may have remarkable impact on the final performance of the system. One of the final goals of the research presented here is to examine this issue.

Various settings mentioned above can be evaluated and compared with each other by some common performance measures such as click-through rate (CTR) or others, which are related (see section 3.4).

The CTR value is a very common characteristic used in on-line advertising and denotes the ratio between the number of times the ad was clicked by a user and the number of times the ad was presented on a web page (impressions). For example, CTR of 3% means that the ad was clicked in 3% of the cases it was presented to the web users.

Since the *CPC model* (cost per click) is currently very popular for charging on-line advertisers for their ads, one of the main goals of a good on-line ad-serving system is to select the ads to be presented to users on web pages so that their CTR values are maximised. In other words, the higher CTR of the presented ad, the higher revenue of the ad-serving system and, at the same time, the more efficient ad campaign (since the number of clicks could be regarded as a measure of user interest in the advertised product, services, etc.)

3 Design of Experiments

The experiments concerning our behavioural targeting model were based on a database concerning archived impressions of a couple of creatives presented to a set of users together with the recorded information specifying which impressions ended up with a click. Thus, we applied a supervised-learning approach to build a binary classifier which, given a fixed ad and an input user profile U, predicts if the ad is likely to be clicked and consequently should be *selected* as being worth to be presented to the user or not.

According to our learning scheme, the set of ad impressions was divided into two subsets: the training subset and the testing subset. In most cases we took the first 10% (chronologically) impressions as the training set and the rest of the impressions as the testing one, however this division is also a subject to experimentation (see subsection 4.2).

In the training phase, we experimented with various machine-learning algorithms (see subsection 3.3) and attribute-transformation techniques (see subsection 3.2).

3.1 Datasets and Experimental Framework

The data used in our experiments comes from real impressions of ads during a one-month period in 2007, on the Polish web pages, which are monitored by the Gemius company.[4] Each experiment concerned a sequence of impressions of a single creative on a single placement. In our experiments we analysed the data granted by the Gemius company for our research. The data represented about 1.5 million ad impressions in total concerning 7 randomly chosen ads. The additional criterion of the ad selection done by the Gemius company was the number of the impressions and clicks so that the appropriate amount of data is available for our statistical analysis.

The data is summarised in the Table 1. The value of *clicking users* denotes the number of users who clicked at least once on any ad presented by the system. Thus, as one can observe, most of the users who clicked on any ad did it only once during the examined period. Another remark on the data is that the 7 creatives, which were available for the experiments could be divided into 2 groups, with regard to the higher (ads 1-4) or lower (ads 5-7) CTR value.

The measurements concerning the user activities, ad impressions and clicks, were collected by the special scripts put in the creatives. The users, by visiting the pages with the monitored ads, trigger the network connections to the measurement servers called *hit-collectors*. Subsequently, the user-driven events are recorded, preprocessed and compressed to be sent to the computing centre where they are stored in a database for off-line analyses[5]

[4] `http://www.gemius.com`

[5] These are the elements of the technology framework created and exploited by the Gemius company for its operations concerning Internet research. The mission of the Gemius company is to deliver professional knowledge concerning the Internet market and research, as well as analytical and consulting services concerning the Internet.

Table 1. Datasets

	creative 1	creative 2	creative 3	creative 4	creative 5	creative 6	creative 7
impressions	508896	101022	76548	302787	150647	158869	158373
users	500286	66959	76240	237107	147569	123735	123333
clicks	28859	1671	1990	12447	687	755	817
clicking users	28817	1648	1990	12160	685	729	788
CTR	5.76%	2.46%	2.60%	3.74%	0.65%	0.48%	0.53%

The computation of the user profiles was based on the analysis of the Polish Web traffic which was prior to the ad impressions under examination and was done by the Gemius company.

The input data for each experiment consisted of a sequence of records representing ad impressions. Each record was of the following structure:

- *ts* - time stamp of the impression
- *cookieID* - unique user id
- *placementID* - unique placement id
- *creativeID* - unique creative id
- *click* - a variable encoding whether the impression ended up with a click (1) or not (0)
- *SetOfAttributesDescribingCookie* - a set of attributes representing the recorded previous behaviour of a user. In our current setting we limit this set to 13 category-based counters described previously

3.2 Data Preprocessing

The collected raw data concerning each user is in the form of a vector $\{u\}_i$, where for any i u_i is the counter of visits of the user to a web page belonging to the category i. Besides using the data in the raw form, we also experimented with various functions to transform the attribute values $\{u\}_i$ representing the user. The techniques are described below. The abbreviated names in brackets refer to the names of attributes present in the tables in the section 4.

- *raw* (LB) – unprocessed numerical value of the attribute
- *norminf* (NI) – normalisation: $u_i/\sqrt{\sum_i u_i^2}$
- *normmax* (NM) – normalisation: $u_i/max_i u_i$
- *logarithm* (LN) – $ln(u_i)$
- *binarisation* (BIN) – the signum function $sgn(u_i)$

3.3 Machine-Learning Algorithms

In the learning phase, we tested multiple well known machine-learning algorithms or approaches such as:

- decision trees (J48[6], RandTree, REPTree)

[6] J48 is the well-known C4.5 decision tree algorithm implemented in Java in the Weka software [2].

- tree-based methods (Ridor, PART, JRip)
- naive Bayes (NB)
- discriminant analysis (DA)

The abbreviations in brackets refer to the tables presented in the Section 4. In our experiments, we used custom implementations as well as the implementations available in the Weka software toolkit. We refer the interested reader to the Weka documentation [2] for the detailed descriptions of the algorithms listed above.

3.4 Evaluation Metrics

The evaluation of the experiments is as follows.

Let $E = \{e\}_i$ denote the sequence (indexed by a set of indexes I) of the recorded impressions of a creative on some placement belonging to the *testing set*. Let C denote the subset of I corresponding to all the impressions for which the creative was actually *clicked* (accordingly to the archived information) and S denote the subset of I corresponding to all the impressions which were *selected* by our system to show the creative as being assessed likely to be clicked. Let $n = |I|$. We use the following metrics.

- original *Click-through rate*: $ctr = |C|/n$
- *Precision*: $prec = |S \cap C|/|S|$
- *Result*: $result = prec/ctr$
- *Cover*: $cover = |S|/n$
- *Recall*: $|S \cap C|/|C| = cover \cdot result$

The names 'Precision' and 'Recall' are borrowed from Information Retrieval, due to their natural correspondence to these concepts.[7]

One of the main goals of our approach is to maximise the *result* metric, which is equivalent to maximising *prec* since *ctr* is constant in our setting, as being representing the historically achieved CTR for the particular ad. Obviously, any values of the *result* metric which are above the value of 1 are considered as being successful. For example, $result = 1.2$ means that the application of the ad selecting system increased the *effective CTR* of the ad by 20%.

One should bare in mind, however, that the values of *cover* or *recall* should also be taken into account in overall evaluation. This is important since their values being close to the value of 0 mean that we almost completely block the presentation of the ad under consideration, which actually reduces the revenues, independently on the value of the precision or CTR achieved in this situation.

[7] It is important to remind, that in real systems, maximising *prec* is in some opposition to maximising *recall*, which is a commonly known fact in the field of Information Retrieval [3].

4 Experimental Results

In this section, we report our experimental results. We experimented with the impressions of ads which were summarised in the Table 1. We compared algorithms, attribute preprocessing techniques and experimented with the training/testing set size ratio. The experiments were conducted in two series, which are presented as follows.

4.1 Comparison of Various Algorithms and Attribute Transformations

The first series of experiments concerned comparison of performance of various machine-learning algorithms and attribute transformation functions. The results of different settings for all ads that were available for our experiments are presented in the Tables 2, 3, 4 and 5, for each ad separately. The abbreviated names for algorithms and attribute transformation functions were defined in the sections 3.3 and 3.2. In all the experiments, the sample is balanced with regard to the number of 'clicked' and 'unclicked' ad impressions.

Table 2. Results - creative 1

creative 1				creative 1, cont.			
algorithm	data	$\frac{prec}{CTR}$	cover	algorithm	data	$\frac{prec}{CTR}$	cover
J48	LB	1.240	0.402	REPTree	LB	1.226	0.536
J48	NI	1.410	0.324	REPTree	NI	1.262	0.510
J48	NM	1.254	0.539	PART	LB	1.191	0.680
J48	LN	1.012	0.902	PART	NI	1.268	0.523
J48	LN, NI	1.013	0.884	NB	LB	1.003	0.967
Ridor	LB	1.158	0.769	NB	NI	1.008	0.928
Ridor	NI	1.335	0.414	JRip	LB	1.232	0.537
RandTree	LB	1.224	0.526	JRip	NI	1.251	0.530
RandTree	NI	1.234	0.540				

Table 3. Results - creative 2 [left part] and creative 3 [right part]

creative 2				creative 3			
algorithm	data	$\frac{prec}{CTR}$	cover	algorithm	data	$\frac{prec}{CTR}$	cover
J48	LB	1.0791	0.3878	J48	LB	1.1238	0.2931
J48	NI	1.0603	0.4791	J48	NI	1.1147	0.3990
J48	NM	1.0333	0.4830	J48	NM	1.1321	0.4398
Ridor	LB	1.0402	0.1858	Ridor	LB	1.0682	0.7243
Ridor	NI	1.0255	0.1623	Ridor	NI	1.2025	0.3557
RandTree	LB	1.1124	0.3796	RandTree	LB	1.0599	0.4863
RandTree	NI	1.0457	0.4714	RandTree	NI	1.0198	0.4199
RandTree	NM	0.9675	0.4830	RandTree	NM	1.0130	0.4354

Table 4. Results - creative 4 [left part] and creative 5 [right part]

creative 4				creative 5			
algorithm	data	$\frac{prec}{CTR}$	cover	algorithm	data	$\frac{prec}{CTR}$	cover
J48	LB	1.4707	0.3969	J48	LB	1.2180	0.6166
J48	NI	1.3999	0.4265	J48	NI	1.0182	0.7102
J48	NM	1.5353	0.3455	J48	NM	1.0196	0.6166
Ridor	LB	1.0947	0.7814	Ridor	LB	1.1952	0.5709
Ridor	NI	1.0822	0.7525	Ridor	NI	0.9831	0.9015
RandTree	LB	1.1966	0.4326	RandTree	LB	1.3555	0.4132
RandTree	NI	1.0150	0.5306	RandTree	NI	0.8598	0.6289
RandTree	NM	1.1777	0.4751	RandTree	NM	1.0661	0.8373

Table 5. Results - creative 6 [left part] and creative 7 [right part]

creative 6				creative 7			
algorithm	data	$\frac{prec}{CTR}$	cover	algorithm	data	$\frac{prec}{CTR}$	cover
J48	LB	1.8876	0.2478	J48	LB	1.1963	0.4296
J48	NI	1.5690	0.3093	J48	NI	1.2677	0.5047
J48	NM	1.1763	0.4632	J48	NM	1.1925	0.4354
Ridor	LB	1.1601	0.5723	Ridor	LB	1.0751	0.7441
Ridor	NI	1.3025	0.4529	Ridor	NI	1.2932	0.2725
RandTree	LB	1.0886	0.4978	RandTree	LB	1.1733	0.5215
RandTree	NI	1.0816	0.4218	RandTree	NI	1.0612	0.5467
RandTree	NM	1.2108	0.3984	RandTree	NM	1.2004	0.4634

The first observation is that the results can be generally considered as *successful* since the achieved $result = prec/CTR$ values are greater than 1 for the vast majority of the cases. In other words, applying the prediction model learnt by the system *increased* the CTR, for all the analysed cases (except a tiny fraction), despite the fact that we use only a simple category-based user profiles in learning.

4.2 The Choice of the Training Sample

In the second series of experiments, we aimed at observing the impact of the choice of the training set on the system performance (see Table 6). We tested three different settings:

1. $10\%all - 1 - smp0$ - the first 10% (chronologically) of the ad impressions, sampled so that the number of clicked impressions was exactly 50% (balanced)
2. $10\%all$ - the first 10% (chronologically) of the ad impressions
3. $20\%all$ - the first 20% (chronologically) of the ad impressions

All the computations concerned with our experimentation were performed at the Gemius company and the numerical results were subsequently reported to the authors.

Table 6. Results of the choice of the training set - creative 1

algorithm	data	training set	# train	% train	0	1	$\frac{prec}{CTR}$	cover
DA	BIN	10% all-1-smp0	6 009	1.1%	48.6%	51.4%	1.15	57.1 %
DA	BIN	10% all	52286	10.0%	94.1%	5.9%	1.28	54.9 %
DA	BIN	20% all	104572	20.0%	93.6%	6.4%	1.30	54.1 %
DA	LB	10% all-1-smp0	6008	1.1%	48.6%	51.4%	1.02	98.9%
DA	LB	10% all	52286	10.0%	94.1%	5.9%	1.02	98.9%

4.3 Discussion

When analysing the above results, reported separately for each ad in the tables 2, 3, 4 and 5, it is hard to find any clear relationship between the classification algorithm or data preprocessing technique applied and the performance. More experimentation and perhaps on a larger number of ads is necessary to draw any conclusions concerning the influence of the applied algorithms and attribute preprocessing techniques on the system performance. The authors plan to continue the research as more data is available for analysis.

The most important observation, however, is, that the applied model of adaptive behavioural targeting seems to be generally successful since it increases the effective CTR in almost all the tested cases. Importantly, for many cases it was possible to increase the CTR by over 20%, and in some cases even by over 40%.

The last experiment, in which various choices of the training set were compared (Table 6) shows that, for a fixed learning algorithm, the choice has some impact on the performance. The class balancing technique does not seem to indicate significant improvement. Making the training set twice bigger did not improve the results significantly. On the other hand, the binarisation technique seems to significantly improve the results. Definitely, as in the first series of the experiments, much more experimentation is needed to draw more reliable conclusions.

There is a great variance of the results across different ads i.e. for some ads it seems to be much 'easier' to build a model than for others, with regard to

Table 7. Results - the values are averaged in 3 groups of creatives: all the creatives (1-7), high-CTR creatives (1-4), and low-CTR creatives (5-7)

		All creatives (1-7)		High-CTR creatives		Low-CTR creatives	
algorithm	data	$prec/CTR$	cover	$prec/CTR$	cover	$prec/CTR$	cover
J48	LB	1.3165	0.3963	1.2284	0.3700	1.4339	0.4314
J48	NI	1.2626	0.4504	1.2460	0.4072	1.2849	0.5080
J48	NM	1.1919	0.4746	1.2387	0.4518	1.1295	0.5051
Ridor	LB	1.1132	0.6212	1.0904	0.6152	1.1435	0.6291
Ridor	NI	1.1746	0.4730	1.1610	0.4211	1.1929	0.5423
RandTree	LB	1.1730	0.4653	1.1484	0.4562	1.2058	0.4775
RandTree	NI	1.0452	0.5083	1.0786	0.4902	1.0008	0.5325
RandTree	NM	1.1059	0.5154	1.0527	0.4645	1.1591	0.5664

all the tested algorithms and data preprocessing techniques. For example, the experiments concerning the creatives 4 and 6 achieved observably higher evaluation metrics for all the tested settings compared with the other examined creatives. On the other hand, for a fixed creative, the achieved results vary in the range of even 80%, but usually are in the range of 20-40%. Definitely, more ads available for experimentation would help greatly to eliminate the noise.

In order to partially alleviate the problem of too low size of our ad sample we grouped ads according to their original CTR into two groups: high-CTR ads (ads 1-4) and low-CTR ads (ads 5-7) and averaged the results in these groups, as well as averaged the results achieved for all the ads. The averaged figures are reported in the Table 7. Now, after averaging it is possible to make some preliminary observations, which are the following:

- the J48 algorithm seems to achieve consistently better performance than other studied algorithms, for the examined sample[8]
- the results seem to be slightly better for the ads with the lower (original) CTR value
- the preprocessing technique seems to have an observable impact on the performance, however it depends on the classification algorithm used. Somehow surprisingly, the NI normalisation preprocessing technique seems to help only for the Ridor algorithm, while the raw-number LB mode works best for the majority of cases.

The averaged results encourage to continue the experimentation on larger sample of ads as soon as such data is available, to clarify the preliminary conclusions made above.

5 Related Work

Behavioural targeting is a quite new topic and, as such, is not very well represented in the scientific literature yet. Some previous work, however, concerns issues that are related to our work, to some extent.

A model which applies genetic programming approach to automatically learn which ads to present on a given placement is presented in [8], and reports over 60% increase in precision over some other systems. However, it concerns the content analysis of the page rather than behaviour of the web users.

An interesting model for matching the most relevant information to an on-line user was recently proposed in [7], though no experimental results are reported in the paper. [10] concerns the problem of predicting a web page category a user might be interested in, however one of the main differences to our work is that the model is based on keywords describing each category. A demographic prediction based on user's behaviour is considered in [6]. In [11] the problem of predicting CTR is studied in the context of search-based ads.

[8] This fact is not very surprising since it is repeatedly reported as a very powerful algorithm in many practical applications.

In general, the methods presented in our paper can be applied independently to the methods described in the papers mentioned above.

6 Conclusions and Further Work

We presented a general framework for experimenting with various settings in machine-learning approach to behavioural targeting. We also proposed an adaptive model and illustrated that our framework is useful for experimental work. We also reported preliminary results of experiments conducted on a real industrial dataset.

A simple instantiation of the general model which takes into account only the category-based profile of users was experimentally studied. The results obtained on real datasets seem to indicate that the model is quite successful (it generally increases the CTR value), despite its simplicity.

At the current stage of our research there are no clear conclusions yet, stating which algorithms or preprocessing techniques work definitely better than others.

However, after averaging the results over groups of ads with regard to their original CTR value, some preliminary observations may be made. Namely, the J48 algorithm seems to work best, and our approach seems to work better for those ads which have lower original CTR value. Surprisingly, we did not observe any significant improvement of performance by normalising the user profiles (NI or NM preprocessing modes) in the current settings. Obviously, the experiments should be continued on larger ad samples. In case, we have access to more data in future, we plan to repeat experiments to obtain more statistically reliable results.

Another issue related to the limitations of the particular dataset we had at our disposal is the fact that the current model is concerned with deciding whether to present only a single ad, which is an obvious simplification of the real situation, where a *subset* of a set of multiple ads should be automatically chosen to be presented on a web page upon a user visit. In future, we plan to extend the model to take into account multiple ad candidates for a single user-page pair, given that we have suitable data available for such research.

In our opinion, the results obtained so far, are very promising and encourage to continue experimentation with more sophisticated models or other algorithms to further improve the performance of the system.

In particular, it seems interesting to experiment with the following settings in the future work:

- introducing the temporal dimension to our model, in particular, to apply time series analysis techniques to build the model
- combining the model with content-based approach
- additional category-based attributes specifying the times spent on each of the categories, with possible division into work-days and week-days, for example
- different choice of categories

– clustering users or creatives
– different choices of the training set

It is also possible to introduce a two-fold user profile, for short-term and long-term behaviour, to better handle some temporally local behaviour.

In the current setting, we define a clear separation between the learning phase and usage of the system. However, we plan to extend the model such that it endlessly adapts to the users and their behaviour, which seems to be a particularly adequate model in the context of a real, on-line ad-serving system.

We feel that the experimental measurements should be repeated on larger and less biased datasets to increase the reliability. Such measurements were not available to the authors in the time of writing but are planned to be made as the part of continuation of the work presented here.

Acknowledgements

The framework was developed at the Gemius SA company with the significant contribution of Piotr Ejdys, the president of the company, who also designed the model. The experiments were implemented and executed by Mariusz Gądarowski, Eliza Bujnowska and Małgorzata Półtorak, Gemius SA. Also the real data was kindly granted for the research purposes by the Gemius company.

We used the Weka [2] software for some of the machine-learning tasks.

References

1. `http://www.facebook.com/business/?beacon` (accessed February 19, 2008)
2. Weka software toolkit, `http://www.cs.waikato.ac.nz/ml/weka/`
3. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press / Addison-Wesley (1999)
4. Dyson, E.: Wall street journal (February 11, 2008), `http://online.wsj.com`
5. Higgs, B., Ringer, A.C.: Trends in consumer segmentation (2007) (accessed February 19, 2008), `http://www.anzmac07.otago.ac.nz/anzmacCD/papers/Higgs1.pdf`
6. Hu, J., Zeng, H.-J., Li, H., Niu, C., Chen, Z.: Demographic prediction based on user's browsing behavior. In: WWW 2007: Proceedings of the 16th international conference on World Wide Web, pp. 151–160. ACM Press, New York (2007)
7. Hubeman, B.A., Wu, F.: Economics of attention: maximizing user value in information-rich environments (August 2007)
8. Lacerda, A., Cristo, M., Gonçalves, M.A., Fan, W., Ziviani, N., Ribeiro-Neto, B.: Learning to advertise. In: SIGIR 2006: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 549–556. ACM Press, New York (2006)
9. Leenes, R.E.: Do you know me? decomposing identifiability. Tilburg University Legal Studies Working Paper No. 001/2008 (January 2008)
10. Ng, V., Mok, K.-H.: An intelligent agent for web advertisements. In: CODAS 2001: Proceedings of the Third International Symposium on Cooperative Database Systems for Advanced Applications, Washington, DC, USA, p. 102. IEEE Computer Society, Los Alamitos (2001)

11. Richardson, M., Dominowska, E., Ragno, R.: Predicting clicks: estimating the click-through rate for new ads. In: WWW 2007: Proceedings of the 16th international conference on World Wide Web, pp. 521–530. ACM Press, New York (2007)
12. Smith, S.: Behavioral targeting could change the game (January 23, 2007) (accessed February 19, 2008)
http://www.econtentmag.com/Articles/ArticleReader.aspx?ArticleID=18964
13. Tene, O.: What google knows? privacy and internet search engines (October 2007)

Integrating Multiple Data Sources for Stock Prediction

Di Wu[1], Gabriel Pui Cheong Fung[2], Jeffrey Xu Yu[1], and Zheng Liu[1]

[1] Dept. of Sys. Eng. & Eng. Mgt., The Chinese University of Hong Kong
{dwu,yu,zliu}@se.cuhk.edu.hk
[2] School of Info. Tech. & Elec. Eng., The University of Queensland
g.fung@uq.edu.au

Abstract. In many real world applications, decisions are usually made by collecting and judging information from multiple different data sources. Let us take the stock market as an example. We never make our decision based on just one single piece of advice, but always rely on a collection of information, such as the stock price movements, exchange volumes, market index, as well as the information from the news articles, expert comments and special announcements (e.g., the increase of stamp duty). Yet, modeling the stock market is difficult because: (1) The process related to market states (up and down) is a stochastic process, which is hard to capture by using the deterministic approach; and (2) The market state is invisible but will be influenced by the visible market information, such as stock prices and news articles. In this paper, we try to model the stock market process by using a Non-homogeneous Hidden Markov Model (NHMM) which takes multiple sources of information into account when making a future prediction. Our model contains three major elements: (1) External event, which denotes the events happening within the stock market (e.g., the drop of US interest rate); (2) Observed market state, which denotes the current market status (e.g. the rise in the stock price); and (3) Hidden market state, which conceptually exists but is invisible to the market participants. Specifically, we model the external events by using the information contained in the news articles, and model the observed market state by using the historical stock prices. Base on these two pieces of observable information and the previous hidden market state, we aim to identify the current hidden market state, so as to predict the immediate market movement. Extensive experiments were conducted to evaluate our work. The encouraging results indicate that our proposed approach is practically sound and effective.

1 Introduction

In the traditional time series analysis, predicting the future movements of a time series is usually based solely on its historical performance. Yet, many factors that are not explicitly lying on the time-series, but are appearing in some other data sources, may have significant impacts on the time series behavior. A typical example of such kind of time series is the stock prices. The fluctuation of stock prices is the consequence of the actions taken by the investors, where their actions, although occasionally irrational, are predominantly understandable and rational [1].

In order to enhance the prediction of the stock price movements, we claim that we should not only focus on the historical performance of a particular stock only, but should

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 77–89, 2008.

Table 1. Notations

Sym.	Description	Sym.	Description
O	an observed market state sequence	S	a hidden market state sequence
o_i	the ith segment in O	s_i	the ith state in S
X	an external event state sequence	$\mathcal{F}$	a set of features
x_i	the ith state of X	f_j	a feature in $\mathcal{F}$
$I(f_{ij},d_k)$	the instant impact of f_{ij} in day d_k	F_j	feature state sequence of f_j
$I(f_j,d_k)$	the impact factor of f_j in day d_k	f_{ij}	the ith bursty state in F_j

also understand the actions of the investors. In this paper, we take news articles as an important data source that would affect the investors' actions, which in-turn will affect the stock prices. By analyzing the news contents, we try to understand investors' thinkings. In fact, previous researches have already shown that there is a strong relationship between stock price movements and the news article contents [11,4,12,18,16,5]. Similar to these researches, we use news articles as an important data source for assisting us to predict the stock price movements. Unlike these researches, we do not only consider the impacts of the news articles, but also take the historical and current movements of the stock prices into account. That is, we consider multiple information sources before making a future prediction.

The major challenges here are: (1) How to formulate a model by combining all the four data sources (market states, historical prices, current prices and news articles impacts) together ? (2) How to dynamically define the time frames of the market states? (3) How to extract the information presented in the news articles? Our system is built based on the assumption of Non-homogeneous Markov Model (NHMM). With the help of NHMM, we aim to identify the hidden market states (up/down) according to the observed stock price changes and the contents of the news articles, so that we can make a future prediction.

The rest of the paper is organized as follows: Section 2 presents our proposed work; Section 3 evaluates our model; Section 4 briefly discusses the related work; Section 5 summarizes and concludes this paper.

2 Proposed Work

2.1 An Overview of the NHMM and the Stock Market

Table 1 shows a list of symbols that would be used throughout this paper. Let $T = \{t_0, t_1, \ldots, t_n\}$ be a sequence of consecutive time segments. Throughout this paper, for simplicity, we assume that the stock market has two states only: up (price rises) and down (price drops).[1] Let $O = \{o_0, o_1, \ldots, o_n\}$ be a sequence containing the observed market states, where $o_i = 0$ if the stock price is dropping at time t_i and $o_i = 1$ otherwise. Let $S = \{s_0, s_1, \ldots, s_n\}$ be a sequence containing the hidden market states, where $s_i = 0$ if the hidden market state is down and $s_i = 1$ otherwise. It is worth noting that the observed market state, o_i, and the hidden market state, s_i, are different. For instance,

[1] Yet, our model can definitely be applied in multiple states.

given a particular stock, even though its current price is rising, it may be in a down state indicating that it will experience a significant drop in the near future. The instant rising can simply be viewed as noise or outliers. Logically, every state may possibly be influenced by some external events. Let $X = \{x_0, x_1, \ldots, x_n\}$ be a sequence of external event states, where x_i will affect the status of s_i.

In the stock market, it is widely recognized that S follows a stochastic process [13], i.e., S is a discrete Markov chain. Under this assumption, the state s_i depends only on its previous state s_{i-1} and its external event state, x_i. It is independent on any other states or O. This memoryless characteristic of S is also known as the Efficient Market Hypothesis [2].

In this paper, we use the information from the most up-to-date news articles to model the current external event state, x_i, and use the historical stock prices to model the previous observed market state, o_{i-1}. Based on x_i and o_{i-1} and the previous hidden market state, s_{i-1}, we try to identify the current hidden market state, s_i, hoping to predict the next immediate observed market state, o_i. In the following sections, we will discuss how we model this situation, as well as how the observed market state sequence and the external event state sequence are identified.

2.2 Generation of the Non-homogeneous Hidden Markov Model

In our Non-homogeneous Hidden Markov Model, since the ith observed market state, o_i, only depends on the ith hidden market state, s_i, and independent on the previous observed states, o_{i-1}, or any of the external event states, x_m for $m \leq i$, the following equation is formulated:

$$P(o_i|\{s_1, s_2, \ldots s_i\}, o_{i-1}, \{x_1, x_2, \ldots, x_i\}) = P(o_i|s_i) \tag{1}$$

Furthermore, since the ith hidden market state, s_i, depends only on its previous hidden market state, s_{i-1}, and the external event, x_i, thus the following equation is formulated:

$$P(s_i|\{s_1, s_2, \ldots s_{i-1}\}, \{x_1, x_2, \ldots, x_i\}) = \begin{cases} P(s_i|s_{i-1}, x_i) & i \geq 2, \\ P(s_1|x_1) & i = 1. \end{cases} \tag{2}$$

In summary, Eq. (1) formalizes the generation process that the observed state only depends on the immediate hidden state, and Eq. (2) addresses the memoryless characteristic of the discrete state sequence, S. Based on these two equations, the components that are unknown and have to be computed are $P(s_i|s_{i-1}, x_i)$ and $P(o_i|s_i)$.

For $P(s_i|s_{i-1}, x_i)$, it is rather easy to be obtained by employing the multinomial logistic regression [8] to parameterize the hidden state transition $P(s_i|s_{i-1}, x_i)$:

$$P(s_i = n|s_{i-1} = m, x_i) = \frac{\exp(\sigma_{mn} + \rho_n x_i)}{\sum_{k=1}^{K} \exp(\sigma_{mk} + \rho_k x_i)} \tag{3}$$

$$P(s_1 = n|x_1) = \frac{\exp(\lambda_n + \rho_n x_1)}{\sum_{k=1}^{K} \exp(\lambda_k + \rho_k x_1)} \tag{4}$$

where, K is the number of hidden Markov states, σ is the baseline Markov state transition matrix, ρ is the corresponding weight vector for each x_i, and λ is the constant

first-state probability vector. The denominator in the summation is used for normalization, such that the resulting values of both equations are within 0 and 1. With the parameterized equations for the hidden Markov state transition probability, we can estimate the parameters λ_i, σ_{ji} and ρ_i by using the classical Expectation-Maximization algorithm [10]. In the following sections, we focus on how to identify $P(o_i|s_i)$.

2.3 Observed Market State Sequence Identification

In the stock market, the two most important and interesting elements are undoubtedly: (1) The slope of the trend generated by the most recent stock prices; and (2) The duration of the trend. As such, we characterize each observed market state, o_i, by two elements: (1) Duration of the observed market state at o_i, l_i, and (2) Slope of the trend generated by the stock prices at o_i, θ_i.

Without loss of generality, let us assume that we have identified both l_i and θ_i (we will discuss how we compute them in the following sections). Since the slope θ_i and duration l_i are independent, we have:

$$P(o_i|s_i) = P(\theta_i|s_i)P(l_i|s_i). \tag{5}$$

In Eq. (5), $P(\theta_i|s_i)$ and $P(l_i|s_i)$ can readily be computed if we have already identified all θ_i and l_i, and hence our modeling part would be done. Since θ_i is the slope of the trend generated by the stock prices in the observed market state o_i, it can be computed easily by simply applying some existing functions (e.g., linear regression) to approximate the original stock prices in o_i. Eventually, our question reduced to how to identify the duration of the observed market state, or l_i.

Intuitively, the problem can be solved easily by applying the traditional time series segmentation approaches [9,17]: Given a sequence of stock prices $\{y_1, y_2, ...\}$, we try to extend the current market state from y_1 to y_i, such that at y_{i+1} the function that approximates the stock prices would exceed a predefined threshold. Then we say that y_i is the end point of the current market state, and the sequence from y_1 to y_i will be transformed into a segment. Then, the algorithm continues to search for a new potential segment starting from y_{i+1}.

Although the traditional time series segmentation approach is able to identify different segments from the time series, it cannot capture the information from the other data sources, such as the information obtained from the news articles. Recall that in our NHMM, the information from the news articles may trigger the transition of hidden market state, which will in-turn affect the immediate observed market state. Hence, we claim that in order to determine the end points of the market states, we should not only focus on the behaviors of the observed market states, but also need to pay attention to the external events (information contained in the news articles). Accordingly, we propose a novel multiple-source segmentation algorithm to achieve our target.

In the proposed multiple-source segmentation algorithm, it determines whether a point y_i in the sequence $\{y_0, y_1, \ldots, \}$ is an end-point of a market state if either of the following two conditions holds:

1. The approximation error of $\{y_1, y_2, \ldots, y_{i+1}\}$ by a given function (e.g. linear regression) is larger than a predefined maximum error threshold. This criteria is the same as the traditional time series segmentation.

2. The similarity between the external events at y_1 and y_i is smaller than a predefined minimum similarity threshold.

For the first condition, it obviously holds. We do not attempt to have any further elaboration on it. For the second condition, its underlying idea is as follows: if the market information at y_i (e.g. information contained in the news articles) is significantly different from the information that already appeared in the market in the past, then it properly implies that some new and important information is flooding into the market now, and the market would usually adjust itself (the stock prices change) according to the newly raised information. In the following section, we account for how we extract the information from the news articles automatically in order to formulate a sequence of external events, X, so that we can use it to assist us to identify the time frame of market states.

2.4 Information Extraction from News Articles

Given a set of news articles, our objective is to extract some useful information in them, such that the information will trigger the stock price changes. Some researchers [3,4,5,12] align the contents of the whole news articles to the stock prices movements, and presume that the whole set of news articles will trigger the stock price changes. We claim that this approach may lead to many noises, as a piece of news article may contain both supportive and unfavorable information. Other researchers [16,18] use expert knowledge to make the judgment. They predefined a set of "important terms" such that if these terms appear in the news articles, then these news articles are assumed to trigger some predefined effects. While this expert knowledge approach can minimize the noise (i.e maximize the precision), it may ignore many domain specific terms or newly appeared terms, therefore reduce the recall of the model. Although we are in favor of the later approach in the sense that we should index some useful terms rather than take all words in the news articles, we do not want to predefine any terms in advance. Instead, we want to extract the useful terms there automatically and dynamically.

We observed that the emergence of an important event is usually captured by a "burst of words": some words suddenly appear frequently when an event emerges, and their frequencies drop when the event fades away. Hence, by monitoring the temporal changes of the word distributions within the news articles, it should be possible to determine whether there is any new event appearing. In this paper, we call these important words as features. . Specifically, if a feature suddenly appears frequently in a bounded time interval (e.g., two days), we say that an important event emerges. Our problem is thus reduced to how the phase"suddenly appearing frequently" is defined.

Let $\mathcal{F}$ be a set of important features and $f_j \in \mathcal{F}$ is a feature in $\mathcal{F}$. In order to determine which of the intervals the feature f "suddenly appears frequently", we try to compute the "probability of bursty" for each day, d_i. Let $P(d_i, f_j; p)$ be the probability that the feature f_j is bursty in day d_i according to p, where p is the probability that f_j would appear in any arbitrary day given that it is not bursty. The intuition is that if the frequency of f_j appearing in d_i is significantly higher than the expected

probability p, then d_i is regarded as a bursty period. We compute $P(d_i, f_j; p)$ according to the cumulative distribution function of binomial distribution:

$$P(d_i, f_j; p) = \sum_{l=1}^{n_{f_j}} P(l; N_i, p) \tag{6}$$

$$P(l; N_i, p) = \binom{N_i}{l} p^l (1-p)^{N_i - l} \tag{7}$$

where N_i is the total number of words appearing in day d_i, and n_{f_j} is the frequency of f_j appearing in d_i. $P(d_i, f_j; p)$ is a cumulative distribution function of the binomial distribution, and $P(l; N_i, p)$ is the corresponding probability mass function.

Therefore, $P(d_i, f_j; p)$ represents the bursty rate of f_j in day d_i. The bursty periods of f_j are identified by setting a specific threshold such that $P(d_i, f_j; p)$ is larger than the threshold. Finally, a bursty state sequence $F_j = \{f_{0j}, f_{1j}, \ldots, f_{nj}\}$ for feature f_j is formed, where $f_{ij} = 1$ if f_j is bursty (suddenly appear frequently) in day d_i, and $f_{ij} = 0$ otherwise.

Eventually, given a feature state sequence, F_j, of feature f_j, we need to identify its impact on the stock prices. In this paper, we try to quantity this impact factor. Broadly speaking, we observe that there are two kinds of information in the market: (1) The information is simple and its impact is trivial, so that the market can digest the information efficiently and immediately (i.e. the stock price fluctuates as soon as the information appears); and (2) The information is complex and its impact is uncertain, so that the market takes time to consume this piece of information (i.e., the stock price fluctuates with a little bit delay with respect to the time that the information arises). Yet, no matter in which of the situations, we further notice that the impact of the information on the stock prices usually decreases as time goes by and eventually fades away. Eventually, based on these observations, Eq. (8) and Eq. (9) are formulated as follows:

$$I(f_{ij}, d_k) = \begin{cases} f_{ij} e^{-\sigma(k-i)} & \text{if } i < k \\ f_{ij} & \text{if } i = k \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$$I(f_j, d_k) = \sum_{i=1}^{|F_j|} I(f_{ij}, d_k), \tag{9}$$

where $I(f_{ij}, d_k)$ is the immediate impact factor of the feature f_j in day d_k; $|F_j|$ is the number of states in F_j; σ is the fading intensity of the impact factor. In Eq.(8), an exponential decay of the impact factor is assumed in this paper. This is based on our empirical findings and extensive testings. Finally, the cumulative impact factor of the feature f_j in day d_k is computed by summing up the immediate impact factor of f_j in d_k and all of its descendant impacts.

Recall that the reason why we need to extract the information from the news articles is that we want to use it to model the external events in our NHMM model, which will in-turn influence the time series segmentation (see Section 2.3). According to Eq. (6) and Eq. (7), we extract the bursty (useful) information from the news articles. Eq. (8) and Eq. (9) are then used to measure how important the extracted information is with

respect to the time dimension. Eventually, the external event state, x_i, at day d_i is formulated according to the impact factors as follows:

$$x_i = \{I(f_1, d_i), I(f_2, d_i), \ldots, I(f_{|\mathcal{F}|}, d_i)\}. \quad (10)$$

Given two external event states, x_i and x_{i+n} where $n > 0$, the similarity of these two external event states can be measured by using any of the existing similarity measures, such as Cosine Coefficient, Jaccard Coefficient or χ^2[14].

Without loss of generality, let $S(x_i, x_{i+n})$ be the similarity of two external event states, x_i and x_{i+n}. Furthermore, assume that x_i is the beginning of a stock trend and there is a predetermined threshold δ. If there exist an n such that $S(x_i, x_{i+n}) < \delta$, then we say that the market states in day d_i and d_{i+n} are different, i.e. a change in the market state occurs at d_{i+n}. The reason is that the information raised in day d_i and day d_{i+n} are very different as the the similarity of the external events is less than a given threshold, δ. As discussed at the end of Section 2.3, if such situation happens, we would segment the stock prices at $(i+n)$, such that the stock prices from i to $(i+n-1)$ belong to a specific trend type, and $(i+n)$ is the beginning of another trend type .

3 Model Evaluation

We evaluate our proposed approach by conducting a market simulation by using two different measurements: (1) The accuracy of the predictions, and (2) The cumulative profits/loss. Four approaches are tested for comparison:

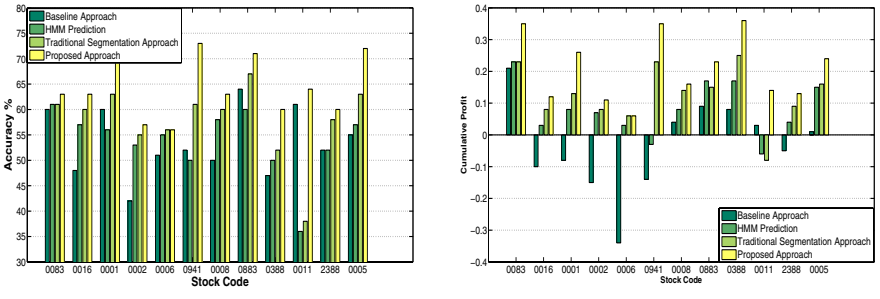
1. **Baseline Approach:** The observed market states are determined by using the traditional time series segmentation approach only. The prediction of the future movement is simply the line joining the last stock price in the previous observed market state and the first stock price of the current market state.
2. **HMM Prediction:** It is the same as Baseline Approach, except that the prediction of the future movement is based on HMM only, without using any of the news articles.
3. **Traditional Segmentation Approach:** It is the same as the Baseline Approach, except that the prediction of the future movement is the mean slope of the predicted hidden state by NHMM.
4. **Proposed Approach**: It is the approach described in this paper.

The stock prices (opening and closing) are archived from the Hong Kong Exchange Market from 1/1/2005 to 12/31/2006, and the corresponding news articles are downloaded from The Standard[2]. Data from 1/1/2005 to 5/31/2006 are used for training, and the remaining are used for evaluation. There are thousands of stocks in the market. We selected the stocks as follows. (1) They are Hang Seng Index (HSI) stocks, because HSI is a market indicator. (2) The stocks are selected from the four main sectors: Properties, Utilities, Commerce & Industry, and Finance. (3) For each of the sectors, we chose the stocks with the largest fluctuation, the largest exchange volume, the highest percentage

[2] http://www.thestandard.com.hk

Table 2. Selected Stocks(Code)

Sectors	Stock (Highest rise)	Stock (Highest drop)	Stock (Largest fluctuation)	Stock (Largest volume)
Properties	0083	0016	0083	0001
Utilities	0002	0006	0002	0006
Commerce	0941	0008	0883	0941
Finance	0388	0011	2388	0005



(a) Prediction Accuracy (b) The rate of return (12 stocks)

Fig. 1. Market simulation results

of rise, and the highest percentage of drop. Accordingly, we selected 12 stocks from the market. They are summarized in Table 2. For the news articles, we identified a total of 928 bursty features. Then, we apply information gain[15] to measure the relationship between the bursty features and the stock prices movement and select M important features for each specific stock.

For the non-homogeneous hidden Markov model (NHMM), unless otherwise specified, we use the following parameters: (1) The number of hidden states, K, is 6; (2) The minimum similarity among events, δ, is 0.01; (3) The influence fading rate, σ is 1; (3) The minimum length of a segment is 3 days; (4) The number of selected features for each stock, M, is 50.

3.1 Prediction Accuracy

The accuracy of the prediction is obtained by checking whether the direction of the predicted trend is the same as the actual real-life trend. For instance, if the prediction for the upcoming trend is "Up" and the upcoming trend is really rising, then we say that the prediction is correct; otherwise, if the prediction is "Down", but the upcoming trend is rising, then we say that the prediction is wrong.

Fig. 1 (a) summarizes the prediction results of the four different approaches on the 12 selected stocks. At a first glance, among the four different approaches, our proposed approach always performs the best, and it outperforms the Baseline Approach significantly. There are four stocks (0001, 0941, 0883 and 0005) whose accuracy obtained by

our approach is higher than 70%. Among these four stocks, three of them come from the largest exchange volume from their corresponding sectors (0001, 0941 and 0005). This suggests that our proposed approach is the most effective when it is applied to the stocks which have the largest amount of transactions. The prediction accuracy of 0006 by using the Traditional Segmentation Approach is the same as that by using our proposed approach. The major reason is that there are extremely few news articles that are related to 0006 directly. As a result, the predictions made by both approaches would eventually be the same, as we do not have any news articles to assist our predictions. The result from HMM Approach further supports our argument of using news articles to assist us with the prediction making. Although it performs better than the Baseline Approach, and sometimes the Traditional Segmentation Approach, it always performs worse than our proposed approach .

To summarize, our proposed approach is most effective when there are many news articles related to the stock (e.g. Commerce and Finance Sector) and may perform similar to the existing approaches for the stocks with few news articles (e.g. Utilities Sector).

3.2 Profits Generated

We evaluate the profits generated by different approaches using a Buy-and-Hold test, where zero transaction cost is assumed [6]:

1. If the prediction of the upcoming trend is positive, then shares of that stock are purchased. If a profit of 1% or more could be made within this detected segment, then all shares are sold immediately; otherwise they are sold at the beginning of the next segment.
2. If the prediction of the upcoming trend is negative, then shares of that stock are sold for short. If the trading price is dropped 1% or more than the shorted price within this detected segment, then shares of that stock are purchased immediately; otherwise they are purchased at the beginning of next detected segment.

Figure 1 (b) shows the cumulative profit measured by the rate of return of the 12 selected stocks. In the previous section, one may argue that the prediction accuracy among different approaches is insignificant. Yet, in terms of the profit generated, the improvement is noticeable . For example, the prediction accuracy between the Traditional Segmentation Approach and our proposed approach on the stock 0001 has only 2% differences. However, the difference of the profit generated between these two approaches is 20%.

Let us take the stock 0005 HSBC (Hong Kong and Shanghai Banking Corporation) for further investigation. Figure 2 shows the predictions made by The Traditional Segmentation Approach (middle) and Our proposed approach (bottom) from 10/12/2006 to 11/20/2006. The dotted lines are the real stock prices, the red lines are the observed market states, and the blue arrows are the predictions made by the corresponding approaches. For the first two predictions (points A and C), both approaches make the predictions at the same time. However, our proposed approach make the third prediction at point E, whereas the Traditional Segmentation Approach makes the third prediction at point F. Although both approaches make all the predictions correctly, the timings of making the predictions are different, where less profit is gained (high loss is incurred)

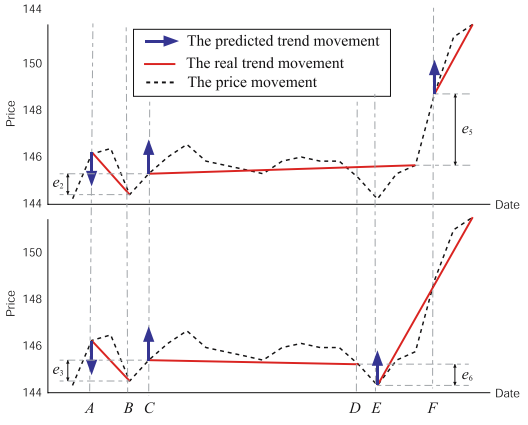


Fig. 2. Predictions on 0005 (HSBC) from 10/12/2006 to 11/20/2006

usually when the predictions are made at a later stage. This is why even though in some cases the prediction accuracy of different approaches is very similar, their resulting profits can be noticeably different.

Our proposed approach can make prediction at point E, rather than at point F. It is not because the error, e_6, exceeds the predefined threshold, but is because the contents of the news articles at point E are very different from those we have archived previously. Finally, it is worth noting that our proposed approach always makes profits, whereas all the other approaches may sometimes incur losses.

3.3 Further Discussions

One of the major differences between our proposed approach and the other three approaches is that we take news articles to assist us when determining the observed market states, such that we can possibly detect the market state changes in a much earlier stage.

Figure 3 shows the differences between using the traditional time series segmentation approach and our proposed multiple-source partitioning approach for identifying the market states. The top panel shows the stock price of the stock HSBC from 2/1/2006 to 3/30/2006. The traditional approach identifies seven segments by monitoring the changes of the stock prices only. The middle panel shows the tracking of the similarities of the associated events. Each bar denotes the similarity between the event happening at that particular time and the event happening at the beginning of the segment that the bar locates. In general, at the beginning of any segment, the similarity is high. It then drops steadily when the time passes. The bottom panel shows the identification results of our proposed multiple-source partition algorithm. Note that at time tick 30, the stock price starts a significant rising trend, and the traditional time series segmentation approach identifies this change until at time tick 42 where a drop trend begins. On the other hand, when we monitor the event similarities, starting from time tick 30, it drops dramatically at time 34 and even reaches 0 at time tick 36, which indicates that a new event appears. Our proposed algorithm notices this change, and

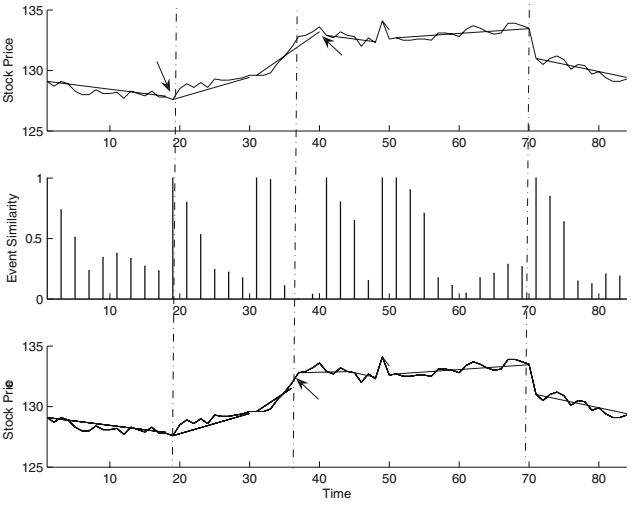


Fig. 3. Identifying market states: traditional segmentation approach vs. our proposed approach

declares a new market state at time tick 36, which is a time much earlier than that of the traditional approach. With information from multiple sources, our proposed algorithm is more sensitive to the changes in the stock market.

Another interesting problem for our approach is whether the bursty features from news articles would help to improve the prediction performance compared to other news based prediction approaches and in what extent it can help. We compare our proposed approach with news sensitive prediction[3], where the training is conducted in three phases: first the news articles relevant to a specific stock are aligned to the stock trend according to their timestamp, then the approach would cluster the news articles into positive and negative sample set, finally, the classification model is built to account for the prediction task. And in the testing phase, the arrival of a relevant news article would trigger the prediction system and an alarm will be generated if the news article is classified as a positive or negative news. The difference in performance of profit generation is highlighted in Table 3.

From Table 3, the cumulative profit of proposed approach outperforms news sensitive prediction in the selected stocks, where in most cases, the margin is larger than 20%. Even for stocks which have large exchange volume and a lot of relevant news articles (0001, 0941 and 0005), the performance of news sensitive prediction is not comparable

Table 3. Comparison between news sensitive prediction and our proposed approach

Stock	MS	NS	Stock	MS	NS
0083	0.35	0.01	0016	0.12	0.10
0001	**0.26**	-0.12	0002	0.11	-0.03
0006	0.06	0.03	**0941**	**0.35**	0.10
0008	0.16	-0.21	0883	0.23	0.09
0388	0.36	0.19	0011	0.14	-0.04
2388	0.13	-0.07	**0005**	**0.24**	0.08

to the proposed approach. The reason is that the news sensitive prediction makes a strong requirement on the dataset that the news articles for both training and prediction should be relevant to the specific stock. But in our experiment setup, the news articles are from a general finance news corpus. Under this settings, the approach in [3] is unable to distinguish the news articles with impact on the price change of a specific stock. The noisy training set eventually leads to the unsatisfying performance of the classification model. On the other hand, proposed looks deeper into the text corpus by identifying the relevant events as a set of bursty features significantly related to stock price changes and achieves much better results. Apart from the reason of dataset, instead conducting prediction purely based on the news articles as news sensitive prediction, our proposed prediction also considers the information from the time series of stock price, which gives the prediction a basis as well as additional confidence.

4 Related Work

The first systematic examination against the impacts of textual information on the financial markets was conducted in [11], which compares the movements of Dow Jones Industrial Average with general news during the period from 1966 to 1972. [18] developed an on-line system for predicting the opening prices of five stock indexes. By combining the weights of the keywords from news articles and the historical closing prices of a particular index, some probabilistic rules are generated using the approach proposed in [16]. [3] proposed a model for mining the impact of news stories on the stock prices, They utilized a t-test based split and merge segmentation algorithm for time series preprocessing and SVM for impact classification. Our approach is different from the previous work because our model is not solely based on classifying text information or analyzing historical stock prices. Our prediction is based on a collection of information from multiple data sources. [5] explored the segmental hidden Markov model to model the pattern of time series, where each state is responsible for the generation of a segment of overall time series. [5] focused on applying the segmental HMM model for pattern matching and did not consider the influence from information of environment. [7] proposed the Non-homogeneous hidden Markov model for relating precipitation occurrence at multiple rain-gauge stations to broad scale atmospheric circulation patterns. [10] described how to capture temporal and multivariate dependencies in the multivariate time-series data, with a review and comparison on HMM and NHMM. These studies just present a theoretical framework for statistical modeling, but do not address how to construct and control the model on complex information environments, like the stock markets.

5 Conclusion

In this paper, we model the stock market process by using a Non-homogeneous Hidden Markov Model (NHMM) which takes multiple sources of information into account when making a future prediction. Our model contains the following three major elements: (1) External event state sequence, which denotes the events happening within the stock market; (2) Observed market state sequence, which denotes the current market status (rise/drop); (3) Hidden market state, which conceptually exists but is invisible

to the market participants. We model the external events by using the information contained in the news articles, and model the observed market state by using the historical stock prices. Based on these two pieces of observable information and the previous hidden market state, we try to identify the current hidden market state, so as to predict the immediate market movement. We compare our proposed approach with three other existing techniques for stock market prediction. The performance of our approach is much superior than the existing approaches. These encouraging results indicate that our proposed approach is practically sound and effective.

Acknowledgment. This work was supported by a grant of RGC, Hong Kong SAR, China (No. 418206).

References

1. Adler, P.A., Adler, P.: The market as collective behavior. In: The Social Dynamics of Financial Markets, pp. 85–105 (1984)
2. Bodie, Z., Kane, A., Marcus, A.J.: Investments, 3rd edn. Irwin, Chicago (1996)
3. Fung, G.P.C., Yu, J.X., Lam, W.: News sensitive stock trend prediction. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS (LNAI), vol. 2336, pp. 481–493. Springer, Heidelberg (2002)
4. Fung, G.P.C., Yu, J.X., Lu, H.: The predicting power of textual information on financial markets. IEEE Intelligent Informatics Bulletin 5(1), 1–10 (2005)
5. Ge, X., Smyth, P.: Deformable markov model templates for time-series pattern matching. In: Proc. of KDD 2000, pp. 81–90 (2000)
6. Hellstrom, T., Holmstrom, K.: Predicting the stock market (1998)
7. Hughes, J.P., Guttorp, P., Charles, S.P.: A non-homogeneous hidden Markov model for precipitation occurrence. Applied Statistics 48(1), 15–30 (1999)
8. Jurafsky, D., Martin, J.H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice-Hall, Englewood Cliffs (2000)
9. Keogh, E.J., Chu, S., Hart, D., Pazzani, M.J.: An online algorithm for segmenting time series. In: Proc. of ICDM 2001, pp. 289–296 (2001)
10. Kirshner, S.: Modeling of multivariate time series using hidden Markov models. PhD thesis, Univedrsity of California, Irvine (2005)
11. Klein, F., Prestbo, J.A.: News and the Market. Henry Regenry, Chicago (1974)
12. Lavrenko, V., Schmill, M.D., Lawire, D., Ogivie, P., Jensen, D., Allan, J.: Mining of Concurrent Text and Time Series. In: Proc. of KDD 2000 Workshop on Text Mining (2000)
13. Luenberger, D.G.: Investment Science. Prentice Hall, Englewood Cliffs (1997)
14. Pang-Ning Tan, M.S., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Reading (2006)
15. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York (1986)
16. Wüthrich, B.: Probabilistic knowledge bases. IEEE Transactions on Knowledge and Data Engineering 7(5), 691–698 (1995)
17. Wu, H., Salzberg, B., Zhang, D.: Online event-driven subsequence matching over financial data streams. In: SIGMOD Conference, pp. 23–34 (2004)
18. Wuthrich, B., Permunetilleke, D., Leung, S., Cho, V., Zhang, J., Lam, W.: Daily prediction of major stock indices from textual www data. In: Proc. of KDD 1998, pp. 364–368 (1998)

Intra/Inter-document Change Awareness for Co-authoring of Web Sites

Stavroula Papadopoulou[1], Claudia-Lavinia Ignat[2], Gérald Oster[2], and Moira C. Norrie[1]

[1] Department of Computer Science, ETH Zurich, Switzerland
{papadopoulou,norrie}@inf.ethz.ch
[2] LORIA, INRIA Nancy-Grand Est, France
{ignatcla,oster}@loria.fr

Abstract. Systems that support the co-authoring of web sites often allow users to freely edit pages. This can result in semantic inconsistencies within and between pages. We propose a change awareness mechanism that monitors intra- and inter-document edits, taking into account changes made to a page and pages connected to it through *html* or *transclusion links*. The effect of all the changes is computed based on various metrics and on different semantic levels according to user preferences. A visualisation tool indicates how much a document and documents linked to it have changed. An edit profile allows users to easily spot parts with "interesting" changes within web pages.

1 Introduction

Until recently, the World Wide Web was used mainly as a read-only medium where authors publish to many readers. Now the trend is toward multiple interacting authors such as seen in the collaborative authoring of web sites and even web pages. Authors are often distributed in time and space, which can make the coordination of their work difficult. To avoid imposing certain strategies or working modes on authors, users are often allowed to freely edit web pages without constraints on when this happens or which parts are modified.

Changes to a web page may affect other parts of the page or pages linked to it and hence lead to semantic inconsistencies within or between pages. Keeping users aware of changes made by other users can help to resolve such inconsistencies or even prevent them. While many popular tools for the collaborative authoring of web pages such as Wikis, blogs and WebDAV applications succeed in enabling collaboration between users, they unfortunately provide almost no awareness to users about the changes done by their collaborators, and therefore do not prevent semantic inconsistencies.

In contrast, the Computer-Supported–Cooperative–Work (CSCW) community has identified awareness as an important feature of collaborative applications. *Change awareness*, defined as "the ability of a person to track the changes that other collaborators have made to a group project" [17], has been investigated by many research teams dealing with collaborative applications for

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 90–105, 2008.

text [4,11], graphical [17], or software engineering applications [2]. To the best of our knowledge, none of the existing CSCW approaches has been applied to provide awareness for the co–authoring of web pages where a web page can indirectly be modified through changes made to a linked page.

Inspired by the CSCW community, we address the need of users to track changes made over time to co–authored web pages. We identify two categories of changes that may appear: *intra–document changes* and *inter–document changes*. The first category includes changes made to a single web page. Users need to be informed in detail about the type of changes, which part of the page is affected, in which way and how much. An overview of the changes should be provided to the users so that they are aware of the group activity and can react quickly to changes that might affect their work. Web pages often contain links to other pages and the target pages may be semantically connected. Additionally, parts of web pages are often reused in other web pages resulting in "transcluded" web pages [10] indirectly linked to "compound" web pages [8] that include the reused information. In both cases, changes to a target/transcluded page may directly or indirectly influence the content of the source/compound page resulting in *inter–document changes*. In the process of collaborative authoring of a web site, it is therefore important that a user is also notified about concurrent changes made to linked documents.

Current solutions only offer the possibility to track changes made to a web page and therefore deal only with *intra–document* changes. While the possibility of following links from a source page is offered by some tracking tools, they do not relate the changes made to the target page with the source page. Further when *intra–document* changes are tracked, the awareness information presented to the user is computed at a single syntactic level of the document. Additionally, existing tools do not provide an overview of changes made or information about the user who made the changes.

In [13] we proposed a mechanism that computes awareness information about *intra–document* changes made to structured documents. We used this mechanism to extend an existing asynchronous collaborative text application to provide awareness information. We conducted a user study [16] to test our hypothesis that the resulting application provide increased awareness information and received positive feedback from the participants. We then applied the mechanism to a semi-synchronous collaborative text editor [15] and an asynchronous collaborative graphical editor [12]. All the editors used a structured document model. In this paper we show how we applied the same mechanism to collaborative situations where web sites are authored. We also show how we extended our awareness mechanism to deal with links between web pages.

We propose a mechanism that based on *html* and *transclusion links*, informs authors of source/compound web pages about changes made to the target/transcluded web pages. Our mechanism additionally computes and visualises change awareness about *intra–document* changes on the different syntactic document levels of one or more web pages. We begin in Section 2 with a motivating example to show the kinds of information that would be useful for users when

co–authoring web pages. In Section 3 we present our approach to computing change awareness for *intra–* and *inter–document* modifications and in Section 4 we introduce a visualisation tool to show the computed awareness information. We discuss related work in Section 5 and conclude in Section 6.

2 Motivating Example

In this section we use an example to motivate the need for users to be informed about intra-document and inter-document changes to a web site. Consider the case of a research group that maintains a web site about their activity. The structure of the website is shown in Figure 1.

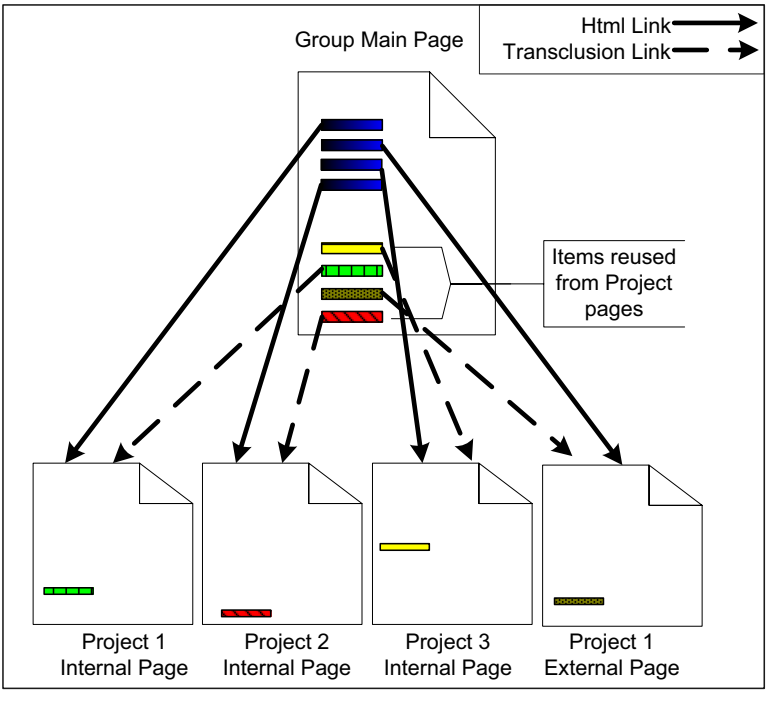


Fig. 1. Structure of a group web site

The group is involved in several projects (Project 1, Project 2, Project 3) with a subset of group members participating in each. The group web site is composed of a main page and a set of project pages. The main page contains a description of the group research topics, a list of the group members, a summary of the group projects together with links to the internal web pages describing the projects, and links to an external web page of Project 1 maintained by a collaborating research group. We assume that the summaries of the group

projects are transclusions of the short descriptions of the projects found on the associated web pages. Figure 1 shows the *html* and *transclusion links* from the group main page to the projects internal and external pages. To keep the group web site consistent, users working on the main page would like to be continuously informed about changes made to:

- *The main page itself.* Users need to be informed in detail about the type and location of changes, as well as their effect on the parts of the page where they were made and the page as a whole.
- *The transcluded parts.* Any change made to the source document of a transcluded part must be tracked and the compound document must be notified. After refreshing the transcluded part to show its latest status, awareness information about the detailed changes made to it need to be given as well.
- *The linked pages.* Users editing the main page should be informed about changes made to the linked web pages as these changes might produce inconsistencies in the main page. For instance, if a new project member is added to an internal project, the complete list of participants on the group main site should be updated. In the same way, if a new software release is announced on the external page of Project 1, this information could be used to update the group main page.

The visualisation mechanism that presents change information to the users should provide an overview of changes as well as details at a selected document level such as a section or paragraph, the transcluded parts and the linked pages.

Some tracking tools allow the possibility to follow links from a source page, but they do not relate changes made to the linked pages with the source page. Therefore, a user working on the source page is not aware about changes made to the linked pages. In the following sections we present our awareness mechanism for intra and inter-document changes. Users working on a web page are informed about changes made to the main page and linked pages without visiting those web pages.

3 Computation of Intra/Inter–document Awareness

In this section we present our awareness mechanism for intra- and inter–document changes. A flexible awareness mechanism is required to compute and visualise awareness information at a granularity level defined by the user, based on their current role and needs. We propose an awareness mechanism for documents conforming to a hierarchical structure. The use of an underlying structured document model enables the addressing of different document parts and document levels and therefore provides the required access to them which enables the tracking of changes made to them and the computation of the required awareness information. Our awareness mechanism includes the concepts of a node, i.e. a document part, an operation, i.e. a change made to a node, various metrics that can be used to measure the effect of an operation applied to a node and visualisation tools

to present the computed information. Finally, it includes the parent–child relationships between the document nodes that form the structured document. The relationships are used to compute the effect of a node's change to another node. All the above concepts are defined in a general way to allow the mechanism be applicable to any collaborative application that uses a structured document model, independently of the document type or the mode of collaboration.

We first explain the document model and the concept of operations used to describe web pages and the changes made to them, respectively. Then, we introduce the architecture of our approach and present in detail how we collect the intra- and inter–document changes made to web pages and compute the corresponding awareness information.

3.1 Document Model and Operations

We view each web page being collaboratively authored as a structured document. Through the structured document model, we can address different parts of a web page at various syntactic document levels and compute awareness information about how much they have changed at different granularity levels. We adopt the definition of a document node introduced in [6].

A node N of a document is a structure of the form
$N =<level, children, history, content>$, where

- *level* is the granularity level, $level \in \{0, 1, \ldots, n\}$ corresponding to node N,
- *children* is an ordered list $\{N_1, ..., N_m\}$ of child nodes,
- *history* is an ordered list of operations referring to child nodes,
- $content = \begin{cases} \text{object stored in node, if } N \text{ is a leaf node} \\ \sum_{i=1}^{n} content(child_i), \text{ otherwise} \end{cases}$

Changes to a web page are mapped to operations applied to specific parts of the page. Examples of operations are the insertion/deletion of a new paragraph to/from a section or of a sentence to/from a paragraph or of a word to/from a sentence etc. We define operations in a similar way to that described in [6].

An operation is a structure of the form op= $<type, level, position, content, length, user>$, where

- *type* is the type of the operation, $type \in \{insert, delete\}$,
- *level* is the level of the node in whose history the operation is kept,
- *position* is a vector of positions specifying the path from the root to the node where the operation is applied,
- *content* is a node representing the content of the operation,
- *length* is a vector with the number of units of each document level inserted or deleted by the operation,
- *user* is the user who generated the operation.

Following the document's structure, the effects of changes made to document parts at lower syntactic levels, such as word or sentence in the case of text, are aggregated to show the total changes made to the document at a higher

syntactic level such as a paragraph. Monitoring changes made to different parts of the document and evaluating their effect returns the required intra–document change awareness.

The operations that transform a web page from one version to another can be created in two ways: (1) by using a special web application, when authoring the page, that captures the changes and (2) by using a diff algorithm [9] between the two versions. When an application that captures user changes is used, the captured operations reflect the actual changes made to the page. When a diff algorithm is used, a set of operations that transform the initial state of the web page to its final state is created. Unfortunately, this set may not be identical to the set of actual changes made by users, resulting in loss of some information. The advantage, though, of a diff algorithm is that it can be used to compute the difference between two versions of any web page without monitoring the authoring procedure. Being aware of these issues, we chose to use a diff algorithm to enable the computation of awareness information for both internal and external pages. When addressing such problems, we believe that being able to offer awareness information, even if operations do not capture the exact changes, is far more beneficial than not delivering any information at all. Of course, in situations where only internal pages are authored and the accuracy of computed awareness is crucial, an authoring application that captures user changes can be used instead of a diff algorithm.

3.2 Architecture

The procedure followed to compute intra– and inter–document changes and the corresponding awareness information is shown in Figure 2. A *PageCache* is maintained locally for each user, including a copy of each of the web pages of interest to the user, as well as a copy of all the pages linked to them. We refer to all of these as *monitoredPage*s. Additional information is kept in each *monitoredPage* showing how much each part has changed. A background process periodically calls the method *updateValues* for each *monitoredPage* to check whether any intra– or inter–document changes occurred and compute their effect.

To compute awareness information for the intra–document changes, we call the method *getIntraValues* for the currently monitored page. *getIntraValues* takes the cached (old) version of the page and the online (current) version from the web and compares them. If they are not identical, the procedure in the block "condition", shown in Figure 2, is followed. Initially, a diff algorithm is used to compute the changes that transform the old version of the page into the current one in terms of operations denoted as *changes*. The *changes* are used by the method *computeIntraValues* to compute the *intraValues* attached to the changed parts of the page. This procedure is detailed in Subsection 3.3. Finally, these values are stored at the *monitoredPage*, the new version of the page is stored at *PageCache* and *getIntraValues* returns the computed intra values associated with parts of the document.

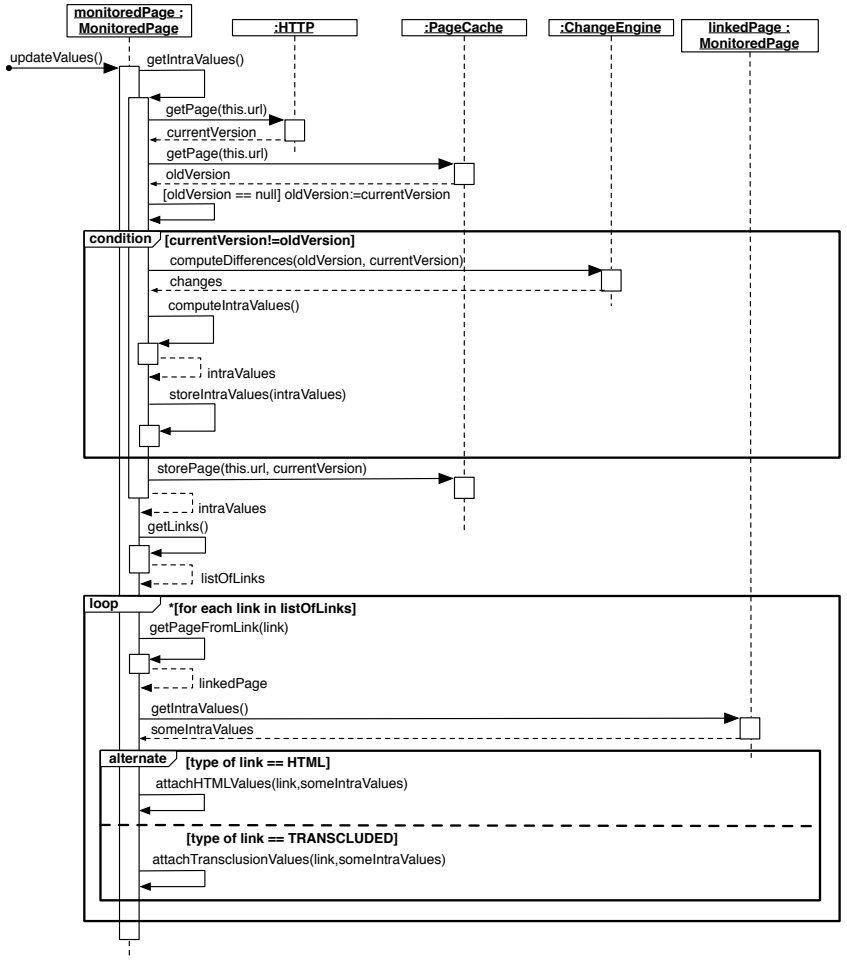


Fig. 2. Awareness mechanism for computing intra– and inter–document awareness

To compute the awareness information for the inter–document changes, the linked pages need to be accessed. Therefore, the list *listOfLinks* with all links of the *monitoredPage* is retrieved. Note that both *html* and *transclusion links* are included in the list and *listOfLinks* is accessed after the current version of the *monitoredPage* has been stored locally. Therefore the list contains the links included in the current version of the page. Any link that was present in the old version but removed in the current version is not included. Similarly, any link not present in the old version but added in the current version is included. We believe it is reasonable to search for changes only in pages currently linked to the *monitoredPage* as the removal of a link is already depicted as a change in the *monitoredPage*.

In the loop that follows, we handle each of the links separately. We initially retrieve the *linkedPage* and compute the intra–document awareness for it by calling the method *getIntraValues* as for the *monitoredPage*. As explained before, any new version of the *linkedPage* is detected and the changes made to it as well as the corresponding awareness information are computed. Finally, the object *someIntraValues* is returned, including the computed awareness information. *someIntraValues* may be null if the current version of the *linkedPage* is identical to the old version and different than null if the versions differ. Note that if the link was added to the current version of *monitoredPage*, the *linkedPage* is inserted in the *pageCache*. The old version of the *linkedPage* is identical to the current one, so the *someIntraValues* will be null. Changes made to the new linked document will be monitored and reported starting from the moment the link is added. This part of the loop is executed for both *html* and *transclusion links*. The intra–awareness information computed for the *linkedPage*s is then attached to the structure of the *monitoredPage* based on the type of the link. The procedures *attachHTMLValues* and *attachTransclusionValues* are described in Section 3.4.

It is obvious from the procedure described here that we compute inter–document awareness information based on first level links only, i.e. we consider changes made only to pages directly linked to the current page. Although the procedure can easily be adapted for other levels, i.e. links included in linked pages, we believe that the information delivered in this case would not be of additional value to the collaborating situations we address in our current work.

3.3 Intra-document Awareness

Since each change made to a page is mapped to an operation, we define the concept of an *opValue* as the effect of the change on the page. The effect of an operation is not a single value, but a collection of values, each of which is computed according to some criteria called *metrics*. Each *metric* addresses specific information and is used to compute awareness information of interest to the users. Therefore, we define the metric $opValue$ as follows.

Definition 1. *$opValue$ is the number of the units of level ulevel that are inserted or deleted by the operation op.*

$opValue(op, ulevel) =$

$$\begin{cases} 0 & if\, level(op) > ulevel \\ 1 & if\, level(op) = ulevel \\ count(unit_j)_{\substack{unit_j \in content(op)\,\&\\ level(unit_j) = ulevel}} & if\, level(op) < Ulevel \end{cases}$$

Two of the metrics we have defined are the number of characters ($ulevel =$ "character level") and the number of sentences ($ulevel =$ "sentence level") inserted or deleted by an operation. An operation inserting a sentence of 30 characters would have an *opValue* of 30 if the metric selected by the user is the number of characters or a value of 1 if the metric selected is the number of sentences. The

second metric would be selected if a user wanted to be informed only about "important" changes on the level of whole sentences inserted or deleted. As a result, any operation that inserts/deletes a unit of a lower level than sentence would have an *opValue* of 0 and be omitted from the computation of awareness information. On the contrary, if a user needs to be informed about spelling mistakes as well, then the first metric is chosen. Note that operations that delete content are handled in the same way as operations that insert content with *opValue* computed in the same way and always having a positive value.

The concept of a *nodeValue* is introduced to describe the effect of all changes of a specific type made to a node in a document's structure, i.e. a document part and all the document parts of lower syntactic document levels that belong to it.

Definition 2. *We define the nodeValue of a node N to be the sum of two components: the sum of the opValue of all the operations applied to the node and the sum of the nodeValue of all the node's children.*

$nodeValue(N, ulevel) =$

$$\sum_{op_i \in history(N)} opValue(op_i, ulevel) + \sum_{N_j \in children(N)} nodeValue(N_j, ulevel)$$

nodeValue of a node at the character level is equal to zero, since its history and children are empty sets. Since there can be more than one type of operation, there is also more than one *nodeValue* defined for each node. For instance, a *nodeDeletionsValue* is defined for a node to represent the sum of the *opValue* of all the delete operations applied to the node and the nodes of lower syntactic document levels that belong to it. Further, since there is more than one *metric* to compute an *opValue*, there is also more than one instance of *nodeDeletionsValue*.

A clear advantage of a structured page is the fact that the parts of the page addressed can be of different levels. For example, this means we can compute the total changes made to each of the paragraphs included in the *Group Main Page* shown in Figure 1, or for each of the sections, etc. The value computed for a specific section reflects the number of paragraphs inserted or deleted to the section and the number of sentences inserted or deleted to the paragraphs etc.

A more detailed description of our awareness mechanism presented in this section can be found at [14]. The concepts described above are applicable to collaborative authoring of structured documents of any type. We already used them to compute awareness information on co–authored text and graphical documents. The formal definition of the concepts as well as details of their use for graphical and text documents can be found in [12] and [14], respectively.

3.4 Inter-document Awareness

We now describe the procedure followed to attach to a *monitoredPage* the awareness information computed for a *linkedPage*. As described in Section 3.2, the awareness information is computed in the same way for all *linkedPage*s linked through either an *html* or a *transclusion link* to a *monitoredPage*. This awareness information comprises a set of *nodeValue*s computed for each of the parts of the *monitoredPage*. To keep the awareness information delivered through the *monitoredPage* up–to–date, we need to attach the computed values to the parts of the *monitoredPage* that are affected.

If the link to the *linkedPage* is an *html link*, the part of the *monitoredPage* where the link is included needs to be updated. We decided to attach to this part all the awareness information computed for the *linkedPage*. In this way, we provide the user the possibility while browsing through the *monitoredPage* and watching intra–document changes, to also be informed about the changes made to the *linkedPage* without loading the page. Our aim is to deliver to the user an edit profile with the quantity of the changes in the *linkedPage*, as well as the parts of the page that changed. In this way, the user can decide whether they need to revisit the linked page and update the information included in the *monitoredPage*. The way the awareness information is presented to the user is shown in Section 4.

If the link to the *linkedPage* is a *transclusion link* then only part of the awareness information computed for the *linkedPage* needs to be transferred to the *monitoredPage*. Through the transclusion mechanism, the part of the *monitoredPage* that is transcluded from a *linkedPage* is updated to include the latest changes made to it. However, updating the transcluded part in the *monitoredPage* is not enough. Awareness information that is attached to this part, needs to be transferred to the *monitoredPage* as well so that users are informed about the amount of changes. To do so, the awareness information computed for the *linkedPage* is scanned and only the values computed for the transcluded part are kept. This mechanism is heavily dependent on the transclusion mechanism and the metadata that is created during the authoring process of the *transclusion link*. This metadata is usually new tags inserted in the transcluded or the composed pages, or other anchors that can be used to define the beginning and end of the transcluded part [7]. Using this data, we identify the transcluded part in the *linkedPage*, extract the awareness information computed for it and copy it to the corresponding part of the *monitoredPage*.

4 Visualisation of Intra/Inter–document Awareness

We revisit our motivating example to present the tools used to visualise the computed awareness information. We use only a representative sample of changes made to two of the example pages to demonstrate how the visualisation tool informs users about different kinds of changes. We assume there are some intra–document changes (annotated with "A" in Figures 3 and 4) made to the main page and the internal page of project 1 (interactive paper project). The main

page has an *html link* to the project page through the phrase “interactive paper” in the first paragraph (B) and the paragraph about the interactive paper project is a transcluded paragraph (C) from the project page. The changes made to the main page are insertions of text in the paragraph about group research. The changes made on the project page are an insertion of a word in the first paragraph (the transcluded paragraph), the insertion of two new paragraphs (the sixth and seventh) and the insertion of a new member in the list of group members (paragraph number nine). Note that titles are handled as paragraphs of one sentence and changes of the project page are considered as inter–document changes for the main page. Finally note that an external page linked through the main page could also be used instead of the internal project page and the computed awareness information would be visualised in the same way.

We assume that the main page contains 10 paragraphs. The window in Figure 3 presents the first 6 paragraphs of them.

In Figure 3, we also show how we visualise the intra–document changes of the main page. Two edit profiles are introduced on the sides of the page. The profile on the right side shows an overview of all changes made throughout the page, while the profile on the left shows all changes made to the portion of the page currently shown. In order to construct the profile on the right side presented in Figure 3, the awareness information computed for each paragraph is used. By using the profile, the user can instantly spot that there have been a lot of changes on the second and the last paragraph and some changes in the sixth and eighth paragraph. The left profile can be thought of as a zoomed version of the right profile. The information provided by the left profile corresponds to the first six paragraphs, which is the part of the right profile marked by the scrollbar.

The reason for the existence of the left profile, is that the information provided by it can be on a different syntactic level. In Figure 3, for instance, the left profile shows changes made to each of the sentences of the first six paragraphs. In this way, a user that would notice the large amount of changes made to the second paragraph, could further filter the awareness information and find how

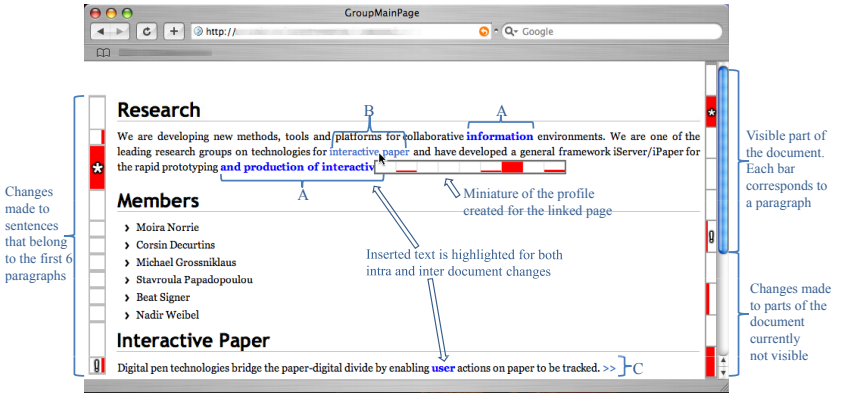


Fig. 3. Visualisation of intra– and inter–document changes made to the main page

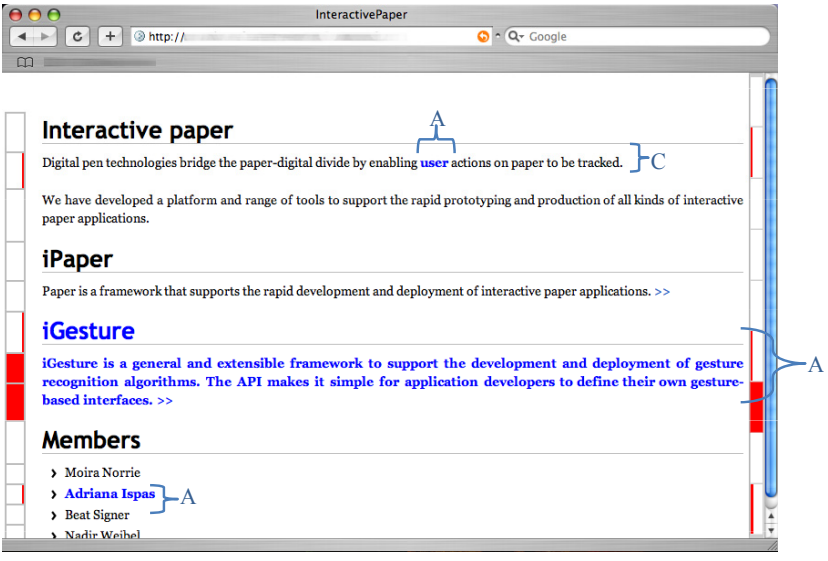


Fig. 4. Visualisation of intra–document changes made to the project page

the changes in the paragraph are distributed to the paragraph sentences. The height of the bar corresponding to each sentence is adjusted to the length of the sentences for the left profile, to ease the mapping of changes to document parts. The width of a bar in each profile represents the normalised number of words inserted or deleted in the corresponding document part. Please note that a user is able to configure the syntactic levels used in the profiles as well as the detail of the information provided through them to visualise information about changes of various types on a user-defined granularity. The intra–document changes made to the project page are also visualised in the same way as shown in Figure 4.

Since the first paragraph of the main page includes an *html link* to the project page, changes made to the project page might influence the content of the main page. For instance, the introduction of a new prototype (iGesture) in the project page might need to be included in group research summary (first paragraph) in the main page. Hence, when a user points with the mouse on the html link, an overview pops up that shows the changes made throughout the project page. We expect this feature to give the user a rough idea of the amount of changes made to the project page as well as the parts of the page that changed the most to help the user decide whether they need to visit the project page to see the changes in detail and whether the content of the main page should be modified as well. With the "*" symbol on the profiles we inform users if there are changes made to pages linked through *html links*.

The changes made to transcluded parts are presented as intra–document changes through the edit profiles. The sixth paragraph of the main page is a transcluded paragraph. The changes made to it are presented through the two

profiles in the main page, exactly in the same way they are presented in the project page. The distinction between the transcluded and compound page is made through the "!" symbol added to the compound page at the parts of the profiles that correspond to the transcluded parts.

Our awareness mechanism and the visualisation tool introduced above aim to increase the amount of awareness information presented to users about changes made to a page and the pages linked to it. In the absence of adequate awareness information, a number of problems could appear. For instance, a user editing the main page would not be informed about changes made at the linked page (interactive paper project). They would not know that the list of members in the interactive paper project is updated since a new member was added. As a result, the list of members in the main page would not be updated to include the new member. Additionally, users would not be informed about the new prototype that is released in the interactive project and therefore, information about the released frameworks on the main page would remain obsolete. Our awareness mechanism supports the users to avoid all of the above situations.

5 Related Work

A closer look at CSCW research reveals various approaches developed to provide change awareness in text-based collaborative applications [4,11], software development collaborative tools [2] and applications with 2D graphical scenes of objects [17]. However, not much work has been done on providing awareness of changes in the co-authoring of web pages.

In [3], a notification mechanism is proposed for tracking changes made to web pages. Users can mark regions of interest on a web page that are stored as bitmaps. The system periodically checks if the selected regions visually differ from the stored ones with a certain severity index fixed by the user. If an interesting region is modified, the user is notified by the system. This approach is image-based and assumes the web page retains its spatial layout. In contrast, our approach provides an awareness mechanism that is attached to the document structure and therefore is independent of page layout changes. Moreover, our approach quantifies changes.

Tools for computing the difference between HTML web pages [5] present superimposed on the document, the changes made to it. While this enables the users to detail the changes made by their collaborators, it does not provide any support in finding "hot areas" of the document where "important" changes have been made. The lack of an overview of all changes made throughout the web page, forces the users to scroll through the document to find areas with "interesting" changes. This can be frustrating and time inefficient especially in case of long documents. A second drawback of existing approaches is that they cannot measure the effect of a change made to a document, or cannot distinguish changes on different syntactic levels. For instance the fact that an insertion of a two–word sentence has a different effect from an insertion of a twenty–word sentence is not depicted by existing collaborative applications. Both are presented as

one change. Finally, it is not possible for a user to be informed separately about spelling mistakes, i.e. changes that altered only some characters and separately about insertions and deletions of whole paragraphs.

Notification tools, such as Watch That Page [19], ChangeDetect [1] and Website Watcher [20], track changes made to specified web pages. Some of them, for instance Website Watcher [20], allow the possibility to automatically add to the list of monitored pages those pages linked to the main web page. Users are then notified if changes are made to one of the monitored pages. However, no information is provided about the relation between the monitored pages and how changes in a linked web page might influence the changes in the main web page. Our approach keeps users aware about changes made to pages linked to the current web page and, in this way, helps users avoid semantic inconsistencies.

Wikipedia [21] provides users with the possibility of tracking changes made to a wiki page by means of a revision history attached to the page. It consists of the old versions and a record of the date and time of every edit, as well as the user who made it. Wikipedia also offers the "watch pages" feature by which users receive email notifications when specified pages change. The notification mechanism provides the number of added or deleted bytes to each page. With our approach, in addition to the intra-document awareness, we provide inter-document awareness where we track changes on linked documents to the web page that might influence the content of that web page. We track changes even on the documents identified by the links added during the process of collaboratively authoring the web page. Moreover, by means of the edit profile, we present changes at different granularity levels.

In [18], an overview of the evolution of a wiki page is presented using a history flow visualisation to provide information about how a group contributed to the web page or how a modification influenced the current version of the web page. Unfortunately, no awareness mechanism is offered for the changes made to the linked pages to the current wiki page. Moreover, the document evolution is computed only on the document level and no information is provided on the changes made to parts of the document.

6 Conclusion and Future Work

We have presented a change awareness mechanism that tracks intra- and inter-document edits in the co-authoring of web pages. We described how metrics are computed to quantify the changes made inside a document as well as on transcluded parts of the document and linked documents. Moreover, we described a visualisation tool based on edit profiles that enable users to have an overview of changes done on a web page and any pages linked to it. This allows users to easily spot "interesting" changes done on document parts and browse these changes on finer granularity levels.

We plan to test the usability of our approach by conducting user studies on the prototype that implements the ideas described in this paper. During user studies we plan to experiment with various visualisations. Finally, one of our

future work directions is to extend our awareness mechanism to take into account not only physically linked documents, but also semantically linked documents in the context of semantic wikis or more general semantic web.

References

1. ChangeDetect. Be the first to know (2008), http://www.changedetect.com/
2. Eick, S.G., Steffen, J.L., Eric, J., Sumner, E.: Seesoft-A Tool for Visualizing Line Oriented Software Statistics. IEEE Transactions on Software Engineering 18(11), 957–968 (1992)
3. Greenberg, S., Boyle, M.: Generating custom notification histories by tracking visual differences between web page visits. In: Proceedings of Graphics Interface (GI 2006), Quebec, Canada, pp. 227–234 (June 2006)
4. Hill, W.C., Hollan, J.D., Wroblewski, D., McCandless, T.: Edit wear and read wear. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 1992), Monterey, CA, USA, May 1992, pp. 3–9 (1992)
5. W3C HTML Diff. service (2008), http://www.w3.org/2007/10/htmldiff
6. Ignat, C.-L., Norrie, M.C.: Customizable Collaborative Editor Relying on treeOPT Algorithm. In: Proceedings of the 8th European Conference on Computer-supported Cooperative Work (ECSCW 2003), Helsinki, Finland, pp. 315–334. Kluwer Academic Publishers, Dordrecht (2003)
7. Kolbitsch, J., Maurer, H.: Transclusions in an html-based environment. Journal of Computing and Information Technology 14, 161–174 (2006)
8. Krottmaier, H., Helic, D.: Issues of transclusions. In: Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education (E-Learn 2002), Montreal, Canada, pp. 1730–1733. AACE (2002)
9. Myers, E.W.: An $O(ND)$ difference algorithm and its variations. Algorithmica 1, 251–266 (1986)
10. Nelson, T.: Literary Machines. Mindful Press (1982)
11. Neuwirth, C.M., Chandhok, R., Kaufer, D.S., Erion, P., Morris, J., Miller, D.: Flexible diff-ing in a collaborative writing system. In: Proceedings of the 1992 ACM conference on Computer-supported cooperative work (CSCW 1992), Toronto, ON, Canada, pp. 147–154 (November 1992)
12. Papadopoulou, S., Ignat, C.-L., Norrie, M.C.: Awareness model to overview modifications in collaborative graphical authoring tools. In: Ninth International Workshop on Collaborative Editing Systems (IWCES 2007) - The 2007 International ACM Conference on Supporting Group Work (GROUP 2007), Sanibel Island, FL, USA (November 2007)
13. Papadopoulou, S., Ignat, C.L., Oster, G., Norrie, M.C.: Increasing awareness in collaborative authoring through edit profiling. In: Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2006), Atlanda, GA, USA (November 2006)
14. Papadopoulou, S., Norrie, M.C.: How a structured document model can support awareness in collaborative authoring. In: Proceedings of the IEEE Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2007), New York, NY, USA, November 2007. IEEE Computer Society, Los Alamitos (2007)

15. Papadopoulou, S., Norrie, M.C.: Shadow document sets for synchronously-aware asynchronous collaboration. In: Proceedings of the 3rd International IEEE Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2007), New York, NY, USA. IEEE Computer Society, Los Alamitos (2007)
16. Papadopoulou, S., Reuss, E., Norrie, M.C.: A user study of edit profiles in collaborative authoring systems. In: Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS 2008), Irvine, CA, USA (May 2008)
17. Tam, J.R.: Supporting change awareness in visual workspaces. Master's thesis, Department of Computer Science, University of Calgary, Alberta (2002)
18. Viegas, F.B., Wattenberg, M., Kushal, D.: Studying cooperation and conflict between authors with history flow visualizations. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2004), Vienna, Austria, pp. 575–582 (April 2004)
19. WatchThatPage. Your monitor for changes on the web (2008), `http://www.watchthatpage.com/`
20. Website-Watcher. Save time, stay informed (2008), `http://aignes.com/`
21. Wikipedia, the free encyclopedia that anyone can edit (2008), `http://www.wikipedia.org/`

Requirements for Rich Internet Application Design Methodologies

Jevon M. Wright and Jens B. Dietrich

Institute of Information Sciences and Technology,
Massey University, Palmerston North, New Zealand
j.m.wright@massey.ac.nz,
j.b.dietrich@massey.ac.nz

Abstract. Rich Internet Applications (RIAs) are quickly becoming the *de facto* standard for interactive web applications on the Internet, featuring rich interfaces that increase user usability and efficiency. These technologies increase the complexity of implementing web applications, making it difficult to address non-functional requirements such as application quality and reliability. There is much activity in developing modelling languages for web applications, but RIAs introduce additional concerns for application developers. Without identifying the requirements of interactive web applications, we cannot quantitatively compare different formal methodologies nor suggest they are robust enough for industry.

In this paper we present a comprehensive list of web application modelling requirements, derived from previous work and existing real-world interactive web applications. We use these requirements to then propose an industry-inspired benchmarking application, which allows us to evaluate approaches to handling the complexity of modelling real-world applications.

Keywords: interactive web applications, Rich Internet Applications, web engineering, requirements, benchmark.

1 Introduction

For the last decade, web applications are increasingly becoming the standard for communication and interaction, allowing any connected user on the Internet to browse information using standardised protocols and a web browser. Many approaches to model these web applications have been proposed in the past, such as WebML [1], UWE [2] and W2000 [3]. Recently, the concept of *Rich Internet Applications* [4] has arguably redefined the environment of web applications – advocating rich user interfaces and improving user participation – and is transforming users from content consumers to providers [5].

This has increased the complexity of web development, and consequently web developers have found a greater need for the use of formal methodologies to assist in the development and deployment of these interactive web applications [4]. However, very little work has been done in evaluating existing methodologies,

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 106–119, 2008.

or proposing a comprehensive list of requirements of RIAs. This paper aims to satisfy these real needs by defining the expressive requirements of interactive web applications, and demonstrating the use of these requirements by proposing a sample benchmarking application.

We provide a brief background and our motivation for this work in Section 2. We then propose our requirements in Section 3, along with a discussion on their development. We combine these requirements into a fully featured benchmarking application, *Ticket 2.0*, in Section 4. A discussion of our contributions and future work is presented in Section 5, and we finally conclude our work in Section 6.

2 Motivation

Software development is a complex activity, and it is expected by industry that the use of formal methodologies and modelling languages to abstract away from this complexity increases the reliability, usability, security and maintainability of this software [6,7]. Web applications are a form of software that presents additional unique challenges and requirements to desktop software [7], and many modelling language approaches have been proposed to solve this additional complexity.

Despite this complexity, developers tend away from using such formal methodologies, instead advocating for proprietary or outdated approaches, even though formal methodologies are expected to be beneficial [6,8]. This may be a symptom of existing modelling languages being unable to express the unique requirements of web applications [7,9]; consequently, a methodology which is more expressible with regards to web applications should be beneficial to the web development community.

Past work on identifying the requirements of RIAs have tended to focus on the qualities of the methodology surrounding it or its software support [4,10], with less focus on the functional requirements of these applications. While web applications may be considered a primitive form of hypermedia [6], they have qualities that cannot be addressed with existing hypermedia modelling approaches [4]. In our paper we propose a comprehensive list of RIA requirements.

Along with correlating these requirements with existing work, we may also prove their validity by highlighting their actual usage in existing web applications. We amplify this step by consolidating all of these requirements into a proposed *benchmarking application*. Similar approaches are used in the domains of business rules [11] and enterprise software modelling [12], and this implementation is crucial to prove the real-world suitability of a formal methodology [6,13].

3 Requirements

Due to its relative infancy, there is little work on identifying the functional requirements of RIAs, with most research to date spent on identifying the technical and process requirements of appropriate methodologies. Whilst it is important

to consider the requirements of the process surrounding a modelling language, this is generally more flexible than the issues raised by the expressiveness of the modelling language itself. This is especially valid with web applications, as web concepts such as sessions and e-mails are largely ignored in existing approaches [14]. Consequently, we chose to develop our RIA requirements by studying real-world examples in industry:[1]

1. **Gmail**: Web-based e-mail by Google.
   `http://www.gmail.com`
2. **Calendar**: Google Calendar, a collaborative online calendar.
   `http://calendar.google.com`
3. **Reader**: Google Reader, an offline-enabled feed reader.
   `http://reader.google.com`
4. **Docs**: Google Docs, a collaborative office suite.
   `http://docs.google.com`
5. **Last.fm**: A social network-enabled music site.
   `http://www.last.fm`
6. **Pages**: Google Page Creator, an online web publishing suite.
   `http://pages.google.com`
7. **Facebook**: A social networking platform.
   `http://www.facebook.com`

In Tables 1 and 2 we present our proposed 59 core requirements of interactive web applications. Each requirement is based on an actual feature of RIAs, and is presented along with an example of their usage. They are grouped into six categories solely for ease of reference. We have purposefully ignored some basic data and presentation requirements;[2] these trivial aspects are covered by requirements such as *View Data* and are omitted for clarity.

These proposed requirements are ideal for evaluating and comparing different web modelling languages, and this approach has been taken before in evaluating older web modelling languages [4,9,10]. Indeed, it would be very useful to evaluate these requirements in a similar manner to Christodoulou et al [10]. As our previous work is concerned with a similar evaluation [14], we instead focus our attention on suitable methods to create and validate modelling language to address these requirements.

4 Benchmarking Application

Benchmarking applications[3] are a technique that may be useful in identifying the expressiveness of different technologies, and this concept has been used before

[1] The interested reader will note that most of these applications are developed by Google; indeed, Google has focused their business model significantly around RIAs.

[2] Such as the ability to use an external database, linking between pages, or being able to display the content in HTML.

[3] Instead of the classic definition of a performance benchmark, this is instead a *functional benchmark*.

Table 1. Interactive Web Application Requirements (1)

#	Requirement	Example
Data		
D1	Static Pages	Gmail: Static help pages
D2	View Data	Gmail: View an e-mail
D3	Update Data	Gmail: Create an e-mail
D4	Pagination	Gmail: Display e-mails in pages
D5	Provide Data Feed	Last.fm: Provide RSS feed of recommendations
D6	Use Web Services	Calendar: Use external iCal feed
D7	Offline Data	Reader: Download new feeds before going offline
D8	Offline Resources	Reader: Download resources before going offline
D9	Web Service Provider	Facebook: Provide Facebook application using API
D10	Uploading Files	Gmail: Adding attachments
D11	Access Server Data	Gmail: Download new message headers
D12	Local Variables/Data	Docs: Download document source to client
D13	Cookies	Gmail: Recall last input language
Events		
E1	Scheduled Events	Calendar: Event reminders on client and server
E2	Client Timer Support	Gmail: Check server for new e-mails
E3	Server Timer Support	Gmail: Check POP3 servers for new e-mails
E4	Async Form Validation	Last.fm: Check in entered event artist data
E5	Client Form Validation	Gmail: Warn user if subject is missing
E6	Server Form Validation	Gmail: Sending an e-mail to an invalid address
E7	User Collaboration	Docs: Two users can work on the same document
E8	Browser-Based Chat	Gmail: Google chat
E9	Out-of-Order Events	Docs: Dealing with edit events with multiple users
E10	Server Transaction Support	Gmail: Purchasing more storage space
Users and Security		
S1	User Authorisation	Gmail: Sign in
S2	Session Support	Gmail: Sign in
S3	User Logout	Gmail: Sign out
S4	Automatic User Auth	Gmail: Log in automatically
S5	User Security	Calendar: Only certain users can access a calendar
S6	Group Security	Calendar: Shared calendars secured to certain groups
S7	Security Levels	Calender: Read/write/change sharing permissions
S8	Single Sign-In Solutions	Google Services; OpenID
S9	Personalisation	Calendar: Display a custom timetable format
User Agents		
A1	Browser Identification	Gmail: Redirect user if user agent fails requirements
A2	User Redirection	Gmail: Redirect to e-mail web links
A3	Multiple Browser Support	Gmail: Load different interfaces depending on agent
A4	Multiple Outputs	Calendar: Provide a feed in iCal, XML, HTML
A5	Client-Side Application	Gmail: Webmail application
A6	Load Additional Scripting	Gmail: *Contacts* menu loads another script
A7	Back Button Control	Gmail: A user cannot go back once logged out
A8	Plugin Support	Gmail: Play MP3 attachment
A9	Plugin Communication	Last.fm: Clicking on a track updates the Flash player
A10	Navigation Control	Gmail: Update URL fragment identifier

Table 2. Interactive Web Application Requirements (2)

#	Requirement	Example
Interaction		
T1	E-mailing Users	Gmail: Can send e-mails
T2	E-mail Unsubscription	Facebook: User can unsubscribe from all e-mails
T3	Mobile Phone Communication	Calendar: Can send text message reminders
T4	Internationalisation Support	Last.fm: Different locales
T5	Multiple Domain Support	Last.fm: Different domains display different locales
User Interface		
U1	Presentation	Calendar: Displaying a particular user interface
U2	Client-side Scripting	Gmail: Home page displaying available space
U3	Drag and Drop	Calendar: Can drag and drop events
U4	Loading Time Support	Gmail: Switch to HTML view after 30 seconds
U5	Keyboard Shortcuts	Calendar: Can browse using keyboard
U6	Opening New Windows	Pages: Open links in new windows
U7	Pop-up Dialog Boxes	Gmail: Can compose an e-mail in a new window
U8	Runtime Interface Updates	Gmail: Update *Unread Mails* in real time
U9	Static Views (HTML)	Gmail: Provide a static HTML view
U10	Modal Dialogs	Pages: Inserting an image shows a modal dialog
U11	Use External Components	Facebook: Transitions with *script.aculo.us*
U12	Provide External Libraries	Gmail, Calendar: A consistent calendar input box

in a variety of different domains [11,12]. In the search for such a benchmarking application for web development however, we have not yet found any web application that matches all of our requirements simultaneously. We suspect this is due to the complexity such an application would burden on a development team, and it is precisely this reason that a structured formal methodology would be appropriate. This also means that there is no suitable application from which to build upon.

Ideally a benchmarking application for RIAs would involve the fields of social networking, e-commerce, web services, scheduled events, business integration and consumer interaction. It would be difficult to adapt common academic scenarios such as library or student applications to address all of these requirements. A sensible option would be adapting an existing web application, but existing applications are designed primarily for user simplicity and not feature usage. Extending an existing application may entangle too much additional complexity[4], which is important to consider when realising that a poorly designed modelling language may require significant model duplication[5] in order to fulfill the benchmark.

Whilst combining all RIA features into one application will exponentially increase its complexity, it is this complexity that will be a valuable learning exercise

[4] Consider re-implementing Gmail from scratch, compared with implementing only a single client-side application.

[5] Consider that an application with client-side, server-side and mobile interfaces may require at least three separate but functionally identical models.

into how a methodology handles semi-realistic web applications. As such, we propose that a simple event ticketing application, combined with social networking features, is ideal. This proposed application meets all of the requirements we proposed in Section 3, as shown in Table 4. We also argue that while developing our own benchmarking application is definitely a challenge, it will be less complicated and more accessible in the long term than trying to extend an existing web application.

In the rest of this section we present our social networking-enabled, event ticketing application titled *Ticket 2.0*. Its business goal is to provide a rich interface for users to browse upcoming events and book tickets using a credit card. They may interact with other users on the site through friends lists and chat rooms on the event detail pages themselves, permitting open discussions and user interaction. It also aims to provide a unified interface for event managers, allowing them to schedule upcoming events and track their progress.

The conceptual structure of the application is presented in Figure 1, and the ticket booking application flow is shown in Figure 2. These figure have been purposely presented without using any existing modelling notations to try and be as independent as possible. Elements shaded gray indicate features that are navigable from every page,[6] likely as part of a common navigation header. Due to space restrictions this is not a complete formal specification, and the following sections will become quite technical, however straight forward for an experienced developer to implement. The full specifications for *Ticket 2.0* are available online at `http://openiaml.org/`.

4.1 Application Properties

The site is provided in two locales, with two separate domains selecting the appropriate display language. The user may be automatically logged in through cookie identification, if this feature is selected by the user. If the user visits with a mobile phone, a smaller set of the application is presented (highlighted in light gray), which does not include manager or administrator functionality. Pages marked with an asterisk may be visited offline if the user has appropriate technology[7] installed. The application is divided into three secure sections, which only particular types of users may access. The *Book Ticket* page involves a client-side application, and is described with detail in Section 4.6.

4.2 Public Pages

Home. The home page describes the application, and allows the user to switch locales. It also allows the user to switch between the mobile and full versions of the application.

Signup, Login, Logout. Allows the user to signup, login, or logout. These pages are secured through HTTPS. When logging in the user is presented with an option to remember their authentication details.

[6] These pages may be called *landmarks* [15].

[7] Such as Google Gears: `http://gears.google.com`.

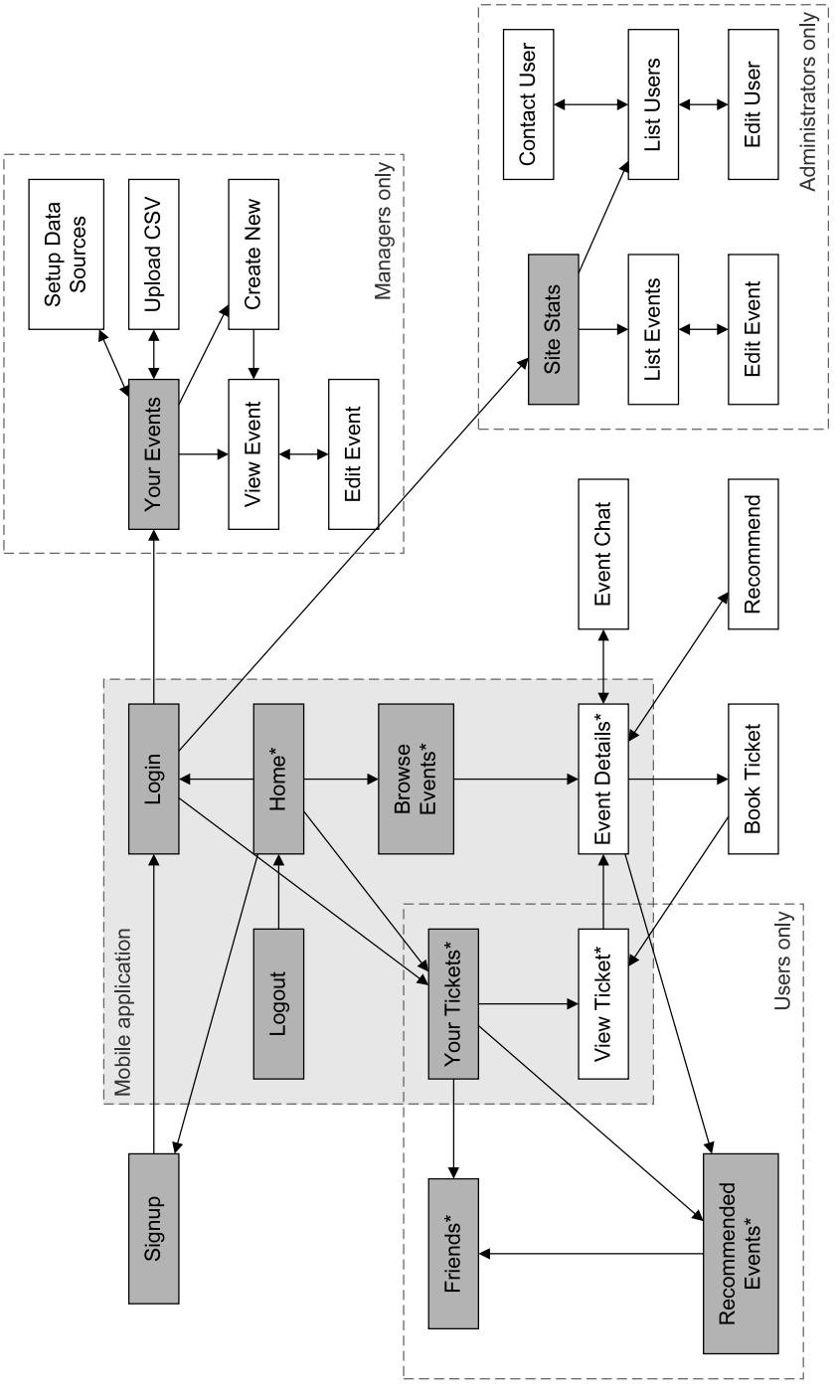


Fig. 1. Ticket 2.0 Application Page Structure

Browse Events. Lists all events in the system, presents a Google Maps mashup [16] of events, and plays MP3 samples for selected events. It also provides a public API for event listings. The user may browse the listings using keyboard shortcuts, and uses a standard search widget. If not logged in, it uses cookies to recall the last browsed location.

Event Details. Provides a Google Maps mashup of the event location. It may play an MP3 file uploaded by the manager related to the event. External links are opened in new windows. From here the visitor may purchase a ticket as described in Section 4.6.

4.3 User Pages

Recommended Events. Displays events the application recommends to the user, in a manner similar to *Browse Events*. This page provides an external API to retrieve a certain users' recommended events, using a token key as authentication. The events are selected both by user recommendations and the system. It provides a drag-and-drop interface to delete unwanted recommendations. If offline, the deletions are saved and submitted once the application is online again. Once per week, the application sends out an automated e-mail to the user, listing new event recommendations.

Recommend. Allows a user to recommend any event to another user or e-mail address. They may enter in multiple targets. If the e-mail address exists on the system and is not currently a friend of the user, a friend request is sent as well. If the target user is currently logged into the application, the recommendation is displayed in a popup window as well on the targets machine.

Event Chat. A simple interactive chat popup window, which provides rudimentary communication between users. Visitors do not have to be currently logged in, but Users may add other Users as *friends*. It uses *script.aculo.us*[8] for event chat transitions.

Friends. Allows the user to view, add and remove friends.

Browse Tickets, View Ticket. Allows the user to browse and view previously purchased event tickets.

4.4 Manager Pages

After logging in conventionally, managers may authenticate themselves additionally using OpenID [17] before any changes are applied.

Create New. The manager may upload an MP3 file for the event. The event description is presented in a rich text editor, and the venue may be selected using an auto-completed text field. This page provides an external API to schedule new events, using a token key as authentication.

[8] A Javascript library providing rich object transitions and interactions: `http://script.aculo.us`.

Your Events, View Event, Edit Event. The manager may view the events they have added, and edit them with an interface similar to *Create New*.

Upload CSV. The manager may upload multiple events in one file, according to a CSV format.

Setup Data Sources. The manager may set up an RSS feed for new events. This RSS feed is checked daily for new events, which are then imported into the system. The manager may specify an additional OpenID authentication server.

4.5 Administrator Pages

This section is restricted to administrators only, allowing them to modify the core content of the site.

List/Edit Events/Users. Lists the events or users on the system, and the administrator may edit their properties or remove them.

Site Stats. Displays a simple overview of the traffic statistics for the site in a client-side application. Updates the fragment identifier while browsing between dynamic pages, which allows the administrator to bookmark the application state.

Contact User. The administrator may send one or many users an e-mail or text message.

4.6 Ticket Booking Process

This process is described graphically in Figure 2. This operates as a client-side application that loads additional scripting to define the display of the venue details, and the user cannot navigate using their back button or history. Before the user reaches this, the user agent is checked for compatibility; if it cannot execute client-side scripting, the user is redirected to a separate static version of the booking process which is functionally similar, but lacks the richness of the client-side user interface.

The application displays a warning message if it takes longer than one minute to load the application. It has a three-minute client-side timeout, and a fifteen-minute server-side timeout; the current timeout situation is displayed in real time, along with the current connection status. If the client-side timeout occurs, the user is warned that the connection has failed; if the server-side timeout occurs, the booking process is cancelled and any reserved seats released. Input validation occurs both on the server and the client, as well as asynchronously in the background, with the client displaying errors in modal popup boxes on failed client-side validation.

Select Seats. Provides a graphical interface to select seats in the venue, displaying a map of the venue and clickable seat regions. If whilst selecting seats

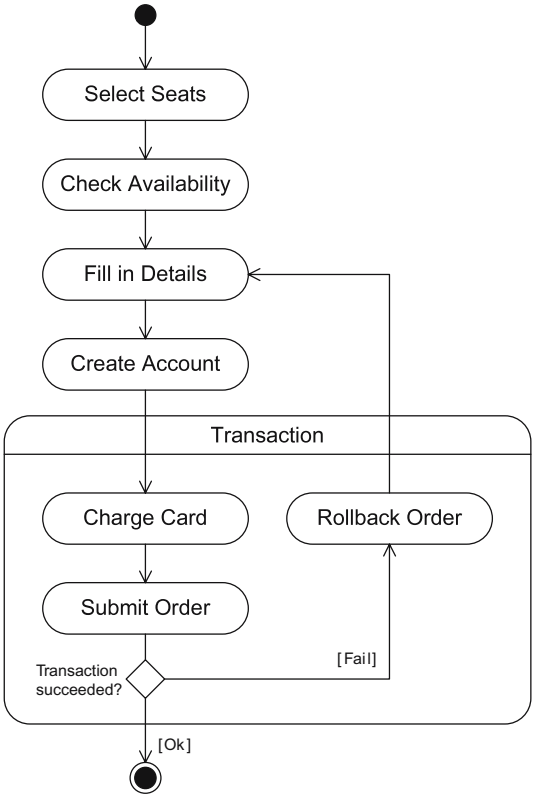


Fig. 2. Ticket 2.0 Booking Process

a seat region is expended on the server, all clients are notified and the region disabled on their displays.

Check Availability. After successfully finding some seats in the desired region, the seats are booked. These are reserved until the user completes the process or the server timeout occurs.

Fill in Details, Create Account. The user enters in the details for purchasing the tickets, and if the user is not currently logged in, they are asked to create an account, or authenticate themselves.

Charge Card. Interacts with an external credit card billing provider, charging the user for the tickets. If either the card charging or order submission steps fail, the transaction is rolled back and the user is asked to confirm their details.

Submit Order. Once an order is submitted, the user is e-mailed a confirmation, and a message is also sent to their mobile phone. The tickets are printed to PDF on the server, which are posted manually every morning by the printing office.

5 Discussion

These proposed requirements fit in well with existing discussions of Rich Internet Applications and hypermedia systems. Precadio et al. [4] propose ten RIA requirements, and finds that both web modelling languages and hypermedia languages are functionally lacking. Similarly, Christoudoulou et al. [10] present fifty hypermedia methodology requirements, and Gu et al. [18] propose sixteen specific web application requirements; many of these relate to the development process. While space limitations prevent us from including a full comparison of our work with existing requirements, we do summarise in Table 3 how our requirements match with those proposed by Precadio et al [4].

We do note that some of our requirements, such as keyboard shortcuts, automatic user authentication, client timer support and browser identification do not yet match up with any previously published requirements; we suggest that these are the new functional properties of RIAs that are currently neglected by existing approaches.

As mentioned in the beginning of this paper, much work has been accomplished in identifying the technical and methodological requirements of web modelling languages. Whilst identifying the functional requirements in this paper, some common themes emerged of these other requirements, such as the importance of a CASE tool; the ability to model using patterns; platform independence; integration with the business model; and management of the development lifecycle [10,18]. Gitzel et al. [7] adds an excellent discussion on the unique non-functional requirements of web applications, such as consistency and predictability. We acknowledge that while expressiveness in a modelling language is vital, it must satisfy these additional requirements in its implementation to be successful [13].

With respect to the ability of existing modelling languages to fulfill these requirements, previous work has already shown that no existing language is expressive enough for RIAs [4,10,14], so we omit such an evaluation in this paper. Future research in this area includes work on extending existing languages in order to address their shortcomings.

Table 3. Comparison of Requirements Proposed by Precadio et. al [4]

Feature	**Matching Requirements**
Interaction	E4, U2, U3, U5
Multimedia	A8
Tool CASE	*n/a*
Visual continuity	A5, U2, U8
Synchronization	E2, E3
N-Tier development	*n/a*
Dynamic data retrieval	D7, D12
Parallel requests to different sources	D6, E4
Personalisation	A3, T4, S9
Interactive collaboration	E7, E9

Table 4. Matching Ticket 2.0 Features to RIA Requirements

Feature	**Requirements Fulfilled**
Application Properties	
Entire Application	D2 D3 E3 S1 S2 S4 S5 T4 T5 U1 U2 U12
Offline	D7 D8
Automated	D6 E1 E10 T1 T2
Mobile	A3 A4 E1
Public Pages	
Home	D1
Signup	E4-6 T1
Login	D13 S1 S2
Browse Events	A8 A9 D4 D5 D13 U5 U11
Event Details	A2 A8 U6 S9
Book Ticket	A1 A5-7 D6 D11 D12 E2-6 E9 E10 T1 T3 U4 U8-10
Recommend	T1 U8 S9
Event Chat	D11 D12 E7-9 U7 U8
Users Only	
Logout	S3
Recommended Events	A8 A9 D4 D5 U3 U5 U8 U11
Friends, Your Tickets, View Ticket	S6
Managers Only	
Your Events	D4 S6 S7 S8
Upload CSV	D10
Setup Data Sources	D6
View Event	A2 A8 U6
Create New	D9 E1 E4-6
Edit Event	E1 E4-6
Administrators Only	
Site Stats	A5 A10 D11 S6 S7
List Events/Users	D4
Edit Event/User	E5 E6
Contact User	T1

In the web application domain, previous benchmarking applications have included conference management systems [19], travel agencies [20] and movie databases [21]. Our contribution is a web application which is clearly aligned with the industry interests of interactive web applications, and specifically fulfills the requirements of RIAs (Table 4).

Other than simply implementing a benchmarking application, it is important to develop metrics to enable quantitative comparisons. This would allow a more precise comparison of different methodologies, and hopefully focus research efforts on addressing these real-world requirements. A discussion of suitable metrics is beyond the scope of this paper, but existing projects such as *Tukutuku* [22] should provide a source of inspiration.

Since this paper was submitted, we have successfully implemented the *Ticket 2.0* application using the Symfony framework for PHP. The implementation of this benchmark has already pointed out some interesting challenges that modelling languages would need to address, such as being able to model the intricate details of interacting with Google Maps and rich text editors, and keeping client-side and server-side interfaces synchronised. The next step in this research is to investigate these challenges and use our findings to improve the growing field of web application modelling.

To interact with a demonstration of this application, the interested reader is referred to the project website at `http://openiaml.org/`. This implementation may be considered as a reference for future research in this field. The reader is also encouraged to retrieve a copy of the benchmark specifications and implement them in their development platform of choice.

6 Conclusion

In this paper we have investigated existing real-world applications, and present a comprehensive list of requirements for Rich Internet Applications. We also propose a benchmarking application called *Ticket 2.0* which embodies all of these requirements in a familiar domain. We believe that this approach is an important step in being able to develop modelling approaches to handle the complexities of interactive application development and compare them quantitatively. By basing this benchmarking application in the same domain as existing industry applications, we also believe this contribution will prove favourable for industry discussion and support.

References

1. Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): A Modeling Language for Designing Web Sites. In: Proceedings of the 9th international World Wide Web conference on Computer networks, pp. 137–157. North-Holland Publishing Co., Amsterdam (2000)
2. Koch, N., Kraus, A.: The Expressive Power of UML-based Web Engineering. In: IWWOST 2002, pp. 105–119 (2002)
3. Baresi, L., Colazzo, S., Mainetti, L., Morasca, S.: W2000: A Modelling Notation for Complex Web Applications. In: Mendes, E., Mosley, N. (eds.) Web Engineering, pp. 335–364. Springer, Heidelberg (2006)
4. Preciado, J.C., Linaje, M., Sanchez, F., Comai, S.: Necessity of Methodologies to Model Rich Internet Applications. In: WSE 2005: Proceedings of the Seventh IEEE International Symposium on Web Site Evolution, Washington, DC, USA, pp. 7–13. IEEE Computer Society, Los Alamitos (2005)
5. Millard, D.E., Ross, M.: Web 2.0: Hypertext by any other name? In: HYPERTEXT 2006: Proceedings of the seventeenth conference on Hypertext and hypermedia, pp. 27–30. ACM, New York (2006)
6. Lang, M., Fitzgerald, B.: Hypermedia Systems Development Practices: A Survey. IEEE Software 22(2), 68–75 (2005)

7. Gitzel, R., Korthaus, A., Schader, M.: Using established Web Engineering knowledge in model-driven approaches. Sci. Comput. Program. 66(2), 105–124 (2007)
8. Taylor, M.J., McWilliam, J., Forsyth, H., Wade, S.: Methodologies and Website Development: A Survey of Practice. Information & Software Technology 44(6), 381–391 (2002)
9. Selmi, S.S., Kraïem, N., Ghézala, H.H.B.: Toward a Comprehension View of Web Engineering. In: Lowe, D.G., Gaedke, M. (eds.) ICWE 2005. LNCS, vol. 3579, pp. 19–29. Springer, Heidelberg (2005)
10. Christodoulou, S.P., Styliaras, G.D., Papatheodrou, T.S.: Evaluation of Hypermedia Application Development and Management Systems. In: HYPERTEXT 1998: Proceedings of the ninth ACM conference on Hypertext and hypermedia, pp. 1–10. ACM, New York (1998)
11. Giurca, A., Wagner, G.: Rule Modeling and Interchange. Symbolic and Numeric Algorithms for Scientific Computing. In: SYNASC. International Symposium, September 26-29, pp. 485–491 (2007)
12. Wampler, D.: Cat Fight in a Pet Store: J2EE vs .NET. Technical report, ONJava.com (2001)
13. Preciado, J.C., Linaje, M., Sanchez, F., Comai, S.: Hypermedia Systems Development: Do We Really Need New Methods? In: IS 2002: Proceedings of the Informing Science + IT Education Conference, Cork, Ireland (2002)
14. Wright, J., Dietrich, J.: Survey of Existing Languages to Model Interactive Web Applications. In: Proceedings of the Fifth Asia-Pacific Conference on Conceptual Modelling (APCCM 2008), Wollongong, NSW, Australia (2008)
15. Nielsen, J.: Hypertext and hypermedia. Academic Press Professional, Inc., San Diego (1990)
16. Ankolekar, A., Krötzsch, M., Tran, T., Vrandecic, D.: The Two Cultures: Mashing up Web 2.0 and the Semantic Web. In: WWW 2007: Proceedings of the 16th international conference on World Wide Web, pp. 825–834. ACM Press, New York (2007)
17. Recordon, D., Reed, D.: OpenID 2.0: a platform for user-centric identity management. In: DIM 2006: Proceedings of the second ACM workshop on Digital identity management, pp. 11–16. ACM, New York (2006)
18. Gu, A., Henderson-Sellers, B., Lowe, D.: Web Modelling Languages: The Gap Between Requirements and Current Exemplars. In: AusWeb 2002: Proceedings of the eighth Australian World Wide Web conference (2002)
19. Schwabe, D.: A Conference Review System. In: First International Workshop on Web-Oriented Software Technology (2001)
20. MDWE 2005 Workshop: The Travel Agency System Example. Technical report (2005)
21. van der Sluijs, K., Houben, G.J., Broekstra, J., Casteleyn, S.: Hera-S: Web Design using Sesame. In: ICWE 2006: Proceedings of the 6th international conference on Web engineering, pp. 337–344. ACM, New York (2006)
22. Mendes, E., Martino, S.D., Ferrucci, F., Gravino, C.: Cross-company vs. single-company web effort models using the Tukutuku database: An extended study. Systems and Software (2007)

SyncRerank: Reranking Multi Search Results Based on Vertical and Horizontal Propagation of User Intention

Satoshi Nakamura, Takehiro Yamamoto, and Katsumi Tanaka

Department of Social Informatics, Graduate School of Informatics,
Kyoto University
Yoshida-Honmachi, Sakyo, Kyoto 606-8501, Japan
{nakamura,tyamamot,tanaka}@dl.kuis.kyoto-u.ac.jp

Abstract. This paper proposes novel interaction techniques for parallel search tasks. The system displays multiple search results, returned by a search engine, side-by-side for each search query. The system enables the user to rerank search results using a reranking algorithm based on vertical and horizontal propagation of his/her intention. A method of recommending operations for specific keywords is also proposed, supporting operations such as a shift to a parallel search with an alternate term, upgrading or downgrading results in terms of a specific viewpoint, and so on. Applications for the proposed system are also discussed.

1 Introduction

Many people now use Web search engines to obtain information in their daily life. It remains unclear, however, whether conventional Web search engines truly satisfy users' requirements and are useful.

For example, when a user is thinking about buying a car, he/she wants to compare car companies, models, options, and so on before making the purchase. In this case, he/she makes a query for a Web search and inputs the name of a company, brand, or car, along with keywords representing viewpoints such as price, quality, and fuel consumption. Then, he/she checks many search results or tries other queries many times before finding the right car. If the user wants to compare multiple cars simultaneously, he/she has to open many browser windows.

As another example, when a user wants to cook dinner using green pepper, pork, and something in his/her refrigerator, he/she has to choose the ingredients and input them as a query many times before discovering an appropriate recipe. If he/she wants to view images of dishes, he/she has to find pages that contain such images.

Finally, when a Japanese user wants to find work related to his/her research by using digital libraries, he/she has to search by using the *ACM digital library*, *IEEE Xplore*, and Japanese digital libraries, using multiple related keywords in different languages. This requires performing similar search processes with each system.

These tasks are not easy for users because conventional systems are not particularly efficient at carrying out such work. The goal of our current work is to enable users to perform such tasks easily. If a user could search for products from different

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 120–135, 2008.

viewpoints simultaneously, he/she could easily check a large amount of information. For example, if the user could search text and images at the same time, he/she could easily pick a recipe for a dish. Similarly, if the user could search Japanese and English digital libraries at the same time, he/she could find related work more easily.

To satisfy such user requirements and develop appropriate applications, we propose novel interaction techniques for search results. In a past work we proposed "*Rerank-by-Example*" [10], which enables the user to rerank search results using an edit-and-propagate operation. We refer to this propagation style as *vertical propagation*. In the current work, we expand this to *horizontal propagation*. Our proposed system displays multiple search results from a search engine side-by-side for each search query. It then enables users to rerank each set of search results synchronously by *horizontal propagation*.

In this work, because it is difficult for users to input complex queries we also propose a method of interaction recommendation for specific keywords, such as shifting to a parallel search using an alternate term, upgrading or downgrading in terms of a specific viewpoint, and so on. By using our system, the user can easily rerank and check multiple search results according to his/her search intention.

The technical contributions of this paper are as follows:

- Interaction for parallel search and algorithm of synchronized reranking
- Interaction recommendation supporting reranking and a shift to a parallel search

In this paper, we first explain the motivation of our work and the basic idea of our system. Next, we describe the system's algorithm, design, and implementation. We proceed to describe typical applications to show the usefulness of our system and discuss its characteristics. Finally, we conclude this work and explain the direction of our future work.

2 Motivation

In a past work, we used a survey to research users' attitudes toward Web searches [9]. We created an online questionnaire of 26 questions which were answered by 1000 subjects. From this survey, we found that the top three user motivations for using Web search engines were "looking up the meaning of a word," "looking up detailed information about a word," and "making comparisons." In addition, we also found that users wanted to use special search systems.

In a conventional Web search system it is difficult for users to compare products in terms of characteristics such as quality, price, and user evaluations. Some users input "comparison" as a query keyword with the products that they want to compare. In this case, however, the system sometimes does not obtain good pages with detailed descriptions of each product.

In the survey, we also found that more than 50% of the subjects normally checked only the top five search results. This suggests that low-ranked search results are almost never checked by more than half of search engine users. We attribute this to the limited interaction between the user and the search engine. We also found that 23.2% of the subjects wanted to rerank Web search results according to their search intentions

because they were unsatisfied with the Web search result rankings returned by conventional search engines. We believe that reranking systems based on user interaction will become very important in the near future.

In addition, 48.1% of the subjects wanted to use a search engine with additional information to help them understand Web search results. This suggests that users sometimes have difficulty judging the usefulness of each search result or the appropriateness of a submitted query.

From these survey results, we can conclude that people require more useful Web search systems. Such systems should enable them to easily compare items, allow them to rerank search results, and present them with additional information to help them understand search results.

3 Basic Idea

Reranking by Flexible Term-based Relevance Feedback

In a past work [10][1], we proposed a system enabling users to rerank Web search results by an edit-and-propagate operation on the client side (Fig. 1). We refer to this approach as "*Rerank-by-Example*." The system propagates the influence of a user's edit operation throughout all the search results. The system includes two edit operations: emphasize and delete.

When the user selects one part of a search result (i.e., the title, the summary, the URL, etc.), the system displays an edit menu with buttons for emphasis and deletion (Fig. 2). The user then applies his search intention by clicking the appropriate button.

When the user emphasizes a keyword or phrase from a search result part, the system determines that the user wants to upgrade search results containing the emphasized keyword or phrase. When the user deletes a keyword or phrase from part of a search result, the system determines that the user wants to downgrade search results containing the deleted keyword or phrase.

In the specific case of emphasizing or deleting part of the URL of a search result, the system determines that the user wants to upgrade or downgrade, respectively, Web pages for which the URL contains the selected text. For example, the user can easily upgrade government sites by emphasizing".gov" or downgrade blogs by deleting "blog"

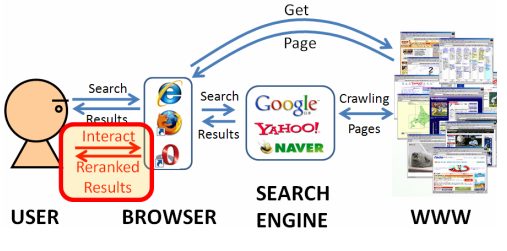


Fig. 1. Interaction between the user and search results

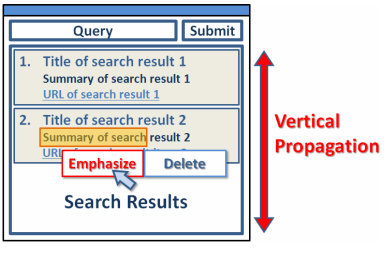


Fig. 2. Edit menu displayed near the search result part selected by the user

[1] http://rerank.jp/

The system calculates a score for each search result according to the target content and the type of edit operation. Then, the system sorts the resulting list of search results and shows the reranked results in the Web browser. Because the system performs these operations on the client side, the results page does not have to be reloaded from the server. We found that users could easily browse 500 search results by using our system.

Parallel Search and Synchronized Reranking by Interaction Propagation

In this work, we extend the idea of *Rerank-by-Example* to parallel searches. When the user inputs queries to a parallel search system, the system obtains search results from a search engine and displays multiple search results side-by-side for each query. For example, when the user wants to compare *A*, *B* and *C*, he/she first inputs *A*, *B* and *C* as queries for a parallel search. Then, the system displays the search results for *A*, for *B*, and for *C* side-by-side. The user can easily check the high-ranked search results for each query, without scrolling the browser window.

In addition, we extend the propagation of reranking search results to all of the lists of search results. For example, when the user emphasizes viewpoint *VP3* (i.e., a specific characteristic of the results) in the search results for *A*, the system synchronously reranks not only the search results for *A* but also those for *B* and *C*. The user can thus simultaneously check each set of reranked search results for *VP3* (Fig. 3), and can then rerank each set of search results according to any other viewpoint, such as *VP1* or *VP2*. In Fig. 4, the user compares *FinePix* with *CyberShot* and emphasizes "zoom" as a view point. Then, the system reranks each of the search results simultaneously.

The user can choose the type of search engine, such as a Web search engine, image search engine, or video search engine. In addition, we allow the user to perform paralel searches in different languages. For example, he/she can input queries for

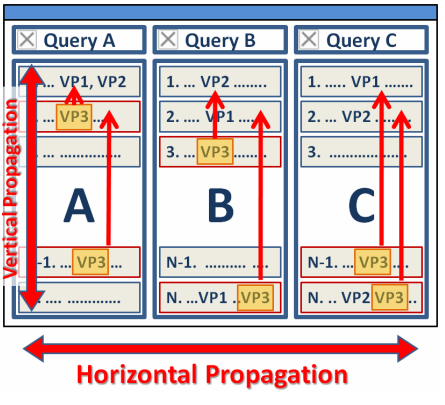


Fig. 3. Synchronous reranking of each set of search results according to the user's edit operation

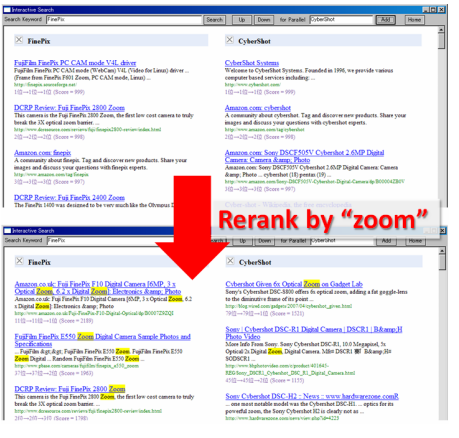


Fig. 4. Example of parallel search and synchronous reranking

parallel searches in Japanese and English to find related work in both languages at the same time. If he/she selects and emphasizes a Japanese term in the Japanese search results, the system translates the emphasized term into English and simultaneously reranks the Japanese results according to the input term and the English results according to the translated term. We expect that users will easily be able to find relevant search results in each language.

Interaction Recommendation

By the way, most search engine users are not advanced users. *Ryen* et al. [7] define an advanced search engine user as one who uses plus and minus signs, double quotes, and special phrases such as "site:" in queries. As calculated from large search query logs, only 8.72% of users are advanced. This suggests that we should design the interaction in our system not only for advanced users but also for non-advanced users. For example, if users have to input complex queries with phrases such as "parallel:" or "compare:", many will not use the parallel search system. We should thus implement a system that non-advanced users will want to use and interact with.

Therefore, we propose modifying the system to recommend operations with terms, such as parallel search with a recommended alternate term for the target query, image search with the target query, reranking by upgrading or downgrading a term, and so on. At the top of the search results, our system shows terms with icons representing operations recommended for the terms (Fig. 5). The user can easily run the recommended operation simply by clicking its icon or text. The user can then check the reranked search results or shift to a parallel search without performing any complex operations.

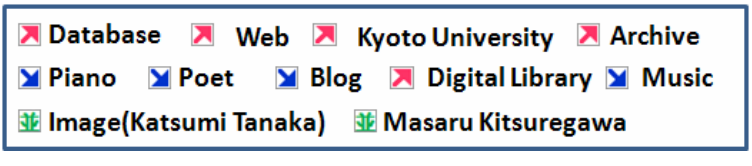


Fig. 5. Examples of interaction recommendation for the query, "Katsumi Tanaka." The red arrow means upgrading search results which include a term, the blue arrow means downgrading search results which include a term. The green character means the parallel search using a term. Image(A) means an image search with query A.

In this interaction recommendation, we take account of the user's search intention. For example, when the user deletes the term "*pianist*" from the search results for "*Katsumi Tanaka,*" the system guesses the user's search intention to be "downgrading search results related to the pianist, Katsumi Tanaka." The system then recommends "*piano,*" "*concert,*" and "*music*" for deletion, and "*professor,*" "*database,*" and "*poet*" for emphasis, enabling the user to easily rerank the search results accordingly.

As another example, when the user emphasizes the term "*database*" in the search results for "*Katsumi Tanaka,*" the system recommends "*Masaru Kitsuregawa*" as an alternate term for a parallel search.

The types of operations for terms are "shift to parallel search with a recommended alternate term," "shift to parallel search with different search engine, language or media," "upgrade search results which include a recommended term" and "downgrade search results which include a recommended term."

4 Algorithm

Interaction Propagation

We also investigated the application of reranking by term-based relevance feedback to a parallel search with synchronized reranking.

When the user inputs a query q for a parallel search, the system obtains the top N search results $R(q) = \{r(q)_1, r(q)_2, \ldots, r(q)_N\}$ (where $r(q)_i$ is the i^{th} search result for query q).

The score for each result is initialized as

$$Score(r(q)_i) = N - i . \tag{1}$$

Assume that the user tries a parallel search with queries q_A, q_B, and q_C. When the user emphasizes or deletes keyword t in the set of results, $R(q_A)$, the system respectively upgrades or downgrades the results including t in $R(q_A)$. It then calculates the score for result $r(q_A)_i$ including t in its title or snippet according to Equation 2:

$$Score(r(q_A)_i)_{new} = Score(r(q_A)_i)_{last} + type \times 2N . \tag{2}$$

The system divides $R(q_A)$ into sets $R^+(q_A)$, whose results include t, and $R^-(q_A)$, whose results do not include t.

Then, the system uses a stop word list to perform morphological analysis on the titles and snippets in $R^+(q_A)$ and $R^-(q_A)$ to obtain the sets of terms $W(R^+(q_A))$ and $W(R^-(q_A))$, respectively, and it calculates the term frequency of term $_{w \in W(R^+(q_A))}$ and $_{w \in W(R^-(q_A))}$. It removes term w_i in $W(R^+(q_A))$ if w_i is in $W(R^-(q_A))$. Next, the system finds the top k most frequent terms in $W(R^+(q_A))$ and $W(R^-(q_A))$. We then use these k terms in $W(R^+(q_A))$ as the co-occurring terms for t.

The system next calculates the score for $r(q_B)_i$ by using the similarities of the co-occurring terms and the target search result, according to Equation 3:

$$\begin{aligned} Score(r(q_B)_i)_{new} = Score(r(q_B)_i)_{last} + type \times Sim(r(q_B)_i, W(R^+(q_A))) \times N . \\ + type \times Exist(r(q_B)_i, t) \times 2N \end{aligned} \tag{3}$$

Here, $Sim(r,W)$ denotes the similarity between the set of frequent terms W and result r. The system only checks the coverage of terms in W by r. If r does not contain any terms in W, then $Sim(r,W)$ is 0. If r contains all the terms in W, then $Sim(r,W)$ is 1. $Exist(r,t)$ is a Boolean value. If r includes term t, $Exist(r,t)$ is 1. If r does not include t, $Exist(r,t)$ is 0. After applying Equation 3, the system reranks all search results according to their recalculated scores. If the operation is emphasis, *type* is 1. If the operation is deletion, *type* is -1.

If the user chooses the Japanese and the English search engines for a parallel search and he/she emphasizes/deletes a phrase in the Japanese search results, the system translates the emphasized/deleted term and co-occurring terms into English by using a translation service.

Interaction Recommendation with a Term for Reranking Search Results
When the user searches with a new query or adds a query for a parallel search, the system obtains the set of top N search results, R, for the input query. Then, after calculating the set of frequent terms W, the system selects the top k recommended terms $u_i \in U$ from W according to the following process.

1. The system obtains the most frequent term w_1 in W.
   (a) If U is empty, the system puts w_1 in U and removes w_1 from W.
   (b) If U is not empty, the system obtains w_2 from W. Then, the system calculates scores for w_1 and w_2 by Equation 4. $DF(c, R)$ denotes the number of search results which satisfy the condition c in search results R. If $Score(w_1)$ is greater than or equal to $Score(w_2)$, the system puts w_1 in U and removes w_1 from W. Otherwise, it puts w_2 in U and removes w_2 from W.

$$Score(w_i) = DF(w_i, R) - DF(w_i \,\&\, (u_1 \mid u_2 \mid ... \mid u_n), R) \tag{4}$$

2. If the number of terms in U is less than k, the system repeats step 1. Otherwise, the process is finished.

Next, the system randomly sets the operation type for each extracted term. The reason of setting the operation type randomly for them is that the system has no information about the user's interest before interaction with the system begins.

When the user emphasizes or deletes term t from the search results, the system divides the result set R into sets R^+, whose results include t, and R^-, whose results do not include t. The system also calculates the corresponding sets of frequent terms, $W(R^+)$ and $W(R^-)$. Then, it checks the top k most frequent terms in $W(R^+)$ and $W(R^-)$. If term w exists in both $W(R^+)$ and $W(R^-)$, the system removes w from $W(R^+)$ and $W(R^-)$.

If the user's operation is emphasis, the system shows terms in $W(R^+)$ with the emphasis icon and terms in $W(R^-)$ with the deletion icon. If the user's operation is deletion, the system shows terms in $W(R^+)$ with the deletion icon and terms in $W(R^-)$ with the emphasis icon.

Interaction Recommendation for a Shift to a Parallel Search
There are two types of shift to a parallel search. One is a shift to a parallel search with a different search type, such as different media, a different search engine, or a different language. The other is a shift to a parallel search with a recommended term, because users sometimes have difficulty finding alternates to the query target.

The recommendation for a shift to a parallel search with a different search type is very straightforward. The system simply recommends the appropriate search type with the current query. For example, the system recommends an image search with "*Auckland*" when the user does a Web search with "*Auckland*." The system usually recommends different media or a different search engine because most users do not use multiple languages in searching.

When the system recommends a shift to a parallel search with a recommended term, it first finds alternate names. Many methods have been proposed for finding alternate candidates for a target object [7, 14]. Ohshima's method [7] simply sends the user's query term with a conjunction "OR" to obtain the alternate terms. Then the frequent adjacent terms to the user's query term are obtained as alternate terms from the returned search results. This method also enables the system to add the background term to obtain alternative terms accurately. In the current work, we use this method for parallel searches. We can use the emphasized or deleted term as the background term.

In addition, we use emphasized or deleted terms to find alternates because many different user intentions are possible in the interaction between the user and the system. For example, when a user emphasizes "*professor*" in the search result "*Katsumi Tanaka*," the system guesses that he/she might want to find the name of an alternate person who is a professor.

5 Design, Implementation and Interaction

Design and Implementation

Our system consists of the browser interface module, the monitoring module, the recognition module, the evaluation module and the reconstruction module (Fig. 6).

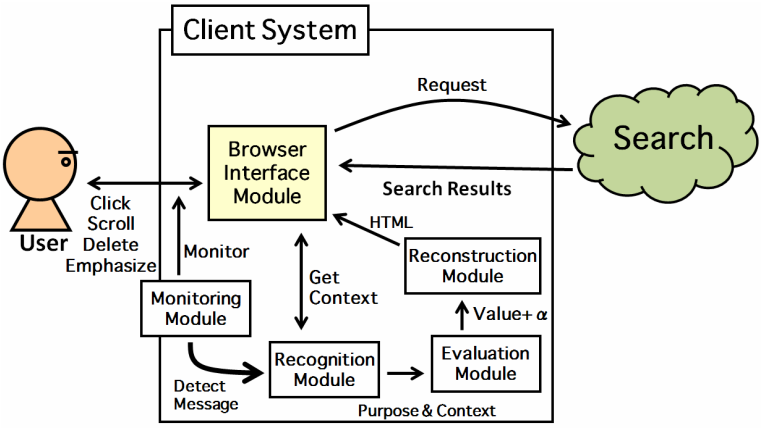


Fig. 6. System consists of five modules

The browser interface module supports all normal Web browser functions and allows the user to emphasize or delete any part of a search result at any time. In addition, the browser interface module renders HTML pages from the reconstruction module via memory, making it possible to render and modify pages without a refresh.

The monitoring module monitors the user's mouse operations to identify what text is selected. If text is selected, it automatically displays the edit menu which consists of the emphasis and deletion buttons, near the current mouse position. If the user does not move the cursor for a certain time after the edit menu appears, the edit menu is automatically hidden. When the monitoring module detects an edit operation from the

user clicking a button or a recommendation icon, it sends the edit information to the recognition module.

After receiving the user's edit information, the recognition module detects the user's purpose by checking the edit type, selected text and blocks. It then determines the user's purpose from the emphasized or deleted part of a search result. Finally, it sends the collected information to the evaluation module.

After the evaluation module receives the user's purpose from the recognition module, it recalculates the scores of all search results using Equations 1, 2, 3 and 4. In addition, it also obtains recommendations of interaction based on the above-referenced algorithms. Then it sends the recalculated results and interaction recommendation to the reconstruction module.

The reconstruction module finally reconstructs the Web page by using the recalculated scores sent by the evaluation module. It sorts the list of search results according to the information from the evaluation module and generates the HTML for rendering by the browser interface module. It also puts interaction recommendations at the top of the search results.

We implemented our system by using *Microsoft Visual C++ .NET 2005*, *Google Web Search API*[2], *Yahoo! Web/Image/Video Search API*[3], and the library of *Mecab*, a Japanese morphological analyzer. In addition, our system uses the Web API of Japanese-English/English-Japanese dictionary powered by *EAST Co. Ltd.*[4], for translation. Additionally, we implemented the analyzer used for digital library search services and so on.

Interaction

In our system, the user first inputs a query by using a text box located at the upper left of the browser window. The system then obtains search results for the input query. It incrementally displays the results according to the data that it receives. Then it analyzes all the results and displays recommended operations for keywords at the top of the results.

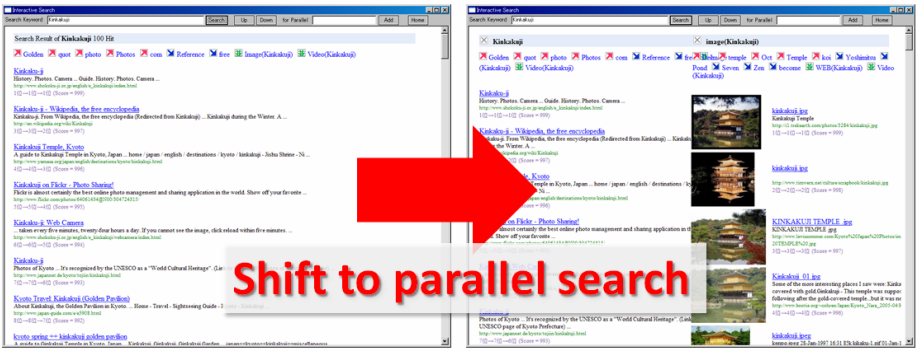


Fig. 7. Shift from single search to parallel search

[2] http://code.google.com/
[3] http://developer.yahoo.co.jp/search/
[4] http://www.btonic.com/ws/

The user normally reads the returned search results. If he/she finds a good result in the high-ranked results, he/she typically visits the linked page. If the user cannot find a good result, he/she reranks the results by clicking a recommended operation or by directly emphasizing or deleting results. If the user still cannot obtain satisfactory results, he/she re-creates the search query.

When the user wants to shift to a parallel search, he/she checks the recommended operations for parallel search. If a good operation is recommended, he/she can shift to a parallel search simply by clicking the recommended operation (Fig. 7).

If there are no good recommendations for a parallel search, the user can instead shift to a parallel search by inputting a keyword in a text box located at the upper right of the browser window. If the user wants to perform a parallel search with images or video, he/she can change the search mode by selecting a different item in a list box located to the right of the text box for parallel search.

In a parallel search mode, when the user reranks the search results by emphasizing or deleting a part of a search result, the system recalculates the score of each search result and displays the reranked search results. The user can also click a recommended operation in a parallel search mode. To exit a parallel search mode, the user clicks the close button located at the top of the search results.

6 Application

We can classify parallel search tasks into comparative search and meta-search.

6.1 Comparative Search

People often make comparisons before traveling, booking a hotel, starting a job search, and so on. There are two types of comparative search tasks:

- The user gives entities for comparison
- The user wants to find an entity from a group of elements

In the first task, the user starts by inputting entity names, such as "*FinePix*" and "*CyberShot*" to compare cameras, or "*Toyota*" and "*Honda*" to compare cars. The search results are then displayed side-by-side under the entity names.

In this task, the user typically reranks the results in terms of viewpoints such as quality, price, usefulness, durability, and so on. The user can thus easily check the results from a different viewpoint by reranking the results. In this task, the user sometimes cannot remember the competitor's names such as "*EXILIM*" and "*Nissan*." Our recommendation system provides such names for users.

In the second task, the user does not know or cannot decide which entity name to use. There might be many variations of an entity name. The entities from which the user can choose, however, may be limited by certain elements that he/she can select. There are patterns for selecting elements, but the user cannot use all of them and is torn between element *A*, element *B*, and other elements.

For example, when the user knows he/she has pork and beef in the refrigerator and wants to cook dinner using one or the other with something else, he/she is sometimes torn between using pork and using beef. In this case, the user first inputs each element

name with something else as a query, such as “pork and potato” or “beef and potato.” Next, he/she checks the search results and reranks them by deleting elements that his/her family dislikes or by emphasizing elements representing foods that are in the refrigerator. Then, he/she decides what to cook by comparing the pork and beef dishes obtained in the reranked results (Fig. 8).

For example, when the user is suffering from a physical disorder, he/she wants to make a tentative diagnosis himself/herself based on the symptoms before going to hospital. In this case, the user first inputs each element name as a query, such as “headache and dry eye” or “headache and hand pain.” Next, he/she checks the search results and reranks them by emphasizing elements representing a symptom that he/she identifies. Then, he/she decides what to do by comparing the sicknesses mentioned in the reranked results.

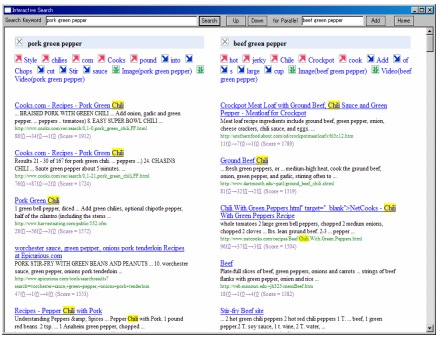


Fig. 8. Comparison between pork and beef dishes simultaneously

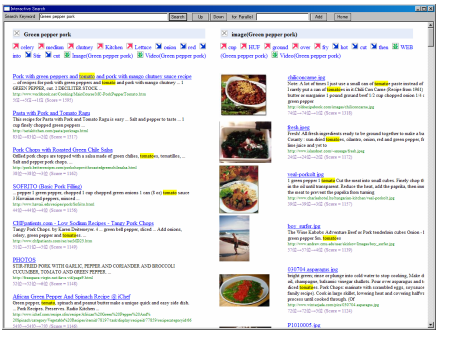


Fig. 9. Comparing the results of a text and image search simultaneously

6.2 Meta-search

We can divide meta-search tasks depending on whether they use the same media or different media. Meta-search with the same media can be further classified into meta-search with different search engines, such as *Google* and *Yahoo!*, and with different languages, such as Japanese and English.

The difference between *Google* and *Yahoo!* search results is not small, particularly for image searches. A user can save time by performing both searches at the same time.

When a user wants to survey published papers which are related to his/her research, he/she has to check many digital library services in different languages. In this case, if the system enables the user to check results from different services and different languages such as *ACM digital library*[5], *IEEE Xplore*[6], *Google Scholar*[7], *CiNii*[8] and so on simultaneously, time will be saved. As another example, a user can apply a meta-search with different languages to find information in each language about a

[5] http://portal.acm.org/dl.cfm
[6] http://ieeexplore.ieee.org/
[7] http://scholar.google.com/
[8] http://ci.nii.ac.jp/

travel destination or to find patent information by using Japanese, American and European patent information retrieval systems in each language.

For a meta-search with different media, the system applies a text search and an image search normally. This search style is useful for finding recipes and travel information.

For example, sometimes a user needs a good idea for cooking. He/she knows what is in the refrigerator and what he/she or his/her family likes and dislikes. In this case, the user can first input the names of elements such as "green pepper and pork." After checking the initial search results, he/she can rerank them by deleting elements that he/she or his/her family dislikes or by emphasizing elements representing foods that are in the refrigerator. In this task, a meta-search using both a text and an image search with the same query supports the user's intention because the image results can spark his/her imagination (Fig. 9).

A meta-search can sometimes help the user understand something. For example, when a user searches for "*Katsumi Tanaka*" with a text search engine the target person might be unclear because there are many people whose name is "*Katsumi Tanaka*" in the world and on the Web. Here, if results from an image search engine showing pictures of different people called "*Katsumi Tanaka*" are added, the user can better determine the correct one. We can say that the results of the image search fill the gap between the user's memory of information and memory of images. This search style has the potential to evolve into a complementary search.

7 Discussion

We distributed our system on the Internet and collected comments from users. As a result, we found that our system is useful for comparing products before buying, for finding recipes, and for making tentative medical diagnoses. In particular, when the user finds a recipe through a meta-search using both a text and an image search with the same query, he/she was able to decide what to cook easily by switching the focus between image search results and text search results.

As described above, the recommendation of operations is useful, enabling users to easily rerank both desired and unwanted search results. The user can also shift to a parallel search simply by clicking a recommended search query or search type. By recommending terms co-occurring with the emphasized or deleted terms in the search results, the user could upgrade or downgrade related search results that did not contain an emphasized or deleted term.

In general, users often have difficulty finding viewpoints for comparison. Our system, however, shows viewpoints in the recommended operations at the top of the search results. The user can thus easily check reranked search results in terms of different viewpoints by clicking the recommended operations.

In our past work [18], we proposed the *inner extended keyword method* and the *outer extended keyword method*. These methods are not particularly useful, however, because the system sometimes makes mistakes in obtaining extended keywords. As a result, it downgrades useful search results or upgrades useless results. In contrast, the current system has no such problems because it simply recommends keywords to users.

On the other hand, we could not check the usefulness of meta-search in different languages and different search engines because we implemented *Google*'s search services and *Yahoo*'s search services only. In the future, we will implement search services such as Japanese digital library services, English digital library services, booking services and so on to show the usefulness of our system.

One problem with our system is the high directivity of current recommendations resulting from the user's previous operation. When the user deletes "*pianist*" from the search results for "*Katsumi Tanaka*," the system recommends "*piano*," "*concert*," and so on. After that, when the user deletes "*piano*" from the results, the system recommends "*concert*" and "*music*." In this series of recommendations, the user has no chance to change the search direction. This is not good for opportunistic searches.

In our prototype system, when the user adds a query for a parallel search, the system does not take account of the user's previous operation in regard to reranking. The user then has to rerank the added search results in the same way. If the system automatically reranked results according to the previous user interaction, parallel searches would be more convenient for the user.

8 Related Work

There are many studies related to a user's search intention. Broder [1] and Rose and Levinson [6] have classified the goals of a user searching the Web into three categories: navigational, informational, and transactional/resource. The purpose of an informational search is to find information about the topic of the user's query. There are also studies on automatic classification of user goals according to queries [8, 20]. Our system is useful for informational search and transactional/resource searches.

There are many existing reranking methods. Relevance feedback [10] is the most popular retrieval system for retrieving documents according to user feedback. Relevance feedback is based on a vector space model [11]. Non-relevance feedback [17] is another retrieval method for documents. It uses only non-relevant documents to find target documents. Tan et al. proposed term-based relevance feedback algorithms [4]. Our method is similar to the term-based relevance feedback method but there are some differences between conventional systems and our system. First, our system enables the user to rerank search results by direct manipulation, whereas in a conventional system, the user only reranks results through indirect manipulation, such as checking the box for a recommended term. A second difference is flexibility. Our system enables users to vary the level of detail for an editing target, such as a word, sentence, or part of a URL, as well as to select specific attributes of the target, such as the title, snippet, or URL.

Yahoo! Mindset [21] allows the user to specify his search intentions to the system. It weights each Web page as "more research" or "more shopping." The user can specify his search intention as research or shopping by using a slide bar, after which the system reranks the search results according to the weight. In this system, however, the user cannot rerank search intentions without prepared factors.

Huynh et al. proposed a system and interfaces enabling users to sort or filter items in Web pages that are created automatically from templates [5]. This system starts by analyzing the structure of a Web page and then extracts the attributes of items on the

page after the user clicks a start button. The user can sort or filter items according to these attributes. The user cannot, however, interact with the items directly and cannot rerank or classify items. In addition, this system does not focus on comparisons.

Comparative Web Browser (*CWB*) [2] is a Web browser with two browser windows. *CWB* displays two Web pages concurrently in a way that enables the content of the pages to be automatically synchronized. *CWB* can be implemented through parametric control by assigning the viewing position of one Web page to a parameter that synchronizes with the position of the other Web page. *B-CWB* [3] is an extension of *CWB*. This system also has two browser windows but shows two similar news articles written in different languages. These systems are useful for checking the differences between two sets of content. Such systems, however, do not focus on browsing Web search results and reranking results.

The system developed by *Yumoto* et al. [19], and the system named *Comparative Web Search* (*CWS*) [9] enable users to compare two objects by analyzing pages obtained from names of these objects. The motivation of their work is similar to our work. However, these systems do not consider interaction between the user and the search results. In addition, they do not support searches with different search engines, language and media.

There are some studies that use cross media. WebTelop [12] is a system that searches for pages related to a TV program that the user is viewing. The system automatically displays a page complementing the TV program. In this system, the user cannot change the complementary pages by his/her search intention or view point. In contrast, our system displays complementary/relative search results synchronously and enables the user to rerank search results according to his/her viewpoint by an edit-and-propagate operation.

9 Conclusion

In this work, we have proposed a parallel search system and algorithms to support this system. Then, we have shown the usefulness of our system by describing its applications.

The contributions of our work are the parallel search with synchronized reranking by propagating an edit operation to other search results, and interaction recommendation with keywords/search types. We believe that these contributions will have an important impact on tasks related to information finding.

In the future, we plan to improve the ranking algorithm. We also plan to find a good method for obtaining alternate terms, viewpoints, and so on because we use a simple method to obtain such terms in the current work. For example, we expect that we will be able to obtain alternate terms by analyzing query logs.

We have implemented our system as an application with no other functions. In the future, we plan to implement our system as a Mozilla Firefox extension in order to increase the number of users.

In this work, we have only focused on editing text. In the future, we plan to try image-based relevance feedback. For example, the system will enable the user to rerank search results according to the shape of an object in an image. We could also apply this idea to video content. To realize such systems, we can use the idea of zooming cross-media [16] because this allows users to browse different media, such as text, images, and video depending on the zoom level.

Acknowledgements

This work was supported in part by MEXT Grant-in-Aid for Scientific Research on Priority Areas: "Cyber Infrastructure for the Information-explosion Era", "Contents Fusion and Seamless Search for Information Explosion" (Project Leader: Katsumi Tanaka, A01-00-02, Grant#: 18049041), and by "Design and Development of Advanced IT Research Platform for Information" (Project Leader: Jun Adachi, Y00-01, Grant#: 18049073), and by "Informatics Education and Research Center for Knowledge-Circulating Society" (Project Leader: Katsumi Tanaka, MEXT Global COE Program, Kyoto University). This work was also supported in part by the National Institute of Information and Communications Technology.

References

1. Broder, A.: A taxonomy of web search. ACM SIGIR Forum 36(2), 3–10 (2002)
2. Nadamoto, A., Tanaka, K.: A Comparable Web Browser (CWB) for Browsing and Comparing Web Pages. In: Proc. of WWW 2003, pp. 727–735 (2003)
3. Nadamoto, A., Ma, Q., Tanaka, K.: Concurrent Browsing of Bilingual Web Sites by Content-Synchronization and Difference-Detection. In: Proc. of WISE 2003, pp. 189–199 (2003)
4. Tan, B., Velivelli, A., Fang, H., Zhai, C.: Term Feedback for Information Retrieval with Language Models. In: Proceedings of SIGIR 2007, pp. 263–270.
5. Huynh, D., Miller, R., Karger, D.: Enabling web browsers to Augment Web Sites' Filtering and Sorting Functionalities. In: Proceedings of UIST 2006, pp. 125–134 (2006)
6. Rose, D., Levinson, D.: Understanding User Goals in Web Search. In: Proceedings of WWW 2004, pp. 13–19 (2004)
7. Ohshima, H., Oyama, S., Tanaka, K.: Searching Coordinate Terms with their Context from the Web. In: Aberer, K., Peng, Z., Rundensteiner, E.A., Zhang, Y., Li, X. (eds.) WISE 2006. LNCS, vol. 4255, pp. 40–47. Springer, Heidelberg (2006)
8. Kang, I., Kim, G.: Query type classification for web document retrieval. In: Proc. SIGIR 2006, pp. 64–71 (2006)
9. Sun, J.T., Wang, X., Shen, D., Zeng, H.J., Chen, Z.: CWS: a Comparative Web Search System. In: Proceedings of WWW 2006, pp. 467–476 (2006)
10. Salton, G.: The SMART Retrieval System Experiments in Automatic Document Processing, pp. 312–323 (1971)
11. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw-Hill, New York (1983)
12. Ma, Q., Tanaka, K.: WebTelop: Dynamic TV-Content Augmentation by Using Web Pages. In: Proceedings of ICME 2003, II, pp. 173–176 (2003)
13. White, R.W., Morris, D.: Investigating the querying and browsing behavior of advanced search engine users. In: Proceedings of SIGIR 2007, pp. 255–262 (2007)
14. Wang, R.C., Cohen, W.: Language-Independent Set Expansion of Named Entities using the Web. In: Perner, P. (ed.) ICDM 2007. LNCS (LNAI), vol. 4597. Springer, Heidelberg (2007)
15. Nakamura, S., Konishi, S., Jatowt, A., Ohshima, H., Kondo, H., Tezuka, T., Oyama, S., Tanaka, K.: Trustworthiness Analysis of Web Search Results. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 38–49. Springer, Heidelberg (2007)

16. Yamamoto, T., Nakamura, S., Tanaka, K.: Rerank-By-Example: Efficient Browsing of Web Search Results. In: Wagner, R., Revell, N., Pernul, G. (eds.) DEXA 2007. LNCS, vol. 4653, pp. 801–810. Springer, Heidelberg (2007)
17. Yumoto, T., Tanaka, K.: Page Sets as Web Search Answers. In: Sugimoto, S., Hunter, J., Rauber, A., Morishima, A. (eds.) ICADL 2006. LNCS, vol. 4312, pp. 244–253. Springer, Heidelberg (2006)
18. Araki, T., Miyamori, H., Minakuchi, M., Kato, A., Stejic, Z., Ogawa, Y., Tanaka, K.: Zooming Cross-Media: A Zooming Description Language Coding LOD Control and Media Transition. In: Andersen, K.V., Debenham, J., Wagner, R. (eds.) DEXA 2005. LNCS, vol. 3588, pp. 260–269. Springer, Heidelberg (2005)
19. Onoda, T., Murata, H., Yamada, S.: Non-Relevance Feedback Document Retrieval Based on One Class SVM and SVDD. In: 2006 IEEE World Congress on Computational Intelligence, pp. 2191–2198 (2006)
20. Lee, U., Liu, Z., Cho, J.: Automatic identification of user goals in Web search. In: Proc. of WWW 2005, pp. 391–400 (2005)
21. Yahoo! MindSet, `http://mindset.research.yahoo.com/`

Web-Based Measure of Semantic Relatedness

Jorge Gracia and Eduardo Mena

IIS Department, Univ. of Zaragoza, María de Luna 1, 50018 Zaragoza, Spain
{jogracia,emena}@unizar.es
http://sid.cps.unizar.es

Abstract. Semantic relatedness measures quantify the degree in which some words or concepts are related, considering not only similarity but any possible semantic relationship among them. Relatedness computation is of great interest in different areas, such as Natural Language Processing, Information Retrieval, or the Semantic Web. Different methods have been proposed in the past; however, current relatedness measures lack some desirable properties for a new generation of Semantic Web applications: maximum coverage, domain independence, and universality.

In this paper, we explore the use of a semantic relatedness measure between words, that uses the Web as knowledge source. This measure exploits the information about frequencies of use provided by existing search engines. Furthermore, taking this measure as basis, we define a new semantic relatedness measure among ontology terms. The proposed measure fulfils the above mentioned desirable properties to be used on the Semantic Web. We have tested extensively this semantic measure to show that it correlates well with human judgment, and helps solving some particular tasks, as word sense disambiguation or ontology matching.

Keywords: semantic web, relatedness, relationship discovering.

1 Introduction

Many applications, in Natural Language Processing (NLP) and other fields, benefit from calculating measures to determine numerically how semantically related two words are. Semantic measures can also be defined between lexically expressed word senses, or between whole texts. Three main kind of measures are considered in the literature about this topic: *semantic similarity*, *semantic relatedness* and *semantic distance*. Unfortunately they have not been interpreted always in the same way by different authors. We adopt here the interpretation given in [3].

1. *Semantic similarity*: It is usually defined by considering lexical relations of *synonymy* (e.g. $\langle car, automobile \rangle$) and *hypernymy* (the meaning of a word is encompassed by the another more general term, as in $\langle car, vehicle \rangle$).
2. *Semantic relatedness*: It covers any kind of lexical or functional association, so it is a more general concept than *semantic similarity*. Dissimilar entities may still be related by many possible relationships, such as *meronymy* (or "part of" relation, as in $\langle finger, hand \rangle$), *antonymy* (opposite meanings, as $\langle hot, cold \rangle$), or any kind of functional relationship or frequent association (for example, $\langle penguin, Antarctica \rangle$, that are not linked by any lexical relation).

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 136–150, 2008.

3. *Semantic distance*: It is the inverse of *semantic relatedness*. The more two terms are semantically related, the more semantically close they are.

In this paper we start discussing the interest of using relatedness measures on the Semantic Web, identifying some desirable characteristics that they must accomplish, mainly: domain independence, universality, and maximum coverage. The set of current relatedness measures between words that fulfil these previous requirements is rather limited. Moreover, we have not found measures that explore relatedness between word senses (expressed as *ontological terms*), instead of just between plain words, and accomplish these properties fully.

Next, we choose a well-founded existent semantic distance [5], that uses the Web as knowledge source, to compute a web-based relatedness measure between plain words.

The main contribution of this paper is that, based on the latter, we define a relatedness measure among ontological terms, by exploring their semantic descriptions. This measure is independent of its final purpose, it does not depend on particular lexical resources, it does not need pre-processing tasks, and can operate with any ontology. An evaluation of the web-based word relatedness measure is also presented, by comparison with human judgment, and exploring the effect of using different search engines to access the Web data. Finally, we evaluate the suitability of the proposed relatedness measure among ontological terms when applied to two particular scenarios: ontology matching and word sense disambiguation. Our results show a good behaviour of the measure in all these experiments.

The rest of this paper is organized as follows. Section 2 summarizes the desirable features for a semantic relatedness measure to be used on the Semantic Web. In Section 3 we discuss some related work. Our proposal of semantic relatedness measure appears in Section 4. Experimental results can be found in Section 5, and conclusions and future work appear in Section 6.

2 Applicability and Desirable Features

The *Semantic Web* is conceived as an extension of the current Web, "in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [1]. Semantic measures play an important role in many Semantic Web applications. We start mentioning *word sense disambiguation*, which is a fundamental problem not only in NLP but in the Semantic Web as well. Disambiguation techniques try to pick up the most suitable sense of an ambiguous word according to the context. For example, the word *plant* could mean[1] "building for carrying on industrial labor" or "a living organism lacking the power of locomotion". It is expected that, in a text about car manufacturing, it is used in the first sense, while the second interpretation may be the right one in a web page about vegetal life. Disambiguation methods compare the senses of ambiguous words with words in the context, measuring how related they

[1] According to WordNet 2.0 definitions.

are. Many traditional methods have used similarity measures in this task [3], but relatedness is more convenient, because the context that activates the right meaning of an ambiguous word can be related to it by any kind of relationship (not only by similarity).

Another example of applicability is *ontology matching*, the task of determining relations that hold among terms of two different ontologies [8]. An ontology is a specification of a conceptualization [11], which facilitates interoperability among heterogeneous systems by specifying their data semantics. Most ontologies describe classes, instances, and relations (properties) among them (we will refer to these elements as *ontological terms* in the rest of the paper). Similarity measures give good results in discovering equivalences and hierarchical based relationships among ontology terms. Nevertheless, other relationships may remain hidden by using only similarity measures. We consider that relatedness measures can complement the use of similarity to improve ontology matching tasks. For example, it is expected that two entities related by a non-taxonomical relationship (e.g. $\langle penguin, Antarctica \rangle$) show a low similarity, whereas a high degree of relatedness could reveal that they are linked in some other way.

Relatedness measures can be used in many other applications, such as analysis of structure of texts, annotation, information retrieval, automatic indexing, or spelling correction [3], as well as entity recognition, Semantic Web mining and semantic ranking [13]. According to [20], relationships are in the core of the Semantic Web. Therefore it is of great importance to provide well founded and useful ways to measure relatedness degree; not only among words but, more interestingly, among concepts, expressed as terms in ontologies.

Desirable features for a relatedness measure. We summarize some characteristics that, in our opinion, are desirable for a semantic measure to be used in current Semantic Web applications.

1. *Domain independence.* Nowadays, an increasing amount of online ontologies and semantic data is available on the Web, enabling a new generation of semantic applications [16]. If we want to develop that kind of domain independent applications, we have to deal with this increasing heterogeneity, without establishing in advance the ontologies to be accessed.
2. *Universality.* Semantic measures, in the highly dynamic context of the Web, must be flexible and general enough to be used independently of their final purpose, and without relying on specific lexical resources or knowledge representation languages.
3. *Maximum coverage.* We consider that, in the context of web applications with no predefined domain, *maximum coverage* of possible interpretations of the words must be warranted. If we are limited to a particular knowledge source, such as WordNet[2], or a certain set of ontologies, we are constraining the scope of our applications.

[2] http://wordnet.princeton.edu/

This latter issue has motivated us to focus on the Web as possible source of knowledge. The Web is an information resource with virtually unlimited potential, where millions of people contribute with billions of web pages. As only a minority of users are domain experts, we assume that the Web is not a high-quality corpus. But, due to its immense size, it is likely that extremes cancel and the average web content still hides meaningful implicit semantics. As a matter of fact, the use of the Web as a corpus is gaining a growing popularity [14].

It is also desirable, for a semantic measure, the ability to establish comparisons not only among *plain words*, but also among *senses of words* which, in a Semantic Web context, will be expressed as *ontological terms*.

Finally, absence of pre-processing tasks are desirable, as well as a portable, flexible and scalable implementation, in order to deal with the highly dynamic and heterogeneous environment of the Web.

3 Related Work

Many semantic measures have been proposed in the past to compute degrees of relatedness among words, texts or concepts. Among all possible classifications, we organize the rest of this section according to the source of knowledge utilized.

3.1 Measures Based on Thesauri and Other Lexical Resources

Most of traditional methods to compute semantic measures exploit particular lexical resources: corpus, dictionaries, or well structured taxonomies such as WordNet. Some of them explore path lengths among nodes in taxonomies. Others exploit glosses (textual descriptions of concepts) in dictionaries, while a last group rely on annotated corpora to compute information content [3,22]. These methods result in a limited coverage: for example, as reported in [21], WordNet 2.1 does not include many named entities (e.g. *The Rolling Stones*) or many specialized terms (e.g. *exocytosis*).

Latent Semantic Analysis [6] is a statistical technique that leverages word coocurrence from large unlabeled corpus of texts. Although it can be successfully used for many purposes, its coverage is still restricted to the corpus used as input, and it needs costly pre-processing tasks.

3.2 Measures Based on Wikipedia

Some recent research efforts has focused on using Wikipedia[3] to improve coverage with respect to traditional thesauri-based methods. Nowadays Wikipedia is rapidly growing in size, and it is not difficult to find new terms and named entities on it. In [21], some classic measures are adapted to use Wikipedia instead of WordNet as knowledge source, showing promising results.

A further step in using Wikipedia is found in [9]. They propose a method to represent the meaning of texts or words as weighted vectors of Wikipedia-based

[3] http://en.wikipedia.org

concepts, using machine learning techniques. According to their results, they provide even better correlation with human judgment than [21].

Although they have clear benefits, Wikipedia is still not comparable with the whole Web in the task of discovering and evaluation of implicit relationships. For example, at the time of writing this paper[4], the terms "stomach disease" and "aspirin" do not appear together in any Wikipedia page, but can be found together in 2360 web pages (according to Yahoo!), so their implicit relationship could be inferred by accessing the Web.

3.3 Measures Based on the Web

In order to guarantee the maximum coverage, we have focused on methods that exploit the Web[5] as source of knowledge. In [2] they propose a similarity measure that combines various similarity scores based on page counts, with another one based on lexico-syntactic patterns extracted from text snippets. Other web-based semantic similarity measures can be found in [4] and [19]. They also explore text snippets, but without adding page counts as they do in [2]. We agree with all these works in using web content to compute semantic measures. However they are designed to capture similarity instead of the more general relatedness.

OntoNL semantic relatedness measure for OWL ontologies [13] explores semantic relationships by computing path-based conceptual distances, as well as exploring commonalties of two concepts. A remarkable advantage of this method is that it considers relatedness between ontological concepts (instead of between words, as almost all the other methods do). However, it depends on the particular syntax of OWL (thus not fulfilling our universality requirement), and some of their parameters have to be experimentally determined for a given domain ontology. Moreover, it only compares terms belonging to the same ontology.

The Cilibrasi and Vitányi's Normalized Google Distance [5] (NGD) uses the relative frequency whereupon two terms appear on the Web within the same documents. NGD is well-founded on information distance and Kolmogorov complexity theories, and it does not preclude any kind of relationship between compared words. It fulfils our desirable requirements mentioned in Section 2. However, it does not perform direct comparisons between ontological terms. In Section 4 we show how we use NGD to construct the relatedness measure among ontological terms that we were looking for.

4 Web-Based Semantic Relatedness

In this section we propose a transformation of the Normalized Google Distance [5] into a word relatedness measure, generalizing it to any web search engine. Then we describe a method to compute a web-based relatedness degree among ontology terms, by taking into account their ontological context. Finally, we consider a mixed way of relatedness measure, between ontology terms and plain words.

[4] March 2008.

[5] At least, the Web reachable by current web search engines.

4.1 Semantic Relatedness among Words

As it was mentioned in Section 3, a semantic distance based on Google page counts is defined in [5]. Actually, as the authors indicate, the discussion about the Google Distance is independent of the particular search engine we use to access the Web. Different search engines use different indexes and retrieval methods, thus providing different results in page counts. This motivated us to try other existent web search engines, in order to compare their behaviour (and to choose the most suitable one eventually).

First of all, we generalized the Cilibrasi and Vitányi's Normalized Google Distance $NGD(x, y)$ between search terms x and y, by calling *Normalized Web Distance* $NWD(x, y)$, the same NGD formula they define in [5], but using any web search engine as source of frequencies.

The smaller the value of NWD, the greater the semantic relation between words, e.g. $NWD_{google}(red, blue) = 0.25$, $NWD_{google}(blue, October) = 0.48$. Although most of NWD values fall between 0 and 1, it actually ranges from 0 to ∞. Nevertheless, to obtain a proper relatedness measure, we need a bounded value (in the range [0, 1]) that increases inversely to distance. This can be obtained with the following transformation, which defines our proposed *web-based semantic relatedness* measure between two search terms x and y as (see [10]):

$$relWeb(x, y) = e^{-2NWD(x,y)} \tag{1}$$

We have considered the following web search engines in our experiments: Google, Yahoo!, Live Search, Altavista, Exalead, Ask, and Clusty[6]. Exalead-based measures show the best correlation with human judgment (see Section 5), closely followed by Yahoo! and Altavista.

Up until this point we have merely "translated" the Cilibrasi and Vitányi's Google Distance into a relatedness measure between words. The main contribution of this paper starts from the following, where we use it to compute relatedness among ontology terms, and where we evaluate extensively its use with different search engines and for different purposes.

4.2 Semantic Relatedness among Ontology Terms

In the following we explain our proposed method to compute relatedness among word senses (expressed as terms from any ontology), instead of only among "plain" words. Initially, previously presented Equation 1 can be applied to *any* pair of search terms indexed by a search engine. Nevertheless, we are interested in providing a measure of how much semantically related a pair of senses (expressed as ontological terms) are. The idea is to exploit some elements of the available semantic description of the ontological terms, and to perform "elementary" comparisons among the words that describe them, by using Equation 1.

[6] http://www.google.com, http://www.yahoo.com, http://www.msn.com, http://www.altavista.com, http://www.exalead.com, http://www.ask.com, and http://www.clusty.com, respectively.

In [10] we proposed a first approach to compute a Google-based semantic relatedness measure between ontological terms. We devised a description of an ontological term by means of combining, with logical operators (ANDs, ORs), synonyms and hyperonyms of the ontological term. A search term was then constructed, to be used as input in a search engine, in order to compute semantic relatedness with respect to other terms.

However, this approach was hampered by a technical limitation of most popular search engines: they do not always follow Boolean logic, not giving the expected results in number of hits[7]. Due to this limitation, we have reconsidered the heuristics used to compute semantic relatedness between ontology terms, proposing the general scheme that we detail in the following paragraphs.

Ontological context. Given an ontological term t, we define its *ontological context*, denoted by $OC(t)$, as the minimum set of other ontological terms (belonging to its semantic description) which locate the term in the ontology and characterize its meaning. Such an informal definition is better understood with a simple example: in WordNet ontology, the class "Java" (in the sense of "an Indonesian island"), is well characterized, as well as distinguished from other senses, by considering its direct hypernym "Island". Another example: in an ontology about trips, the property "star" could be well distinguished by specifying the domain it belongs to: "Hotel".

To define the set of elements (other ontological terms) that can be found in the ontological context of the considered term t, we follow this strategy:

1. t is a class $\Rightarrow$ $OC(t)$ is the set of its direct hypernyms
2. t is a property $\Rightarrow$ $OC(t)$ is the set of its domain classes
3. t is an instance $\Rightarrow$ $OC(t)$ is the class it belongs to

We do not establish in advance the source where *ontological context* is obtained from. It could be extracted from two different ontologies, of from a run-time integration of many of them (as we do in [23]), for example. In the following we consider any two ontologies as source of *ontological context*, without considering whether they integrate information from other sources (other ontologies, or even lexical resources) or not.

Relatedness computation. Let us suppose that a and b are two ontological terms (classes, properties or instances) belonging to ontologies O_a and O_b, respectively. We consider that, in the search space of the Web, each sense represented by a and b can be characterized by taking into account two levels of their semantic description: *Level 0*) the term label and its synonyms, and *Level 1*) its *ontological context*, as it was defined in previous paragraphs. Let us call $Syn(a) = \{syn_{a_1}, syn_{a_2}, ...\}$ and $Syn(b) = \{syn_{b_1}, syn_{b_2}, ...\}$ the set of synonyms (equivalent labels, including the term label) of ontological terms a and

[7] For example, they fail the distributive property: the query *"driver AND (car OR train)"* gives 12,700,000 hits in Google (at the time of writing this paper, on March 2008), while the query *"(driver AND car) OR (driver AND train)"* returns 1,140,000.

b, respectively, and $OC(a) = \{oc_{a_1}, oc_{a_2}, ...\}$ and $OC(b) = \{oc_{b_1}, oc_{b_2}, ...\}$ the terms of their ontological context. Notice than $|Syn(x)| \geq 1$ and $|OC(x)| \geq 0$. Then, comparisons are performed as follows:

$$rel_0(a,b) = \frac{\sum_{i,j} relWeb(syn_{a_i}, syn_{b_j})}{|Syn(a)| \cdot |Syn(b)|} \qquad \begin{matrix} i = 1..|Syn(a)| \\ j = 1..|Syn(b)| \end{matrix} \tag{2}$$

Equation 2 provides a measure of how related both terms are at *Level 0*, by averaging the different degree in which different synonyms of the compared terms appear together on the Web. A sum is performed (instead of a maximum, for example) to let all synonyms take part in comparisons.

$$rel_1(a,b) = \frac{\sum_{i,j} rel_0(oc_{a_i}, oc_{b_j})}{|OC(a)| \cdot |OC(b)|} \qquad \begin{matrix} i = 1..|OC(a)| \\ j = 1..|OC(b)| \end{matrix} \tag{3}$$

Equation 3 averages the relatedness at *Level 0* among ontological context terms, to measure how related a and b are at *Level 1*. Notice that it cannot be computed if one of the terms lacks in semantic description (thus not having ontological context: $|OC(x)| = 0$). If that is the case, we still can proceed as we will see in Section 4.3.

Therefore Equations 2 and 3 provide the degree of semantic relatedness between the two corresponding levels that characterize terms a and b [8]. We weight these values to provide a final relatedness between two ontological terms as follows:

$$rel(a,b) = w_0 \cdot rel_0(a,b) + w_1 \cdot rel_1(a,b) \tag{4}$$

where $w_0 \geq 0, w_1 \geq 0$ and $w_0 + w_1 = 1$.

Figure 1 illustrates, with a very simple example, the result of computing $rel(a,b)$ by combining elementary word relatedness calculations. In the example $Syn(a) = \{$"*astronomy*", "*uranology*"$\}$, $Syn(b) = \{$"*star*"$\}$, $Syn(oc_a) = \{$"*physics*"$\}$ and $Syn(oc_b) = \{$"*celestial body*"$\}$. Elementary computations for *Level 0* obtain values for $relWeb($"*astronomy*", "*star*"$)$ and $relWeb($"*uranology*", "*star*"$)$ (0.257 and 0.158 respectively). Their combination leads to $rel_0(a,b) = 0.207$. Similarly, $rel_1(a,b) = 0.247$, and a final $rel(a,b) = 0.227$ is obtained (with $w_0 = w_1 = 0.5$).

Additional remarks. There are other possible ways to define $rel(a,b)$. For example one could consider an arbitrary number of higher levels in the semantic characterization of a term. However, we have restricted its number following this intuition: The higher a word is in the hierarchy that characterize an ontology term, the lesser *information content* it has [17], so it is less significant to characterize the ontology term. For example "Java" (in the sense of "an Indonesian

[8] Notice that Equation 2 is computable in polynomial time on $|Syn(a)|$, $|Syn(b)|$ and Equation 3 is computable in polynomial time on $|OC(a)|$, $|OC(b)|$, $|Syn(oc_{a_i})|$, $|Syn(oc_{b_j})|$. Therefore by controlling the values $|Syn|$ and $|OC|$ we handle the performance of the computation.

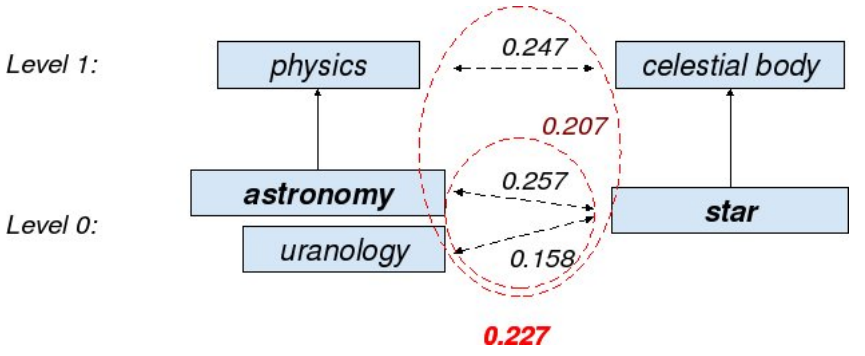


Fig. 1. Example of relatedness computation between ontology terms

island") is closer characterized by the word "island" (direct hypernym) than by "physical object" or "thing" (higher in the hierarchy, thus with less *information content*). Therefore we consider two levels of semantic description. Additional levels could introduce many low significant words in comparisons, also increasing the computation time, as we have found out empirically.

Nevertheless, other variations of the method may be explored in the future, such as weighting differently the synonyms in Equation 2, or considering alternative definitions of $OC(t)$.

4.3 Semantic Relatedness among Ontology Terms and Words

Finally, there are possible scenarios where a *mixed* relatedness measure, between an ontology term t and a plain word w, is required. In that case, we use the following to compute *Levels* 0 and 1:

$$rel_0(t, w) = \frac{\sum_i relWeb(syn_{t_i}, w)}{|Syn(t)|} \qquad i = 1..|Syn(t)| \tag{5}$$

$$rel_1(t, w) = \frac{\sum_i rel_0(oc_{t_i}, w)}{|OC(t)|} \qquad i = 1..|OC(t)| \tag{6}$$

combining their results in this way:

$$rel(t, w) = w_0 \cdot rel_0(t, w) + w_1 \cdot rel_1(t, w) \tag{7}$$

where $w_0 \geq 0, w_1 \geq 0$ and $w_0 + w_1 = 1$.

These previous equations provide a numerical value that indicates the relatedness degree between a sense, described as a term in an ontology, and a word. It can be useful for different purposes, such as the disambiguation experiment we describe in the following section.

Notice that the measures proposed in this section accomplish the good properties we were looking for: they use the Web as source of knowledge, they can be applied to any ontology, and do not rely on particular lexical resources or ontology languages.

5 Experimental Evaluation

In this section, we discuss the experiments that we have carried out to test our proposed relatedness measure. Firstly, we discuss the lack of gold standards to evaluate relatedness measures, presenting our own experiment for a proper evaluation of word relatedness degree, with respect to human judgment. We used it to evaluate how Cilibrasi and Vitányi's based relatedness behaves, in comparison with some other well-established semantic measures. Secondly, the potential of the relatedness measure among ontology terms that we propose in Sections 4.2 and 4.3, is shown in the context of two particular tasks: disambiguation and ontology matching.

5.1 Correlation with Relatedness Human Judgment

Shortage in gold standards to evaluate semantic relatedness measures is a well known problem [3]. The small amount of available data is still inadequate, and it is mainly oriented to evaluate similarity, not relatedness. Obtaining large-enough set of pairs and their correspondent human judgments, with a solid methodology, is still a pending task for relatedness evaluation. Meanwhile, the two more utilized benchmark are: Miller and Charles's data set [15], and WordSim353[9]. The first one is a set of 30 pairs of nouns and their similarity degree according to human opinion. The second is a larger set of 353 word pairs, where subjects were asked for a relationship degree slightly broader than similarity. However, it is still not a relatedness evaluation, and its methodology has been largely discussed [12].

Wikipedia-based methods [21,9] include an evaluation with some of these benchmarks. To let a direct comparison with one of these methods, we computed Equation 1 with Miller and Charles's similarity data sets. Our result, a linear correlation coefficient with human opinion of 0.54, is slightly higher than the 0.46 obtained by WikiRelate [21]. However, this is only partially illustrative, as the experiment only considers *similarity*, instead of the more general *relatedness*.

As these benchmarks are not good enough to evaluate relatedness, instead of only similarity, we were motivated to create a new test data set, focused on relatedness evaluation, as we describe in the following paragraphs.

The experiment we have carried out is similar to Miller and Charles's one. We selected 30 pairs of English nouns, where some kind of relationship are present in most of them: similarity (such as $\langle person, soul \rangle$), meronymy (e.g. $\langle hour, minute \rangle$), frequent association (e.g. $\langle penguin, Antarctica \rangle$), and others. There are, however, other weakly related pairs (e.g. $\langle transfusion, guitar \rangle$). A group of 30 university graduated people (from seven different nationalities, and high skilled in English language) were asked to assess the semantic relatedness between those pairs, in a scale from 0.0 to 4.0 (from no relatedness at all, to identical or strongly related words). In our survey we asked for "how much related the words are", taking into account any possible relationship that could

[9] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/wordsim353.html

connect their meanings, therefore not considering only similarity but the more general concept of relatedness. Table 1 shows the selected pairs of words, as well as the average ratings for semantic relatedness according to human judgment. Word pairs are shown in the table in the same order as they were shown to human evaluators.

Table 1. Group of selected noun pairs and average relatedness according to humans. Score ranges from 0 to 4.

$Word_1$	$Word_2$	$Score$	$Word_1$	$Word_2$	$Score$
car	driver	3.14	person	soul	2.84
transfusion	guitar	0.05	theorem	wife	0.34
professional	actor	2.12	mathematics	theorem	3.30
person	person	4.00	atom	bomb	2.63
city	river	1.85	pencil	paper	2.90
theft	house	1.99	power	healing	1.25
cloud	computer	0.32	hour	minute	3.38
river	lake	3.19	blood	transfusion	3.28
blood	keyboard	0.12	xenon	soul	0.07
dog	friend	2.51	nanometer	feeling	0.11
ten	twelve	3.01	penguin	Antarctica	2.96
citizen	city	3.24	yes	no	3.00
sea	salt	2.87	computer	calculator	2.81
keyboard	computer	3.25	car	wheel	3.02
letter	message	3.16	pen	lamp	0.65

We computed Equation 1 to obtain semantic relatedness values for many different search engines. Results are summarized in Table 2, where comparisons with other WordNet-based measures [22] are provided. WordNet-based measures were computed by using WordNet::Similarity software[10]. We show in Table 2 the measures that obtained the best results[11]. We consider that a linear correlation coefficient it is not appropriate in this evaluation, because some measures produce noticeable non-linear results (such as adapted Lesk measure), thus difficulting comparisons. For this reason we use the Spearman correlation coefficient, which compares corrected ranks of assessments rather than absolute values.

These results show a high correlation between web-based measures and human judgment, thus confirming the validity of using the Cilibrasi and Vitányi's distance as basis to compute relatedness among words. Most of search engines analyzed provide higher correlation than the compared WordNet-based traditional methods, with the remarkable exception of Live Search and Clusty. A detailed discussion of the reasons for the differences between search engines is out of the scope of this paper.

[10] See http://talisker.d.umn.edu/cgi-bin/similarity/similarity.cgi, where additional information about the used measures is available.

[11] Differences with some results we obtained in [10] are due to the use of a later version of WordNet::Similarity tool, based on WordNet3.0 instead of WordNet2.0.

Table 2. Spearman correlation coefficients with human judgment. It ranges from -1 (total disagreement) to +1 (total agreement).

Web-based Measure	Value	WordNet-based Measure	Value
Exalead	**0.78**	Vector	0.62
Yahoo!	0.74	Resnik	0.56
Altavista	0.74	Adapted Lesk	0.56
Ask	0.72	Wu & Palmer	0.47
Google	0.71	Hirst & St-Onge	0.46
Live Search	0.44	Lin	0.46
Clusty	0.41	Leacock & Chodorow	0.41

5.2 Application to Word Sense Disambiguation

In [10] we proposed a multiontology disambiguation method to discover the intended meaning of a set of user keywords. The algorithm relies on a semantic relatedness measure to establish comparisons among word senses, obtaining a relevance degree for each keyword sense, to pick up the most probable one.

In this experiment we used our web-based relatedness among ontology terms and words, computed with Equation 7, within this disambiguation algorithm[12]. We chose Yahoo! as search engine in this experiment due its balance between good correlation with human judgment (see Table 2) and fast time response (for example, four times faster than Exalead). Other choices could be suitable also, such as Google, the only one faster than Yahoo! in our experiments, but however with a correlation with humans slighty worse than Yahoo!.

We illustrate the process by disambiguating the same list of highly ambiguous nouns used in the experiment described in [7] (*glass, earth, plant*). For each word, we have randomly selected ten sentences from SemCor2.0 corpus[13] where it appears with well defined WordNet senses. The target of the experiment is to disambiguate each selected word within each of their ten sentences, thus running 30 different disambiguation processes. We use the other words in the sentence as context to disambiguate, selecting a window of two significant words around the ambiguous one. Then we apply our disambiguation algorithm, to obtain the sense we consider most relevant, and compare it with the SemCor annotation. In Table 3 we show the averaged ratio of successful disambiguations for each keyword. We compare it with two baselines: one corresponding to random selection of senses, and other corresponding to selection of senses with highest frequency of use.

This example illustrate the usefulness of the web-based relatedness measure when applied to disambiguation tasks: even dealing with highly ambiguous words (e.g. *earth* has 6 possible senses in this experiment) we have a high probability of picking up the most suitable one with our method (almost 40% greater than a random selection).

[12] We limited to values $|Syn(x)| = 4$ and $|OC(x)| = 3$, using also $w_0 = w_1 = 0.5$.
[13] http://www.cs.unt.edu/∼rada/downloads.html

Table 3. Disambiguation results

Ambiguous word	Precision	Random baseline	Max. freq. baseline
glass	30%	14%	30%
earth	60%	14%	60%
plant	80%	25%	40%
AVERAGE	**57**%	18%	43%

5.3 Application to Ontology Matching

In this experiment we explore the behaviour of our web-based relatedness measure among ontology terms, in the context of an ontology matching experiment. We used Yahoo! as search engine, for the same reasons explained in Section 5.2.

In [18], it is described an ontology matching experiment between NALT and AGROVOC ontologies[14]. The method they apply derives semantic mappings by dynamically selecting, exploiting, and combining multiple and heterogeneous online ontologies. Each mapping is defined by $\langle a, b, r \rangle$, where a and b are the mapped terms form NALT and AGROVOC ontologies, respectively, and r is the kind of relationship. To assess the quality of the inferred mappings, an extensive manual evaluation was performed, obtaining a precision value of 70%.

We have reused the same experimental data of this ontology matching experiment, but for a slightly different purpose. The goal of our test is to compare human assessment in mappings evaluation, with an assessment based on using our relatedness measure. We expect that, in general, a valid mapping has a greater relatedness degree, between the involved terms, than an invalid one.

We randomly selected a set of 160 human assessed mappings, equally divided between invalid[15] and valid ones. Then, we computed semantic relatedness measures between each pair of the mapped ontology terms, by applying Equation 4.

Using a threshold of 0.19 to assess the validity of mappings according to relatedness values[16], we obtain that 79% of relatedness-based assessments are correct according to human based judgment. This result shows that our measure highly correlates with human assessment of mappings, even when specific domain data are involved (both ontologies belong to agriculture domain).

6 Conclusions and Future Work

In this paper we have studied the state of the art of semantic relatedness measures, and their great interest for many applications. We have also identified some desirables features for a semantic relatedness measure to be used on the

[14] They are ontologies belonging to agriculture domain.
See http://www.few.vu.nl/∼wrvhage/oaei2006/

[15] From the set tagged as "invalid due to incorrect anchoring" in [18].

[16] This optimal threshold has been empirically inferred, and can be reused later in larger experiments.

Semantic Web: domain independence, maximum coverage and universality. In our study we have chosen a well-founded semantic distance (the Cilibrasi and Vitányi's one) to compute the relatedness between two plain words by using the Web as knowledge source. We have shown experimentally that it correlates well with respect to human judgment (even better than some other preexisting measures), exploring also the effect of using different search engines.

Due to the interest of computing relatedness between word senses (expressed as ontology terms) instead on only between plain words, we have taken this word relatedness as basis, to define a new measure with the following characteristics:

1. It computes relatedness among terms from different ontologies (even on different domains), by exploiting their semantic descriptions.
2. It does not depend on specific lexical resources or knowledge representation languages. As it uses the Web as source of data, it maximizes the coverage of possible interpretations.
3. It is general enough to be applied for many different purposes (such as word sense disambiguation, ontology matching, or others).

We have shown that this relatedness measure behaves well when applied to disambiguate a set of polysemous words. It also reproduces well the evaluation of ontology mappings according to human opinion.

As future work, we plan the use of larger data sets in our experiments. Moreover, we devise new ontology matching experiments, in order to exploit the full potential of this measure in relationship discovering tasks.

Acknowledgments. We thank Marta Sabou and Mathieu d'Aquin for providing us with the ontology matching data set we have used in our experiments. This work is supported by the CICYT project TIN2007-68091-C02-02.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American (May 2001)
2. Bollegala, D., Matsuo, Y., Ishizuka, M.: Measuring semantic similarity between words using web search engines. In: Proc. of WWW 2007, Banff, Canada (2007)
3. Budanitsky, A., Hirst, G.: Evaluating wordnet-based measures of semantic distance. Computational Linguistics 32(1), 13–47 (2006)
4. Chen, H.-H., Lin, M.-S., Wei, Y.-C.: Novel association measures using web search with double checking. In: Proceedings of the COLING/ACL 2006, Morristown, NJ, USA. Association for Computational Linguistics (2006)
5. Cilibrasi, R.L., Vitányi, P.M.: The Google similarity distance. IEEE Transactions on Knowledge and Data Engineering 19(3), 370–383 (2007)
6. Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Hashman, R.: Indexing by Latent Semantic Indexing. Journal of the American Society for Inf. Science (1990)
7. Duca, A.: Sketchnet: Knowledge-based word sense disambiguation. In: Proc. of EUROLAN07 Doctoral Consortium (2007)
8. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (2007)

9. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of The Twentieth International Joint Conference for Artificial Intelligence, Hyderabad, India (2007)
10. Gracia, J., Trillo, R., Espinoza, M., Mena, E.: Querying the web: A multiontology disambiguation method. In: Sixth International Conference on Web Engineering (ICWE 2006), Palo Alto (California, USA). ACM, New York (2006)
11. Gruber, T.R.: Towards principles for the design of ontologies used for knowledge sharing. In: Formal Ontology. Kluwer, Dordrecht (1993)
12. Jarmasz, M., Szpakowicz, S.: Roget's thesaurus and semantic similarity. In: Proceedings of the RANLP-2003, pp. 212–219 (2003)
13. Karanastasi, A., Christodoulakis, S.: Ontology-driven semantic ranking for natural language disambiguation in the ontonl framework. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 443–457. Springer, Heidelberg (2007)
14. Kilgarriff, A., Grefenstette, G.: Introduction to the special issue on the web as corpus. Computational Linguistics 29(3), 333–348 (2003)
15. Miller, G.A., Charles, W.G.: Contextual Correlates of Semantic Similarity. In: Language and Cognitive processes (1991)
16. Motta, E., Sabou, M.: Next generation semantic web applications. In: 1st Asian Semantic Web Conference. LNCS. Springer, Heidelberg (2006)
17. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: 14th International Joint Conference on AI, Montreal (Canada) (1995)
18. Sabou, M., Gracia, J., Angeletou, S., d'Aquin, M., Motta, E.: Evaluating the semantic web: A task-based approach. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 423–437. Springer, Heidelberg (2007)
19. Sahami, M., Heilman, T.D.: A web-based kernel function for measuring the similarity of short text snippets. In: Proc. of the 15th Int. WWW Conference (2006)
20. Sheth, A., Arpinar, I., Kashyap, V.: Relationships at the Heart of Semantic Web: Modeling, Discovering and Exploiting Complex Semantic Relationships, vol. 139. Springer, Heidelberg (2003)
21. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using Wikipedia. In: AAAI. AAAI Press, Menlo Park (2006)
22. Ted Pedersen, S.P., Banerjee, S.: Maximizing semantic relatedness to perform word sense disambiguation. Research Report UMSI 2005/25 (2005)
23. Trillo, R., Gracia, J., Espinoza, M., Mena, E.: Discovering the semantics of user keywords. Journal on Universal Computer Science (JUCS). Special Issue: Ontologies and their Applications 13(12), 1908–1935 (2007)

Locally Expandable Allocation of Folksonomy Tags in a Directed Acyclic Graph

Takeharu Eda[1,2], Masatoshi Yoshikawa[2], and Masashi Yamamuro[1]

[1] NTT Cyber Space Laboratories, Japan
{eda.takeharu,yamamuro.masashi}@lab.ntt.co.jp
[2] Kyoto University, Japan
yoshikawa@i.kyoto-u.ac.jp

Abstract. We propose a new classification system based on an analysis of folksonomy data. To find valuable resources from current social bookmark services, users need to specify search terms or tags, or to discover people with similar interests. Our system uses semantic relationships extracted from the co-occurrences of folksonomy data using PLSI and allocates folksonomy tags in a directed acyclic graph. Compared to the hierarchical allocation method of a tree, our method guarantees the number of children nodes and increases the number of available paths to an objective node, enabling users to navigate the resources using tags.

1 Introduction

Web services like blogs and wiki have become very popular. In these services, many entries are updated frequently and many links are generated automatically, degrading the value of web links. There are also a lot of spam blogs, called splogs, which also make link analysis and weighting of web pages difficult.

One method of overcoming these drawbacks, social bookmark services (SBMs) (e.g. del.icio.us[1], Hatena Bookmark[2]) have attracted a lot of attention. They use a bottom-up taxonomy system called *folksonomy*. SBMs offer users bookmarking functions and store bookmark entries on the server. Users can attach tags with varying numbers of keywords to their entries and make comments on entries. Compiling bookmark entries is a good way of ranking fresh user-chosen web pages [15]. Users are loosely connected by bookmarking the same entries.

Folksonomy can be considered one way of classifying resources (URLs). Tags are viewpoints selected by users and used later as clues to find resources. Almost all net surfing users know how to bookmark URLs. In many cases, SBM interfaces are provided as web browsers' plug-ins, which have almost the same functions as default bookmarking functions. Because there is only a small gap between users and SBM systems, users can easily start using SBMs. That is the reason why SBMs are such a popular resource annotating system.

[1] http://del.icio.us
[2] http://b.hatena.ne.jp

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 151–162, 2008.

One folksonomy problem is the explosion of the number of tags. Many SBMs offer simple *tag clouds* where tags are ordered in the order of syllabary or usage frequency. However, it is difficult for users to navigate resources using tag clouds since they have no semantic order.

We propose a new hierarchical allocation method of folksonomy tags by extracting the relationships among them. We compute the feature vectors from co-occurrence data of folksonomy and measure and estimate the abstract differences between tags. We also propose a tag allocation algorithm into DAG (Directed Acyclic Graph). Our method enables users to intuitively understand the connections among tags. Our contributions are as follows.

- We propose a new method of allocating folksonomy tags into DAG (Directed Acyclic Graph). We calculate feature vectors using Probabilistic Latent Semantic Indexing (PLSI), measure them using probabilistic vectors, and estimate abstract upper or lower level differences among tags by comparing the entropy value of the tags.
- We implement both our method and the existing hierarchical tag allocation algorithm. We discuss the differences of between the techniques and compare them in experiments.
- In the experiments, we compare not only the hierarchical structure but also the vector models employed in both approaches, and show that the statistical method (PLSI) can grasp hidden semantics from observed bookmarks.

The paper is organized as follows. In Section 2, we survey related work on folksonomy and clarify our research problems. In Section 3 we propose our new tag allocation method. In Section 4 we conducted qualitative experimentation. Finally, we describe future work and conclude the paper in Section 5.

2 Related Work

The number of social web services supporting user online activities and friendships continues to increase. Folksonomy is attracting a lot of attention as a way of accumulating users' viewpoints on resources and is expanding its role from tagging only URLs into pictures, movies, papers, and etc. These services offer users' scores and comments as well as tags.

A lot of work has been done on folksonomy [11], [13]. Brooks and Montanez [1] proposed automatic tagging based on content and discussed hierarchical allocation of tags. Since it is possible to consider hierarchically allocated tags as an ontology, one of their conclusions is that such a technique is useful in many real applications. However, they also noticed that a tree structure is too strict for users as a map.

There are a lot of case studies on folksonomy data [6], [9]. Golder and Huberman [4] analyzed the distributions of tags, users, and resources in del.icio.us and showed that different users act differently. Sen et al. [12] did a case study of MovieLens and found that tags are organized based on user effort. Niwa et al. [10] consider folksonomy a method of making web page recommendations. They

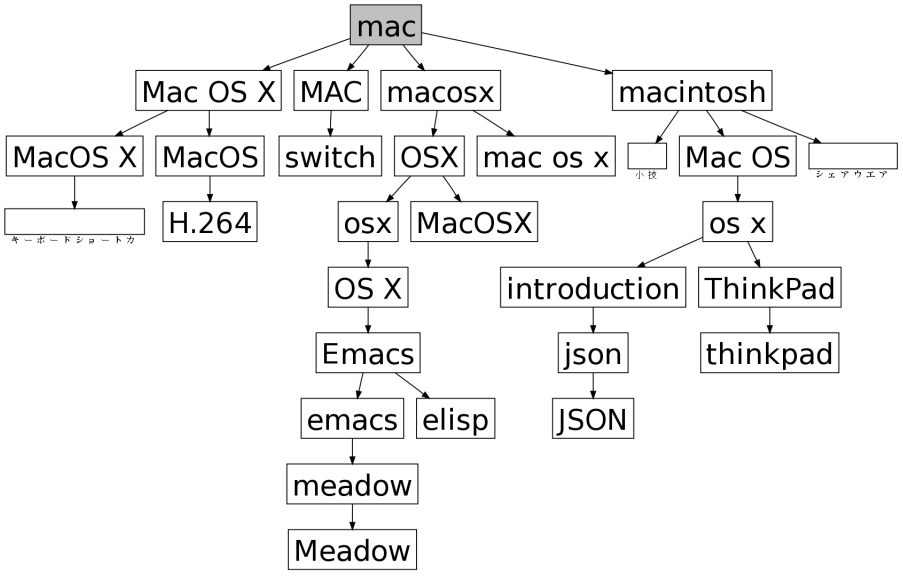


Fig. 1. Heymann's Tree (**EXT** Approach)

insisted that synonym and ambiguity problems can be solved by clustering tags, but they did not evaluate applicability of tags as a classification system. Dubinko et al. [3] visualize frequently used tags in a time series. Since SBMs keep gathering fresh bookmark entries, users can see the history of popular URLs by using such timeline visualizations.

To solve the problem of synonyms and ambiguity in tags, Wu et al. [14] proposed a probabilistic indexing model for folksonomy triples (user, tag, resource). One of the main advantages of their method is the ability to index all attributes at the same time. Another method for tag disambiguation is based on bipartite network analysis [2].

If we consider the problem is one of automatic organization of tags by computers, the obvious choice is to use hierarchical clustering algorithms. However, in such hierarchical clustering, the difficult problem is to determine the labels for inner nodes.

Heymann et al. [5] proposed a method of allocating tags in a tree based on the *folksonomy vector model*. In this model, tag vectors are composed of the number of people who tagged each URL. The dimension of the tag vector is the number of tagged URLs. After extracting tag vectors from folksonomy triples, their method measures the distances among tags with cosine similarities and calculates the centrality of each tag by considering whole tag sets as a network. The centrality they used is betweeness centrality, which can be quickly approximated. They proposed the extensible greedy algorithm where tags are sorted in this order of centrality, and the algorithm starts from the most central tag and builds a tree in top-down manner. However, their method has several problems. One is the

problem of determining the position of a tag to another tag as a child. That is, a node must always have one parent node. Consider real classification systems like Open Directory[3] and Google Directory[4]: we find it natural that several nodes have several parents, making it easier for users to find resources. Another problem is the sparseness of folksonomy vectors, which causes mis-allocation of nodes near the leaf. Fig. 1 show a Heymann's tree which rooted at "mac"[5]. We see that several tags around leaves are far from the topic related to "mac".

3 Proposed Method

We propose a new method of allocating folksonomy tags into DAG. The left side and right side of Fig. 2 illustrate the processing flow of proposed method and Heymann's approach [5], respectively. Both approaches consist of two steps, tag vectorization and hierarchical allocation of tag vectors. In the following, our approach, the PLSI vectors and the DAG allocation algorithm, is referred to as **DAG** and Heymann's approach, the folksonomy vectors and the extensible greedy algorithm, is referred to as **EXT**. At first, we calculate feature vectors using PLSI and we call the vectors PLSI vectors. PLSI exposes hidden relationships among item sets in statistical way; leading to better precision than naive folksonomy vectors.

PLSI vectors of tags are allocated in DAG by determining children of each PLSI vector. We find k nearest neighbors of each vector and set the them as children if a child's abstract level is lower than the parent. We estimate abstract upper or lower level differences among tags by comparing the entropy value of PLSI vectors.

We used three-mode PLSI [14] for 3-tuple co-occurrence data composed of user, tag, and resource. Although PLSI vectors for user, tag, and resource are calculated at the same time, we use only PLSI tag vectors.

3.1 Folksonomy Indexing Using PLSI

PLSI enables us to index co-occurrence data (a subset of the direct product of several item sets) with given dimension feature vectors. This can be understood as projection from each set to a semantic vector space. Another understanding of PLSI is clustering of both documents and terms with *soft* membership functions to clusters.

Although PLSI was first developed for two-mode co-occurrence data (document×term) as a probabilistic variant of LSI [7] , it has been extended to three-modes in several studies. By calculating distances among PLSI vectors, we can index the closeness among items. Since PLSI uses the semantic spaces among item sets, the closeness among PLSI vectors is based on the closeness in the semantic space.

[3] `http://www.dmoz.org`
[4] `http://directory.google.com/`
[5] This was drawn in the setting in Section 4.

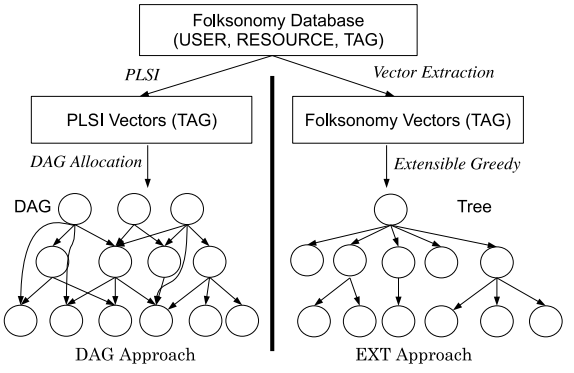


Fig. 2. Overall Processing Flow

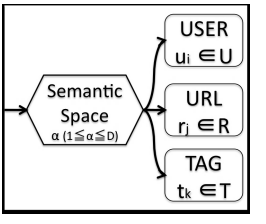


Fig. 3. Probabilistic Model

Fig. 3 shows the model for PLSI used in our method. U, R, and T stand for user set, resource set, and tag set, respectively. The dimension of a semantic space is D. Occurrence probability from $\alpha(\in [1, 2, \ldots, D])$ to $u_i \in U$, $r_j \in R$, $t_k \in T$ is denoted by $p(u_i|\alpha), p(r_j|\alpha), p(t_k|\alpha)$. We assume that occurrence probability of α is $p(\alpha)$. The equation below holds for u_i, r_j, t_k, and the semantic space[6].

$$p(u_i, r_j, t_k) = \sum_{\alpha=1}^{D} p(\alpha)p(u_i|\alpha)p(r_j|\alpha)p(t_k|\alpha)$$

The left-hand side can be observed from the data set, and probabilities in the right-hand side can be estimated using EM algorithms with initial values [14]. EM algorithm consists of E (Expectation) step and M (Maximization) step. Both steps are alternated repeatedly until convergence. In our implementation, tempered EM algorithms are used to avoid local optimums.

Using Bayes' rule, we can calculate conditional probabilities from items to the semantic space. By considering probability from an item to α dimension as a α dimension value in a vector space, we consider all items vectors in D dimensional vector space.

Examples of Calculated PLSI Vectors. Fig. 4 shows an example of PLSI vectors. The horizontal and vertical axes show dimension and probability, respectively. All tags in Fig. 4 related to "mac os x" have a high probability around the 75-th dimension.

Entropy Values of PLSI Vectors. Entropy of tag t_k is calculated as follows.

$$H(t_k) = -\sum_{\alpha=1}^{D} p(\alpha|t_k) \cdot log(p(\alpha|t_k))$$

[6] $p(u_i, r_j, t_k)$ stands for occurrence probability of a triple.

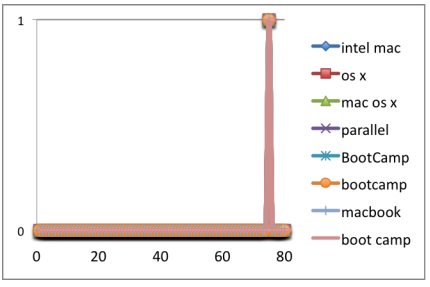


Fig. 4. Example of PLSI Vectors

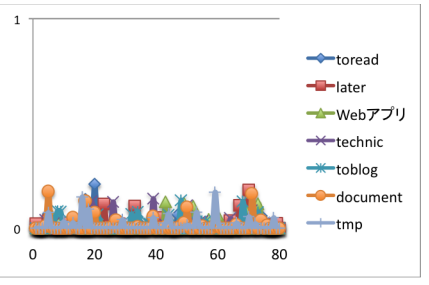


Fig. 5. High Entropy PLSI Vectors

The entropy value equals 0 when the PLSI vector has only one peak and increases as the distribution approaches normal. Entropy is considered as a measure of ambiguity of probabilistic vectors. Fig. 5 shows example PLSI vectors which are of high entropy values, while Fig. 4 shows low entropy vectors.

Distances among PLSI Vectors. In order to measure tag similarity we need distance among PLSI vectors. In this study, we used JS divergence, which is a distance between probabilistic distributions and considered to have better accuracy for information retrieval tasks [8]. JS divergence is calculated as follows[7].

$$D_{js}(t_k, t_l) = \frac{1}{2}[D(t_k||avg(t_k, t_l)) + D(t_l||avg(t_k, t_l))]$$

3.2 DAG Allocation Algorithm

DAG DAG is a directed graph without a directed cycle. Any node leads to an terminal node if the number of nodes in DAG is finite. DAG is intuitively considered to have flow or order of nodes.

Fig. 6 shows a DAG constructed by our method.

We use the entropy value of PLSI vectors to choose this flow and allocate tags in DAG. Table 1 shows the DAG allocation algorithm, which is quite simple. For all nodes, we set children nodes with `set_children`, in which, we first search the nearest neighbors and designate the node as a child if its entropy is less than the parent's entropy. In the data structure constructed by our algorithm, it is impossible to go back to the starting node when navigating directed edges from parent to child. That means that the constructed data structure is a DAG.

One advantage of DAG is that there is no cycle when a user navigates one direction, that is, the user never goes back to the same node while navigating in DAG. The allocation algorithm in Table 1 chooses neighbors whose distances are less than a certain threshold. However, sometimes no children are found. In that case, the condition can be extended to the k nearest neighbor nodes depending on the distribution of JS divergence values. We discuss the distribution of distances in Section 4.

[7] $D(q||r)$ is KL divergence between q and r.

Table 1. DAG Allocation Algorithm

input: PLSI vector list. $\{v_i\}_{i=1}^{i=m}$
output: DAG where PLSI vectors are allocated in.
1 **for**$(i = 1; i < m + 1; i+ = 1)$
2 `set_children`(v_i, k, n)
3 **endfor**

Function: `set_children`
input: Node: v, Threshold: k, Number of Chidren: n
1 $\{w_i\}_{i=1}^{i=n}$: Get all nodes near to v of the distance within k
2 **for**$(i = 1; i < n + 1; i+ = 1)$
3 if$(w_i.entropy < v.entropy)$ then
4 Set w_i as a child of v.
5 **endif**
6 **endfor**

The time complexity of the DAG allocation algorithm is $O(n^2)$, which is in the same class as the extensible greedy algorithm [5]. Though the extensible greedy algorithm is an offline algorithm, which needs to construct the whole hierarchical structure beforehand, our algorithm is an online algorithm that enables us to expand children locally around the required node.

4 Evaluation

We conducted several experiments to test DAG's performance in the following areas.

- Correlation and difference between cosine similarity and JS divergence.
- Precision of children nodes calculated by DAG and the extensible greedy algorithm.

We compared our method with the extensible greedy algorithm using the folksonomy vector model [5]. Both approaches are described in Table 2.

We implemented folksonomy vectors, PLSI vectors, the extensible greedy algorithm, and the DAG allocation algorithm in the Ruby programming language. We first constructed both data structures, a tree and a DAG, and stored them on disk. During the evaluation, we extracted the required sub-structure from disk. We used graphviz[8] to visualize the sub-structures.

As for the data set, we crawled popular entries at Hatena Bookmark in October 2006. We got a list of popular tags from `http://b.hatena.ne.jp/t` and retrieved a set of URLs that were linked from the site. HTML files were scraped into folksonomy triples and stored in a RDBMS. Before PLSI, we filtered out the users, tags, and resources that appeared less than five times, which caused us to filter triples out too, for the purpose of noise reduction.The data set is shown in Table 3. All

[8] `http://graphviz.org`

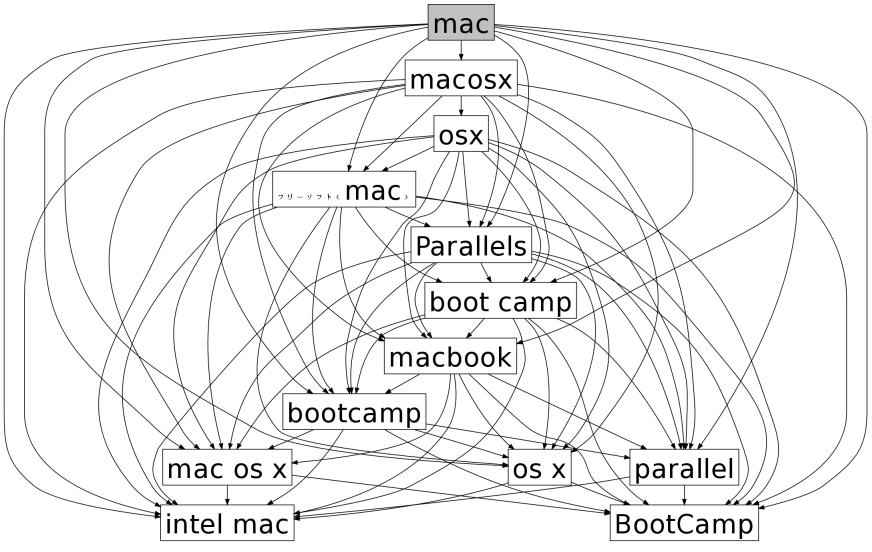


Fig. 6. DAG (**DAG** Approach)

Table 2. DAG and **EXT** Approaches

	DAG	**EXT**
Vector	PLSI Vector	Folksonomy Vector
Distance	JS Divergence	Cosine Similarity
Data Structure	DAG	Tree

the parameters of the algorithms were chosen empirically as shown in Table 4. EM iterations were chosen with the same setting used in [14]. We tried several dimensions for the semantic space and chose the dimension by checking kNN of sampled tags by users. **DAG**'s thresholds k and n were determined in an ad-hoc manner since **DAG** has the ability to expand children locally. In the case of **EXT**, we tried several thresholds and determined the threshold to avoid the very shallow tree[9].

4.1 Differences in Distances

Fig. 7 shows the distribution of distances between 300 randomly chosen tags; the horizontal axis represents cosine similarity and the vertical axis represents JS divergence.

Although, there is no clear association between cosine and JS divergence, we notice that there were several points whose cosines equaled 0 but whose JS divergence varied over a wide range. To verify this point, we calculated the cases

[9] If we increase the threshold of **EXT**, many nodes become children of the root.

Table 3. Data Set

Tags	5496
Users	11713
Resources	4019
Triples	1190504

Table 4. Parameters

DAG	Dimension of Semantic Space	80
	EM Iterations	80
	Threshold: k	0.2
	Num. of Children: n	11
EXT	Threshold: k	0.00001

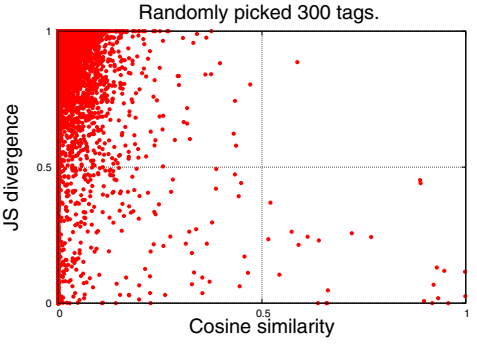


Fig. 7. Distribution of Cosine and JS Divergence

where two tags become maximally distant from each other, that is, the cosine similarity equals 0 and the JS divergence equals 1, respectively. The result is that almost half(45%) of the folksonomy vector pairs in the data set were maximally distant from each other. In contrast, only a few(7%) of PLSI vectors are maximally far from each other. This means that the statistical method(PLSI) enables us to measure the tags which are furthermost in folksonomy vector model.

Revealed Connections

Table 5 shows tag pairs whose cosine similarity equals 0 and whose JS divergence is near 0. Tag pairs are randomly chosen, and only English words are shown[10]. Using **EXT** approach we were unable to detect the relationships shown in Table 5, and detected for our **DAG** approach. These trends are applicable to tags in Japanese.

4.2 Comparison of Local Structures

We randomly sampled tags and compared children nodes rooted at the tag. For all sampled tags, we counted the number of child nodes in both methods and evaluated the precision of child nodes by hand. Since both **DAG** and **EXT** approaches do not aim for the extraction of exact parent-child relationships like *is-a* and *part-of*, the correct answers were judged from the points of relatedness and ancestor-descendant relationship.

[10] Since Hatena bookmark is Japanese bookmark service, there are many tags in Japanese.

Table 5. Revealed Connections. (Example pairs of tags cosine equals 0 and JS divergence is near to 0.)

CRC	partition
CSS3	xhtml
KDDI	Softbank
attention economy	nagios
mod_proxy_balancer	rrdtool
j2ee	BDD
ASP.NET	Trac
Rails	memcached
Winny	Privacy
vim	memcached
GoogleEarth	google mars

recover	nLite
Wii	RMT
VB.NET	Trac
KDDI	WILLCOM
3DCG	flash
prototype	wysywig
WinFS	foldershare
trends	NAMAAN
liveup	BRAVIA
Plagger	LINUX
trac	unison

The average number of child nodes and the average precision are shown in Fig. 8. **DAG**'s average precision is 11% less than that of **EXT**. However, **DAG**'s average number of child nodes is eight. This means that the **DAG** approach has 2.7 times as many child nodes for just an 11% decrease in precision compared to **EXT**. Furthermore, the **DAG** approach's ability of *local expansion* is confirmed from Fig 9.

4.3 Summary

The PLSI vector model enables us to reveal the hidden relationships among tags with JS divergence distance. Several software names which run on Apple's "mac" do not appear in Fig. 1 but appear in Fig. 6. This property is the main advantage of applying the statistical method for indexing co-occurrence data like folksonomy triples.

The **DAG** approach has 2.7 times as many child nodes as the **EXT** approach with 11% less precision when parameters are statically determined. However, the **EXT** approach cannot expand children locally while **DAG** approach enables us to expand children with only a small drop in precision. **DAG** can be used in

(average)	Num. of Children	Precision
DAG	8.00	0.64
EXT	3.00	0.75

Fig. 8. Average Number of Children and Precision

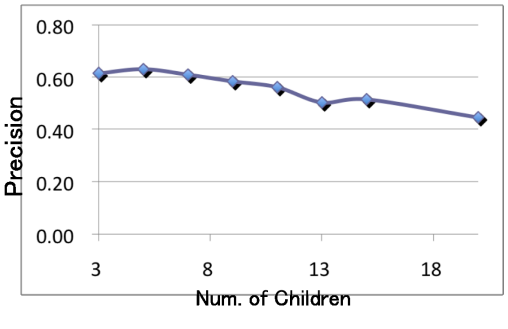


Fig. 9. Precision-Num. of Children Curve

applications such as user navigation tool bars to give users more navigation choices instantly on their demand.

5 Conclusions

We proposed a new method of allocating tags in DAG and evaluated our approach qualitatively comparing to the existing hierarchical tag allocation algorithm. We analyzed folksonomy triples with PLSI and allocate tags in DAG.

Experiments showed that PLSI vectors are scattered in a small dimensional vector space compared to the naive folksonomy vector space, which leads to the revelation of hidden relationships among tags. We found that **DAG** offers 2.7 times as many child nodes with 11% less precision in the statical parameter setting. We also showed that our approach has the ability to expand children locally with small precision degradation. These capabilities can be applied to interactive folksonomy navigation systems.

To exploit the locally expandable tag hierarchy, threshold adjustment is an interesting future work. Increasing the thresholds means that tag islands are grouped into larger islands. By snapshotting DAGs with several thresholds, we might see the hierarchical clusters above DAGs. Since folksonomy systems are continuing to compile new entries, visualization of DAG in a time series is another future work.

References

1. Brooks, C.H., Montanez, N.: Improved annotation of the blogosphere via autotagging and hierarchical clustering. In: Proc. International World Wide Web Conference, pp. 625–632. ACM Press, New York (2006)
2. Yeung, C.A., Gibbins, N., Shadbolt, N.: Tag Meaning Disambiguation through Analysis of Tripartite Structure of Folksonomies. In: Proc. International Conferences on Web Intelligence and Intelligent Agent Technology (2007)
3. Dubinko, M., Kumar, R., Magnani, J., Novak, J., Raghavan, P., Tomkins, A.: Visualizing tags over time. In: Proc. International World Wide Web Conference. ACM Press, New York (2006)
4. Golder, S., Huberman, B.A.: The structure of collaborative tagging systems. Journal of Information Science (2006)
5. Heymann, P., Garcia-Molina, H.: Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems. Technical report, `http://heymann.stanford.edu/taghierarchy.html`
6. Heymann, P., Koutrika, G., Garcia-Molina, H.: Can Social Bookmarking Improve Web Search? In: Proc. International Conference on Web Search and Data Mining (2008)
7. Hofmann, T.: Probabilistic latent semantic analysis. In: Proc of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Stockholm (1999)
8. Lee, L.: On the effectiveness of the skew divergence for statistical language analysis. In: Proc. International Workshop on Artificial Intelligence and Statistics, pp. 65–77 (2001)

9. Lux, M., Granitzer, M., Kern, R.: Aspects of Broad Folksonomies. In: Proc. International Conference on Database and Expert Systems Applications (2007)
10. Niwa, S., Doi, T., Honiden, S.: Web Page Recommender System based on Folksonomy Mining. In: Proc. International Conference on Information Technology: New Generations (2006)
11. Sabou, M., Gracia, J., Angeletou, S., dAquin1, M., Motta, E.: Evaluating the Semantic Web: A Task-Based Approach. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 423–437. Springer, Heidelberg (2007)
12. Sen, S., Lam, S.K., Rashid, A.M., Cosley, D., Frankowski, D., Osterhouse, J., Harper, M.F., Riedl, J.: Tagging, communities, vocabulary, evolution. In: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work. CSCW, New York, NY, USA, pp. 181–190 (2006)
13. Vo, J.: Tagging, Folksonomy & Co - Renaissance of Manual Indexing? unpublished (2007), http://arxiv.org/abs/cs/0701072v2
14. Wu, X., Zhang, L., Yu, Y.: Exploring Social Annotations for the Semantic Web. In: Proc. International World Wide Web Conference, pp. 417–426 (2006)
15. Yanbe, Y., Jatowt, A., Nakamura, S., Tanaka, K.: Can Social Bookmarking Enhance Search in the Web? In: Proceedings of the 7th ACM/IEEE Joint Conference on Digital Libraries (2007)

Computing Relaxed Answers on RDF Databases

Hai Huang[1], Chengfei Liu[1], and Xiaofang Zhou[2]

[1] Faulty of ICT, Swinburne University of Technology, Australia
{hhuang,cliu}@groupwise.swin.edu.au
[2] School of ITEE, The University of Queensland, Australia
zxf@uq.edu.au

Abstract. Database users may be frustrated by no answers returned when they pose a query on the database. In this paper, we study the problem of relaxing queries on RDF databases in order to acquire approximate answers. We address two problems for efficient query relaxation. First, to ensure the quality of answers, we compute the similarities of relaxed queries with regard to the original query and use them to score the potential relevant answers. We also propose the algorithm to get most relevant answers as soon as possible. Second, to optimise query relaxation process, we characterize a type of unnecessary relaxed queries which do not contribute to the final results and propose the method to prune them from the query relaxation graph. At last, we implement and experimentally evaluate our approach.

Keywords: RDF Database, RDF Query, Query Relaxation.

1 Introduction

Several RDF repositories have been developed in recent years such as Jena [1], Sesame [2], rdfDB [3]. Some of them can scale to million triples. With RDF databases growing in size and complexity, it becomes impractical to expect the users know enough about the contents and the structure of a database. So even when a user has a clear idea about the query condition, the database may still return an empty answer. In this case, the user has to change the query condition and try it again, which might become a "trial-and-error" process. Obviously, database users may be frustrated by this tedious process. The database system copes with it by relaxing the query condition automatically when the query fails to produce answers or enough answers the user expects. With query relaxation, the user can get exactly matched and non-exactly matched relevant answers.

A way of retrieving approximate answers is making the query condition more general in order to match potential answers. In RDF databases, structures do not come in the shape of well-defined schemas but in terms of semantic annotations that confirm to a schema called RDF schema (or RDFS ontology), which describes taxonomies of classes and properties. To enable relaxation, a query can be generalized to capture enough relevant answers using information provided by RDF schema.

For example, consider that a user wants to retrieve 10 proceedings editors who acquired doctoral degree from for University of Queensland (UQ) and work for

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 163–175, 2008.

Swinburne University (Swin) from the given RDF database. The user issues the following SPARQL query [4] on a database:

```
PREFIX   rdf:< http://www.w3.org/1999/02/22-rdf-syntax-ns# >
PREFIX   abc: < http://www.w3.org/abc.owl# >
SELECT ?X
WHERE {
   ?X   abc:doctoralDegreeFrom   abc:UQ.
   ?X   abc:worksFor     abc:Swin.
   ?X   abc:ProceedingsEditorOf   ?Y.
   ?Y   rdf:type   abc:Proceedings.
}
```

Fig. 1. A SPARQL Query

The system should return the exactly matched answers of the query. If no exactly matched answers return or the matched answers are not enough (less than 10), the system might relax the query condition using information in the RDFS ontology (defined in Fig.1) such as rewriting the condition (?X, doctoralDegreeFrom, UQ) to (?X, DegreeFrom, UQ) or (?Y, type, Proceedings) to (?Y, type, Book) for retrieving some potential relevant answers. Since the property "DegreeFrom" is more general than "doctoralDegreeFrom" and "Book" is the super class of "Proceedings" defined in the RDFS ontology.

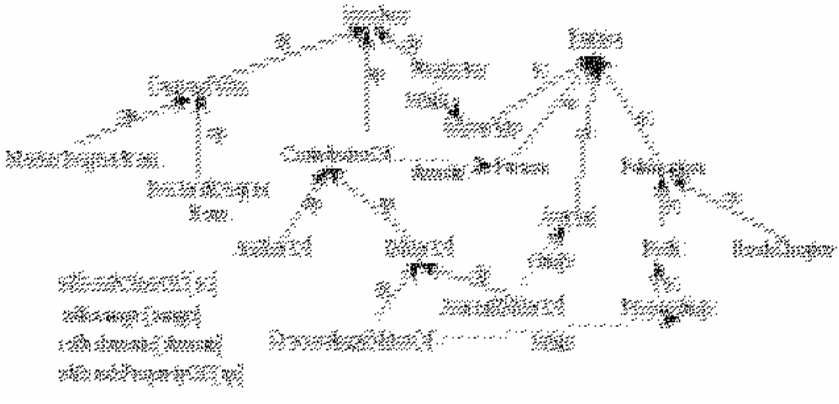


Fig. 2. A Sample of RDFS Ontology

From this example, two observations are in order. First, to guarantee the quality of answers, it is desirable to score the possible relaxed queries. Second, since several ways for relaxing the query condition may be available and the number of relaxations is huge, it is necessary to design an efficient relaxation algorithm that controls the query relaxation process and get most relevant answers as soon as possible. In this paper, we address these problems and make the following contributions:

(1) We measure the similarity degrees of the relaxed query with regard to the user query and use them to score the relevant answers.

(2) We design the algorithm to execute the relaxed queries according to their similarity score and return their answers incrementally.
(3) We characterize a type of *unnecessary relaxed queries* which do not contribute to the final results and propose the method to prune them from the query relaxation graph.

The remainder of this paper is organized as follows. We give the overview in section 2. We define the similarities of relaxed queries in section 3. Section 4 describes the relaxation algorithm and characterizes a type of unnecessary relaxed queries. Section 5 discusses related work. Section 6 presents an experimental evaluation of our approach.

2 Preliminary

2.1 Data and Query Model

A triple $t(s,p,o) \in (I \cup B) \times I \times (I \cup B \cup L)$ is called an RDF triple, where I is a set of IRIs (Internationalized URIs), B a set of blank nodes and L a set of literals. In a triple, s is called subject, p the property (or predicate), and o the object. RDF triples can be classified as schema triples (which describe the schema information) and data triples (which describe the instance data). In this paper, we consider data model where information is represented as a collection of RDF data triples stored in a relational database.

Each triple can also be represented as a node-arc-node link. The directed arc is the predicate and the nodes are its subject and object. A set of such triples is called an RDF graph. An RDF query is referred to as an RDF graph pattern, which consists of triple patterns. An **RDF triple pattern** $t(s,p,o) \in (I \cup V) \times (I \cup V) \times (I \cup V \cup L)$, where V is a set of variables disjoint from the sets I, B and L. **RDF graph pattern** $G = (q_1, q_2, ..., q_n)$, $q_i \in T$, where T is a set of triple patterns. Now we define the query model as follows.

Given a user query Q = (Atom (Q), K), where K is the number of answers that users desire and Atom (Q) = {<q_i, w_i>} defines the triple pattern with its weight w_i ($0<w_i<1$, default value of w_i is 1/m, where m is the number of triple patterns). For example:

$Q = \{< q_1, 0.2>, < q_2, 0.3>, < q_3, 0.3>, < q_4, 0.2>\}$
$q_1 = (?X, abc:phdDegreeFrom, abc:UQ)$
$q_2 = (?X, abc:worksFor, abc: Swin)$
$q_3 = (?X, abc:ProceedingsEditorOf, ?Y)$
$q_4 = (?Y, rdf:type Proceedings)$
$K=10$

2.2 Query Relaxation Model

Our techniques are based on logical relaxation of query conditions. In [5], the authors propose two kinds of relaxation for triple pattern which exploit RDF entailment to relax the queries.

Simple relaxation on triple pattern: Given RDF graphs G_1, G_2, a map from G_1 to G_2 is a function u from the terms (IRIs ,blank nodes and literals) in G_1 to the terms in G_2, preserving IRIs and literals, such that for each triple $(s_1, p_1, o_1) \in G_1$, we have $(u(s_1),$

$u(p_1)$, $u(o_1)) \in G_2$. This simple relaxation exploits the RDF simple entailment. AN RDF graph G_1 simply entails G_2, denoted by $G_1 \underset{simple}{\Rightarrow} G_2$, if and only if there is a map u from G_2 to G_1: $G_2 \xrightarrow{u} G_1 : \frac{G_1}{G_2}$. We call triple pattern t_2 the *simple relaxation* of t_1, denoted by $t_1 \underset{simple}{\prec} t_2$, if $t_1 \underset{simple}{\Rightarrow} t_2$ via a map u that preserves variables in t_1. For example, there is a map u from the terms of triple pattern (?X, type, ?Y) to (?X, type, Proceedings) that makes u("?X")= "?X", u("type")= "type" and u("?Y")= "Proceedings". So (?X, type, Proceedings) can be relaxed to (?X, type, ?Y) by replacing "Proceedings" with "?Y" and (?X, type, Proceedings) $\underset{simple}{\prec}$ (?X, type, ?Y).

Ontology relaxation on triple pattern: This type of relaxation exploits RDFS entailment in the context of an ontology (denoted by *onto*)**.** We call $G_1 \underset{RDFs}{\Rightarrow} G_2$, if G_2 can be derived from G_1 by iteratively applying rules in groups (A), (B) (sc, sp are rdfs:subclassOf and rdfs:subpropertyOf for short):

Group A (Subproperty) (1) $\frac{(a, sp, b)(b, sp, c)}{(a, sp, c)}$ (2) $\frac{(a, sp, b)(x, a, y)}{(x, b, y)}$

Group B (Subclass) (3) $\frac{(a, sc, b)(b, sc, c)}{(a, sc, c)}$ (4) $\frac{(a, sc, b)(x, type, a)}{(x, type, b)}$

Let t_1, t_2 be triple patterns, where $t_1 \notin$ closure (*onto*), $t_2 \notin$ closure (*onto*). We call t_2 an *ontology relaxation* of t_1, denoted by $t_1 \underset{onto}{\prec} t_2$, if $t_1 \cup onto \underset{RDFs}{\Rightarrow} t_2$. It includes relaxing type conditions and properties such as: (1) replacing a triple pattern (a, type, b) with (a, type, c), where (b, sc, c)$\in$ closure(*onto*). For example, given the ontology in Fig.2, the triple pattern (?X, type, Proceedings) can be relaxed to (?X, type, Book). (2) replacing a triple pattern (a, p_1, b) with (a, p, b), where (p_1, sp, p)$\in$ closure(*onto*). For example, the triple pattern (?X, ProceedingsEditorOf, ?Y) can be relaxed to (?X, EditorOf, ?Y).

Definition 1 ***(Relaxed Triple Pattern)*:** Given a triple pattern t, t' is a relaxed pattern obtained from t, denoted by $t \prec t'$, through applying a sequence of zero or more of the two types of relaxations: simple relaxation and ontology relaxation.

For example, the triple pattern (?X,ProceedingsEditorOf, ?Y) $\prec$ (?X, Contributer Of , ?Y)

Definition 2 ***(Relaxed Query Pattern)*:** Given a query pattern Q (t_1, t_2,...,t_n), Q' (t_1', t_2'... t_n') is the relaxation of the Q, denoted by $Q \prec Q'$, if there exists at least one pair (t_i, t_i') that we have $t_i \prec t_i'$.

Definition 3 ***(Relevant Answers)*:** A relevant answer to the query Q is defined as a match of a relaxed query of Q.

A user query can be relaxed through replacing a term in the query by a more general term or replacing values to variables. For instance, a query Q (shown in Fig.1) has relaxed queries such as Q'={(?X, DegreeFrom, UQ),(?X, worksFor, Swin),

(?X, ProceedingsEditorOf, ?Y),(?Y, type, Proceedings)}. Q″={(?X, dotoralDegree-From, UQ),(?X, worksFor, Swin),(?X, ProceedingsEditorOf, ?Y),(?Y, type, Book)}. Obviously, there are many different ways to relax a query and the amount of relaxations will be large. Returning all answers of these relaxed queries is not desirable. So we rank the relaxed queries based on their similarities to the original query and execute the most similar relaxed queries first in order to achieve most relevant answers. Given the user query Q and the relaxed queries Q′, Q″ of query Q, If Q′ is more similar to Q than Q″, then we return the results Q′ prior to the results of Q″. Thus, the query relaxations require measuring the similarities of the relaxed queries with regard to the original query.

3 Similarities of Relaxed Queries

Since the amount of possible query relaxations may be large, blindly relaxing query would not be helpful for getting the most relevant answers. To guarantee the quality of answers, it is desirable to measure the similarities of the relaxed query to the user query and choose the most similar one to execute on the database first. An RDF query is referred to as a graph pattern, which consists of triple patterns. Thus we compute the similarities of the relaxed triple patterns and then combine them to compute the similarities of the relaxed queries.

3.1 Similarities of Triple Patterns

Note that an RDF triple pattern can be represented as a node-arc-node link with two nodes (subject, object) and one arc (property). Hence we introduce the similarity between nodes as well as arcs and integrate them to define our similarity of the relaxed triple patterns. We define a finite set of class names *IC* and property names *IP*.

Similarity between nodes: In the triple pattern q_1, if one node (subject or object) is a class $c_1 \in IC$, defined in the RDFS ontology, and it is relaxed to its super class c_2 through ontology relaxation we exploit the notion of the Least Common Ancestor (LCA) to compute the similarity of two nodes based on the ontology. The LCA of two nodes c_1 and c_2 in the IS-A hierarchy is the node that is an ancestor of both c_1 and c_2, where depth of a concept node is the maximum length of from the concept to the root of the taxonomy.

$$sim(c_1, c_2) = \frac{2 \times depth(LCA(c_1, c_2))}{depth(c_1) + depth(c_2)}$$

For example, Sim("Book","Proceedings")=2×2/(2+3)=0.8. If the node (subject or object) in the triple pattern is relaxed to a variable through simple relaxation, we define the similarity of the two nodes is *0*. For example, (?X, type, Proceedings) is relaxed by (?X, type, ?W), we have *sim*("Proceedings","?W")=0.

Similarity between arcs: If the property $P_1 \in IP$ in the triple pattern is relaxed to its super property P_2 through ontology relaxation, since the hierarchy (sub property relationship) on properties is also defined in the RDFS ontology, the similarity definition on nodes can be transferable to arcs. We have:

$$sim(p_1, p_2) = \frac{2 \times depth(LCA(p_1, p_2))}{depth(p_1) + depth(p_2)}$$

$LCA(p_1,p_2)$ indicates the least common super property of p_1 and p_2. And the depth of a property node is the maximum length from the property to the root of the sub property taxonomy.

For example, *sim*("proceedingsEditorOf","EditorOf")=2×2/(2+3)=0.8 If the predicate in the triple pattern is relaxed to a variable through simple relaxation, we define the similarity of the two arcs as 0. For example, (?X, doctoralDegreeFrom, UQ) is relaxed to (?X, ?W, UQ), then we have sim("doctoralDegreeFrom" , "?W")=0.

Similarity of triple patterns: Given a triple q(s, p, o) and its relaxation q′(s′, p′, o′), we integrate similarity between nodes and arcs to define the similarity score of the relaxed triple patterns as follows:

$$score(q, q') = sim(s, s') + sim(p, p') + sim(o, o') \quad (1)$$

3.2 Similarities of Relaxed Queries and Relevant Answers

Given a query pattern Q (q_1, q_2,…,q_n) and its relaxation Q′ (q_1', q_2',…,q_n'), we can compute the semantic similarity *comScore* between them as follows:

$$comScore(Q, Q') = \sum_{i=1}^{n} w_i \times score(q_i, q_i') \quad (2)$$

Where w_i is the weight of triple pattern q_i in the query pattern Q. The scoring function *comScore* is *monotone* in the sense that if Q′ (q_1', q_2',…,q_n') and Q″ (q_1'', q_2'',…,q_n'') are the relaxations of query Q and $score(q_i, q_i') \geq score(q_i, q_i'')$, then $comScore(Q, Q') \geq comScore(Q, Q'')$.

In general, we can define the score of a relevant answer as the similarity of relaxed the query it matches. However, a relevant answer may match the different relaxed queries, e.g., ("John", AuthorOf, "WISE conference") would match queries (?X, AuthorOf, "WISE conference") and (?X, ContributerOf, "WISE conference"). To deal with this situation, we define the score of a relevant answer as follows:

Definition 4 (Score of a relevant answer): The score of a relevant answer is the maximum *comScore* among all of the relaxed queries it matches.

4 Query Relaxation Processing

In this section, we focus on properly relaxing a user query that produces a sequence of relaxed queries based on their similarity scores. To achieve it, the query relaxation algorithm is presented. We also characterize a type of "unnecessary relaxed queries" which do not contribute to the final results and propose the method to prune them. At last, the execution algorithm on a database is proposed.

4.1 Relaxation Algorithm

We design the relaxation algorithm that ensures the desired quality of answers. The relaxed queries are executed in a sequence based on their similarities with regard to the original query. For instance, Fig.1 shows the query Q={(?X, doctoralDegreeFrom, UQ), (?X, worksFor, Swin), (?X, ProceedingsEditorOf, ?Y), (?Y, type, Proceedings)} and it has several relaxed queries, such as: Q1={(?X, DegreeFrom, UQ), (?X, worksFor, Swin), (?X, ProceedingsEditorOf, ?Y), (?Y, type, Proceedings)}. Q_4={(?X, dotoralDegreeFrom, UQ), (?X, worksFor, Swin), (?X, ProceedingsEditorOf, ?Y), (?Y, type, Book)}. Since *comScore*(Q,Q_1)=0.964>*comScore*(Q,Q_4)=0.944 (shown in Fig.3), which means that query Q_1 is more similar to the user query and hence should be executed first. We now present a query relaxation strategy, which selects the next most similar relaxed query to execute on the database till enough results are returned. We first construct the relaxation graph and then give the relaxation algorithm based on this graph.

Ordering the relaxations of triple patterns: Given the user query denoted by $Q(q_1^{(0)},q_2^{(0)},\ldots,q_n^{(0)})$, we compute the relaxations of each triple pattern and rank them by their similarity scores. $q_i^{(0)}$ indicates the original triple pattern and $q_i^{(m_i)}$ indicates the m_ith relaxation of triple pattern $q_i^{(0)}$. An example is shown in Table 1. If the original triple pattern $q_i^{(0)}$ is relaxed to $q_i^{(m_i)}$, we say that the triple pattern $q_i^{(0)}$ goes forward m_i steps. Obviously, if $m_i \le m_j$, then $score(q_i^{(0)},q_i^{(m_i)}) \ge score(q_i^{(0)},q_i^{(m_j)})$.

Relaxation Graph: The relaxation graph is a directed graph representing the relaxed queries. Each node in the relaxation graph is a relaxed query. Q_2 is Q_1's *succeeding node* if the corresponding triple patterns in two queries are same except only one triple pattern in Q_1 that is relaxed one step to the corresponding triple pattern in Q_2. Q_1 is also called Q_2's *preceding node*. For instance, $Q_3(q_1^{(0)},q_2^{(0)},q_3^{(1)},q_4^{(0)})$ is $Q(q_1^{(0)}$

Table 1. Ranking of the relaxed triple patterns of query Q with scores

$q_1^{(0)}$:(?X,doctoralDegreeFrom, UQ) ; 1 $q_1^{(1)}$:(?X, degreeFrom, UQ); 0.82 $q_1^{(2)}$:(?X,doctoralDegreeFrom?W_1); 0.54 …	$q_2^{(0)}$:(?X,worksFor,Swin); 1 $q_2^{(1)}$:(?X,worksFor,?W_3); 0.55 $q_2^{(2)}$:(?X,?W_4,Swin); 0.5 …	$q_3^{(0)}$:(?X,ProceedingsEditorOf, ?Y); 1 $q_3^{(1)}$:(?X,EditorOf,?Y); 0.7 $q_3^{(2)}$:(?X,CotributerOf,?Y); 0.6 …	$q_4^{(0)}$(?Y,type,Proceedings); 1 $q_4^{(1)}$(?Y,type,Book);0.72 $q_4^{(2)}$(?Y,type,Publication); 0.65 …

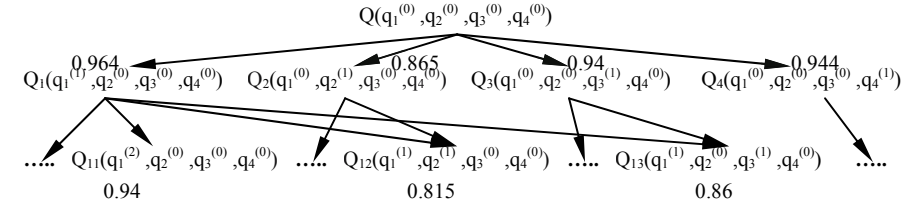


Fig. 3. Relaxation graph of query Q

$q_2^{(0)}$,$q_3^{(0)}$,$q_4^{(0)}$)'s succeeding node. An *edge* from node Q_1 to Q_2 exists if and only if Q_2 is Q_1's succeeding node. The relaxed queries can be categorized into different levels. For a relaxed query $Q_j(q_1^{(m_1)}, q_2^{(m_2)}, \ldots, q_n^{(m_n)})$, if $m_1+m_2+\ldots+m_n=h$, then Q_j is in the level *h*. An example of query relaxation graph is shown in Fig.3. Note that the relaxation graph we consider has several interesting properties, which will serve as the bases for query relaxation algorithm.

Property 1. Given a relaxed query $Q_j(q_1^{(m_1)}, q_2^{(m_2)}, \ldots, q_n^{(m_n)})$ and its succeeding nodes $Q_{j1}(q_1^{(m_1+1)}, q_2^{(m_2)}, \ldots, q_n^{(m_n)})$, $Q_{j2}(q_1^{(m_1)}, q_2^{(m_2+1)}, \ldots, q_n^{(m_n)}), \ldots, Q_{jn}(q_1^{(m_1)}, q_2^{(m_2)}, \ldots, q_n^{(m_n+1)})$, we have $comScore(Q,Q_j) \geq comScore(Q,Q_{ji})$, $i \in [1, n]$, and at least one relaxed query in level *h* has higher *comScore* than all relaxed queries in level $h' > h$.

Proof. Let query $Q_j(q_1^{(m_1)}, \ldots, q_i^{(m_i)}, \ldots, q_n^{(m_n)})$ and $Q_{ji}(q_1^{(m_1)}, \ldots, q_i^{(m_i+1)}, \ldots, q_n^{(m_n)})$ be the relaxations of query $Q(q_1^{(0)}, q_2^{(0)}, \ldots, q_n^{(0)})$ where Q_{ji} is a succeeding relaxation node of Q_j, it holds that $score(q_i^{(0)}, q_i^{(m_i)}) \geq score(q_i^{(0)}, q_i^{(m_i+1)})$. By monotonicity of scoring function *comScore*, we have $comScore(Q,Q_j) \geq comScore(Q,Q_{ji})$. It shows that the *comScore* of one relaxed query is higher than its succeeding nodes. So the relaxation node with maximum *comScore* in level *h* must has higher *comScore* than all relaxed queries in level $h' > h$. □

Property 2. Given the original query $Q(q_1^{(0)}, q_2^{(0)}, \ldots, q_n^{(0)})$, the relaxed query Q′ with the *h*th highest *comScore* must be in the level *h* or less.

Proof. When x=1, the relaxed query with the highest comScore must fall in level 1 since property 1. We assume that when x=h, it holds that the relaxed query with the *h*th highest comScore falls in level h. When x =h+1, we can find at least one relaxed query in level h+1 has higher comScore than all relaxations in level h′>h+1 according to property 1. So when x =h+1, we can get h+1 relaxed queries within h+1 level. □

We exploit Property 1 and Property 2 to define the following basic relaxation algorithm that generates the next most similar relaxed queries incrementally. From property 2, the relaxed query with the highest *comScore* must be in the first level. Thus we first get the succeeding nodes of the original query and choose the query with the highest *comScore* to execute on the database. If the number of answers is not enough, then we choose the next relaxed query with the second highest *comScore* from the first or second level.

Basic Relaxation Algorithm /* Input: $Q(q_1^{(0)}, q_2^{(0)}, \ldots, q_n^{(0)})$, *K* the number of results required ; Output : *Result* */

```
1   Initial Result= Φ, Candidates= Φ, Processed= Φ;
2   Insert Q's succeeding nodes into Candidates;
3   While | Result |<K and |Candidates| >0 do
4       Select the Qi with the highest comScore from Candiates
5       Insert Qi's succeeding nodes into Candidates;
6       R←Execute (Qi);
7       Result ←Result ∪ R;
8       Add Qi to Processed;
9       Remove Qi from Candiates ;
10  End while
11  Return Result;
```

For example, the user query Q{(?X, doctoralDegreeFrom, UQ), (?X, worksFor, Swin), (?X, ProceedingsEditorOf, ?Y), (?Y, type, Proceedings)}(in Fig.1) is executed and the query Q's succeeding relaxed queries Q_1-Q_4 are added into *Candidates*. If the number of answers is still not enough, we choose the relaxed query Q_1{(?X, DegreeFrom, UQ), (?X, worksFor, Swin), (?X, ProceedingsEditorOf, ?Y), (?Y, type, Proceedings)} that has the highest *comScore* in *Candidates* and execute it on the database. We also add Q_1's succeeding relaxed queries into *Candidates*, and remove Q_1 from *Candidates*. This process is repeated till enough answers are acquired or no more candidate is left.

4.2 Pruning Unnecessary Relaxations

In the basic relaxation algorithm above, when a relaxed query is selected, we add its succeeding nodes (relaxations) into *Candidates* for comparison of next round. However, the succeeding relaxations may be useless to generate new answers compared to the preceding relaxed query and there are unnecessary relaxations. For instance, in our running example, the query Q has one of its relaxation Q_4{(?X, DegreeFrom, UQ), (?X, worksFor, Swin), (?X, ProceedingsEditorOf, ?Y), (?Y, type, Book)}, which replaces the term "Proceedings" in query Q to "Book" only. However, this relaxation Q_4 will not generate new answers compared to the answers of its preceding node (query Q). Because the triple (?Y, type, Proceedings) has join dependency with the triple pattern (?X, ProceedingsEditorOf, ?Y). In the relaxed query Q_4 "Proceedings" is relaxed to "Book", but the domain of property "ProceedingsEditorOf "is still "Proceedings" (according to the RDFS ontology defined in Fig.2) and this relaxation will generate no new answers compared to the answers previously returned by the query Q. The basic algorithm relaxes the classes or properties in the triple patterns of the query *locally*. It fails to consider the join dependency between the triple patterns through common variables. Consequently, some relaxed queries may not return new answers. There are two cases of unnecessary relaxations that fail to contribute new answers, one in generalizing the subject or object term of a triple pattern, while the other in generalizing the property.

We first define the boolean operator "$\leq$"on classes and properties. If class c_1 is the subclass of c_2, we have $c_1 \leq c_2$. Similarly, if property p_1 is the sub property of p_2, we have $p_1 \leq p_2$.We also define the operator "*subclass*(*C*)" which computes the set of sub classes of *C* and "subproperty(P)", which computes the set of sub properties of *P*.

Case 1: Given a query Q $\{q_1,\ldots,q_i(?x, type, c_1),\ldots,q_j(?x, p_2, o_2)$ or $(s_2, p_2, ?x),\ldots,q_n\}$, its succeeding relaxation Q′ $\{q_1,\ldots,\underline{q_i'(?x, type, c)},\ldots,q_j(?x, p_2, o_2)$ or $(s_2, p_2, ?x),\ldots,q_n\}$ is obtained from query Q through replacing class c_1 to c, where $c_1 \leq c$, p_2 is a property and $o_2 \in (I \cup V \cup L)$, $s_2 \in (I \cup V)$.

$$\Delta\omega = subclass(c) - subclass(c_1)$$

If $\exists\xi \in \Delta\omega$ and $\xi \leq domain(p_2)$(or $range(p_2)$), then the relaxed query may have new answers generated. Otherwise, the relaxed query has no new answer generated compared to query Q.

When generalizing the property of a triple pattern in the query, new answers could be obtained through matching the generalized property or its sub properties. Because of join dependency between triple patterns, the domain (subject) and range (object) of the generalized property or its sub properties may not match with other properties in the query. This is formalized as follows:

Case 2: Given a query $Q\{q_1,\ldots,q_i(?x, p_1, o_1) ,\ldots,q_j(?x, p_c, o_2),\ldots,q_n\}$, its succeeding relaxation $Q'\{q_1,\ldots,\underline{q_i'(?x, p, o_1)},\ldots,q_j(?x, p_c, o_2),\ldots,q_n\}$ is obtained from the query Q through replacing the property p_1 with p, where $p_1 \leq p$; p_c is a property and $o_1, o_2 \in (I \cup V \cup L)$. After relaxation, new answers could be added through matching the property p or its sub properties. Notice that $q_i'(?x, p, o_1)$ has join dependency with $q_j(?x, p_c, o_2)$ through variable "?x". If $D=subproperty(p)\text{-}subproperty(p_1)$ and $(\bigcup_{p_i \in D} subclass(domain(p_i))) \cap (subclass(domian(p_c))) = \Phi$, then the relaxed query Q′ has no new answer generated. It is straightforward to generalize the method to the situation that q_i and q_j take the form $(s_1, p_1, ?x)$ and $(s_2, p_c, ?x)$.

Now we give the optimised relaxation algorithm, which checks the usefulness of new relaxed query and skip those unnecessary relaxed queries.

Optimised Relaxation Algorithm /* Input: $Q(q_1^{(0)}, q_2^{(0)}, \ldots, q_n^{(0)})$, *K* the number of results required ; Output : *Result */*

```
1   Initial Result= Φ, Candidates= Φ, Processed= Φ;
2   Insert Q's succeeding nodes into Candidates;
3   While | Result |<N and Candidates| >0 do
4       Select the Qi with the highest comScore from the Candiates;
5       Insert Qi's succeeding nodes into Candidates;
6       For each of Qi's preceding nodes Qj in Processed
7           If   Qi is an unnecessary relaxation of Qj then
8               Goto step 13
9           End if
10      End for
11      R←Execute (Qi);
12      Result ←Result  ∪ R;
13      Add Qi to Processed;
14      Remove Qi from Candiates ;
15  End while
16  Return Result
```

5 Related Work

Hurtado et al. [5] propose an rdf query relaxation method through RDF(s) entailment producing more general queries for retrieving potential relevant answers. Our work focuses on using heuristic information based on the similarity degree of relaxed queries with regard to the original query to control the relaxation process for ensuring the desired cardinality and quality of answers, which is not studied in [5].

Answering top-k selection queries is related to our work. Fagin et al. [6, 7] introduce a set of novel algorithms, assuming sorted access and/or random access of the objects is

available on each attribute. Carey and Kossmann [8] optimise top-k queries when the scoring is done through a traditional SQL order by clause, by limiting the cardinality of intermediate results. Other works [9, 10, and 11] use statistical information to map top-k queries into selection predicates which may require restarting query evaluation when the number of answers is less than K. Natsev et al. [12] introduce the J* algorithm to join multiple ranked inputs to produce a global rank. These works define the semantics of the relevant answers based on numeric conditions in selection and join predicates, which can be quantified as value differences. In contrast, we relax query conditions and rank answers based on semantic information provided by RDF ontology.

Cooperative query answering [13, 14, 15] are designed to automatically relax user queries when the selection criteria is too restrictive to retrieve enough answers. Such relaxation is usually based on user preferences and values. In contrast, in our work, we compute the semantic similarities between the relaxed queries and the original query. Efforts have been made to the problem of query relaxation on XML data [16]. Amer-Yahia et al. [16] compute approximate answers for weighted patterns by encoding the relaxations in join evaluation plans. The techniques of approximate XML query matching are mainly based on structure relaxation and can not be used to handle query relaxation on RDF data directly.

6 Experiments

Based on the Lehigh University Benchmark LBUM [17], we generate the dataset which contains 6,000k triples. The data is stored in and managed by Mysql 5.0.11. All algorithms are implemented using Jena SDB [18] (which provides for large scale storage and query of RDF datasets using conventional SQL databases) and run on a windows XP professional system with P4 3G CPU and with 512 M RAM. We developed 5 queries which are shown in Table 2.

Benefits of the optimised algorithm: In this experiment, we compare the basic relaxation algorithm and optimised algorithm. We fix the number of answers K=100 and performed the 5 queries on the data which contains 6,000k triples. Fig.4 (a) and (b) show the relaxation steps and execution time for each query. We use query Q_4 as an example to illustrate how the optimisation algorithm reduces the relaxation steps.

Table 2. Queries for experiments

Q_1:(?x,type,TeachingAssitant)(?x,teachingAssistantOf,http://www.Department0.University0/Course3)(?x,mastersDegreeFrom,http://www.Department0.University0.edu)
Q_2:(?x,teacherOf,?z)(?x,ub:worksFor,University0)(?x,type,AssistantProfessor)
Q_3:(?x,advisor,?y)(?y,type,AssistantProfessor)(?y,researchInterest',Research12')(?y,worksFor,http://www.University0.edu)
Q_4:(?x,advisor,?y)(?y,type,Professor)(?y,worksFor, http://www.University0.edu)
Q_5:(?x,type,JournalAticle)(?y,publicationAuthor,?x) (?y,type,Professor)

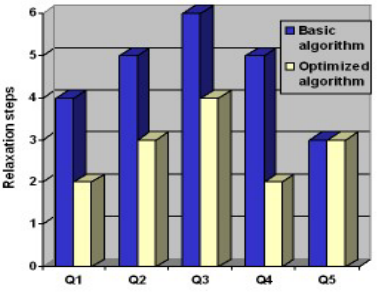


Fig. 4. (a). Relaxation Steps

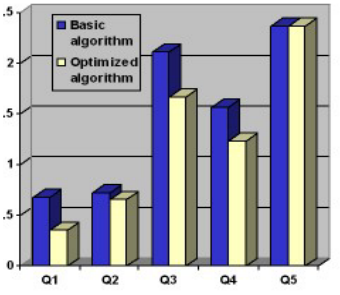


Fig. 4. (b). Query Response Time

Query Q_4 is relaxed to its relaxation Q_4' through replacing *"Professor"* to *"Person"* only. In Q_4' the range of property *"advisor"* is still *"Professor"*. So only replacing *"Professor"* to *"Person"* will not generate new answers compared to the answers of query Q_4 and Q_4' is a unnecessary relaxation to query Q_4. Fig.4 (a) and (b) show that the optimisation algorithm can reduce the relaxation steps and execution time.

Efficiency of query relaxation: Fig.4 (b) shows that the running time of 5 queries for K=100 relaxed answers. Q_5 is the most expensive query. Because the query condition of Q_5 is more general and with the increase of relaxation steps the relaxation process costs much. Q_1 is the most efficient one since its query condition is more concrete. It takes less running time than Q_5, though the relaxation steps of Q_1 are less than that of Q_5. Moreover, our relaxation algorithm can return relevant answers incrementally and users can stop the relaxation process at any time when they are satisfied with the answers generated.

7 Conclusion

This paper addressed the issue of relaxing the user query on RDF databases and computing most relevant answers. We measured the similarity degrees of the relaxed queries with regard to the user query and designed the algorithm to retrieve the most relevant answers as soon as possible. We characterized a type of unnecessary relaxed queries which do not contribute to the final results and proposed the method to prune them from the query relaxation graph. The experiments validated our approach. A further optimisation is subject to our future work such as multiple query optimisation on a sequence of relaxed queries by reusing the intermediate results of selection or join operations.

References

1. Jena, `http://jena.sourceforge.net/`
2. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: International Semantic Web Conference, pp. 54–68 (2002)

3. Guha, R.: rdfDB: An RDF Database, http://www.guha.com/rdfdb/
4. SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/
5. Hurtado, C.A., Poulovassilis, A., Wood, P.T.: A Relaxed Approach to RDF Querying. In: International Semantic Web Conference, pp. 314–328 (2006)
6. Fagin, R., Lotem, A., Naor, M.: Optimal Aggregation Algorithms for Middleware. In: PODS (2001)
7. Fagin, R.: Fuzzy Queries in Multimedia Database Systems. In: PODS, pp. 1–10 (1998)
8. Carey, M.J., Kossmann, D.: On Saying "Enough Already!" in SQL. In: SIGMOD Conference, pp. 219–230 (1997)
9. Bruno, N., Chaudhuri, S., Gravano, L.: Top-k selection queries over relational databases: Mapping strategies and performance evaluation. ACM Trans. Database Syst. 27(2), 153–187 (2002)
10. Chen, C.-M., Ling, Y.: A Sampling-Based Estimator for Top-k Query. In: ICDE, pp. 617–627 (2002)
11. Hristidis, V., Koudas, N., Papakonstantinou, Y.: PREFER: A System for the Efficient Execution of Multi-parametric Ranked Queries. In: SIGMOD Conference, pp. 259–270 (2001)
12. Natsev, A., Chang, Y.-C., Smith, J.R., Li, C.-S., Vitter, J.S.: Supporting Incremental Join Queries on Ranked Inputs. In: Proc. of the 27th International Conference on Very Large Data Bases, pp. 281–290 (2001)
13. Godfrey, P.: Minimization in Cooperative Response to Failing Database Queries. Int. J. Cooperative Inf. Syst. 6(2), 95–149 (1997)
14. Chu, W.W., Yang, H., Chiang, K., Minock, M., Chow, G., Larson, C.: CoBase.: A Scalable and Extensible Cooperative Information System. J. Intell. Inf. Syst. 6(2/3), 223–259 (1996)
15. Kleinberg, J.M.: Authoritative Sources in a Hyperlinked Environment. J. ACM 46(5), 604–632 (1999)
16. Amer-Yahia, S., Cho, S., Srivastava, D.: Tree Pattern Relaxation. In: Jensen, C.S., Jeffery, K.G., Pokorný, J., Šaltenis, S., Bertino, E., Böhm, K., Jarke, M. (eds.) EDBT 2002. LNCS, vol. 2287, pp. 496–513. Springer, Heidelberg (2002)
17. Guo, Y., Pan, Z., Heflin, J.: An Evaluation of Knowledge Base Systems for Large OWL Datasets. In: International Semantic Web Conference, pp. 274–288 (2004)
18. Jena SDB, http://jena.hpl.hp.com/wiki/SDB

Semantically Enhanced Entity Ranking

Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, and Wolfgang Nejdl

L3S Research Center
Leibniz Universität Hannover
Appelstrasse 9a, 30167 Hannover, Germany
{demartini,firan,iofciu,nejdl}@l3s.de

Abstract. Users often want to find entities instead of just documents, i.e., finding documents entirely about specific real-world entities rather than general documents where the entities are merely mentioned. Searching for entities on Web scale repositories is still an open challenge as the effectiveness of ranking is usually not satisfactory. Semantics can be used in this context to improve the results leveraging on entity-driven ontologies. In this paper we propose three categories of algorithms for query adaptation, using (1) semantic information, (2) NLP[1] techniques, and (3) link structure, to rank entities in Wikipedia. Our approaches focus on constructing queries using not only keywords but also additional syntactic information, while semantically relaxing the query relying on a highly accurate ontology. The results show that our approaches perform effectively, and that the combination of simple NLP, Link Analysis and semantic techniques improves the retrieval performance of entity search.

1 Introduction

Entity search is becoming an important step over the classical document search as it is done today on the Web. The goal is to find real-world entities (e.g., persons, places, events) relevant to a query more than just finding documents (or parts of documents) which contain the searched keywords. Ranking entities according to their relevance to a given query is crucial in scenarios where the amount of information is too large to be managed by the end user. A correct ranking can help systems in presenting the user only with entities of interest, and avoiding the user having to analyze the entire set of results. As a concrete example, suppose the user wants to find which creatures are herbivorous mammals. The informational need, in this case, is answered by a set of instances of the class *mammals* of the type *herbivorous* instead of pages just mentioning those mammals. Moreover, a single document might contain several relevant entities. Instead of the user having to aggregate different information sources, the system should be able to present entity overview pages as results.

In this paper we present our approach to ranking entities in Wikipedia and we investigate how semantic information influences results. In Wikipedia many, but not all (see pages starting with "List of", e.g., *List of European countries*),

[1] Natural Language Processing.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 176–188, 2008.

are assumed to represent a real world entity. The text content of a page can be seen as the entity description. Therefore, it is possible to evaluate entity ranking algorithms, that make use of entity descriptions, on the Wikipedia corpus. Evaluation is done on the Wikipedia XML corpus provided within the INEX 2007 Initiative[2]. The evaluation results show that making use of Natural Language Processing techniques and leveraging link and semantic information improve the quality of the results enabling a better search experience.

The rest of the paper is organized as follows: In section 2 we present and compare the previous approaches in entity search and ranking. In section 3 we present the overall system architecture describing an ontology based on Wikipedia and WordNet, that we use for improving the effectiveness of the entity ranking algorithms (section 3.1), and the dataset used in the evaluation of the developed system (section 3.2). In section 4 we formalize the ranking algorithms we propose and in section 5 we present the evaluation results and the comparison among the proposed algorithms. Finally, in section 6 we conclude the paper and describe future improvements.

2 Related Work

There are only few systems that deal with entity search and ranking. ESTER[4] is a system which combines full-text and ontology search and supports prefix search and joins. The authors have applied their search engine to the English Wikipedia and as supporting ontology they have used YAGO[12][3]. ESTER focuses on efficiency rather than precision and recall. Another framework, presented in [5], focuses on finding different types of entities (e.g., phone numbers, emails, etc.) on the Web. While their main accent is on scaling on Web-size datasets for a limited number of entity types, we can better manage heterogeneous sets of entity types. Many approaches rely on Wikipedia as dataset for producing important research results, allowing the extension and the semantic exploitation of this knowledge base [8]. A previous approach already used category and link information present in Wikipedia for extracting semantic information [6]. They organize Wikipedia categories as a graph, and, based on the number of links, they try to discover relations between categories. Our approach also uses existing category linkage information present in Wikipedia together with syntactic information and a highly accurate ontology. Other approaches focused on extracting entities from query logs [11,13]. In extension to their approaches, we focus on how to rank entities in order to provide the relevant ones to the end user. In the context of ranking entities the NAGA system has been proposed [10,9]. It is a highly effective Web-based semantic search engine which uses advanced scoring methods for ranking results from a knowledge base of entities and relationships. The drawback of this system is that it uses a graph-based query language which might be not self-explanatory for the common Web surfer. In our settings we use a keyword query typed

[2] `http://inex.is.informatik.uni-duisburg.de/2007/`

[3] Presented later in section 3.1.

by the user together with an explicit informational need describable in free text.

3 System Architecture

3.1 The YAGO Ontology

YAGO[12][4] is a large and extensible ontology that builds on entities and relations from Wikipedia. Facts in YAGO have been automatically extracted from Wikipedia and unified with semantics from WordNet[5], achieving an accuracy of around 95%. All objects (e.g., cities, people, even URLs) are represented as entities in the YAGO model. The hierarchy of classes starts with the Wikipedia categories containing a page and relies on WordNet's well-defined taxonomy of homonyms to establish further *subClassOf* relations. We make use of these *subClassOf* relations in YAGO, which provide us with semantic concepts describing Wikipedia entities. E.g, knowing from Wikipedia that *Married... with Children* is in the category *Sitcoms*, we reason using YAGO's WordNet knowledge that it is of the type *Situation Comedy*, same as *BBC Television Sitcoms*, *Latino Sitcoms*, *Sitcoms in Canada*, and 8 more.

3.2 The INEX Wikipedia Collection

The document collection used for evaluating our approaches is the Wikipedia XML Corpus based on an XML-ified version of the English Wikipedia in early 2006 [3]. The considered Wikipedia collection contains 659,338 Wikipedia articles. On average an article contains 161 XML nodes, where the average depth of a node in the XML tree of the document is 6.72. The original Wiki syntax has been converted into XML, using general tags of the layout structure (like *article, section, paragraph, title, list,* and *item*), typographical tags (like *bold, emphatic*), and frequently occurring link-tags. For details see Denoyer and Gallinari [3].

The official topics have been manually assessed by the INEX 2007 participants. The set contains adapted ad hoc topics along with entity ranking designed topics. We report in table 1 an example of an INEX 2007 Entity Ranking Topic:

Although this task seems easy given the Wikipedia corpus, we have to retrieve results matching the sought type. Relevant results for the example given in table 1 would thus be: "Texas Star", "London Eye", or "Big-O", all of these being ferris wheels. Irrelevant results, although they still contain some information related to the Topic, would be: "London" (where "London Eye" is mentioned as a tourist attraction), or even "Ferris wheel" (the page describing what a ferris wheel is).

3.3 System Integration

In this section we describe the architecture of the Entity Ranking System we used. The architecture design is presented in figure 1. The first step is the creation

[4] Available for download at `http://www.mpii.mpg.de/~suchanek/yago`
[5] `http://wordnet.princeton.edu/`

Table 1. INEX Entity Ranking Topic example

Topic	#67
Title	Ferris and observation wheels
Description	Find all the Ferris and Observation wheels in the world.
Narrative	I have been to the "Roue de Paris" last Sunday and enjoyed it. I would like to know which other wheels exist or existed in the world, to find out the highest and what buildings you can see from each.
Category	ferris wheels

of the inverted index from the XML Wikipedia document collection. Starting from the raw structured XML documents, we created a Lucene[6] index with one Lucene document for each Wikipedia page. We created fields for each article element (e.g., *title*, *text*, *categories*, *links*, ...) which are searchable in parallel with an integrated ranking.

After the creation of the index, the system can process the INEX Entity Ranking 2007 Topics. Different approaches are adopted for building queries out of INEX Topics (as shown later in detail in section 4). Four modules are used interchangeably or complementary as main resources for our algorithms (see figure 1): *Wikipedia Taxonomy* is used for getting Wiki Category Links, *Entity Linkage Information* is needed for exploring outgoing Links of Wikipedia entities, the *Lexical Analyzer* performs simple Natural Language Processing tasks, while the *YAGO* ontology is used as underlying knowledge base. The INEX Topic is processed using these modules in order to create a disjunctive Lucene query starting from Title, Description, and Narrative information, along with the specified Category from the Topic. After the generation of the query, the index can be queried and a ranked list of retrieved entities is generated as output.

4 Algorithms

Our approaches are based on semantic techniques, Natural Language Processing, Link Analysis, and combinations of those. Queries for entity retrieval are built out of the Topics, enhanced with the different techniques, and searched throughout the different index fields.

4.1 Naïve Approach

This is the baseline approach, where the query is constructed from both textual and contextual information given in the Topic. For the textual part of the query we consider the analyzed text from Title, Description, and Narrative of the Topic which we search in the *title* and *text* fields of the indexed Wikipedia pages. In the contextual part of the query we consider Category information from the

[6] `http://lucene.apache.org`

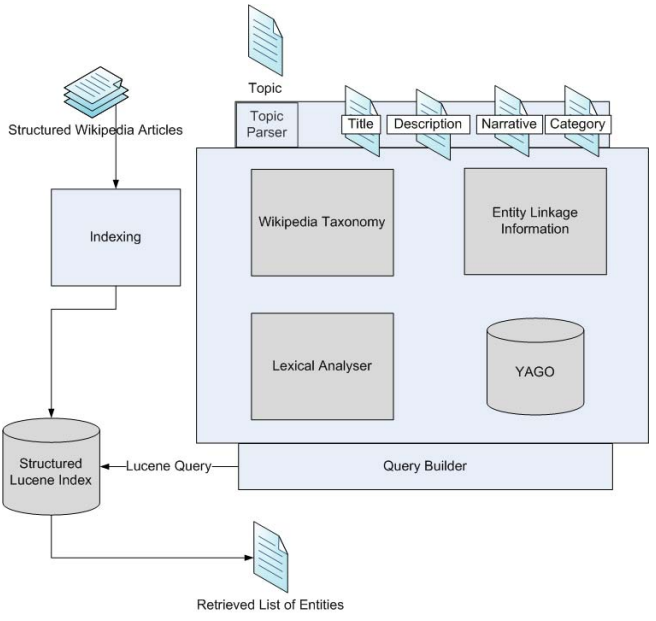


Fig. 1. Architecture of the Entity Ranking System

Topic which we search in the *category* field of the index. We do not make this part of the query mandatory as category information available in Wikipedia is not always accurate or sometimes not even present. Nevertheless, the existing entity-in-category information influences the results for the respective entities. The retrieval is done using the default Lucene implementation, i.e., the vector space model with cosine similarity and TFxIDF weighting. For example, for the Topic in table 1 using the *naïve approach* (with stopword removal), a Lucene query for searching in the *text* and in the *category* of Wikipedia pages, is of the form:

text:(ferris observation wheels) text:(find ferris observation wheels world) text: ("roue de paris" last sunday enjoyed would like know other wheels exist existed world highest buildings) category:(ferris wheels)

4.2 Categories Based Search: Using a Supporting Ontology

While the Category contained in the Topic should contain most or all of the retrievable entities, this is for many Topics not the case. Wikipedia is constructed manually by different contributors, so that category assignments are not always consistent or accurate. Many categories are very similar and in some of these cases the difference is very subtle. Thus, very similar entities are sometimes placed in different categories by different contributors (e.g., hybrid powered automobiles appear inconsistently either in the *Hybrid Vehicles* or the *Hybrid Cars* category, and very seldom are they in both).

In the previous approach the given Category in the Topic was used to make the query more likely to retrieve entities from within that category. The methods described here construct an additional list of categories closely linked to the given one in the Topic. This extended list of categories is then used instead of the one Category in query construction. We use three types of category expansion: Subcategories, Children, and Siblings.

Subcategories. Wikipedia itself has a hierarchical structure of categories. For each category we are presented with a list of subcategories. This list of subcategories is taken as-is and added to the query. E.g., some of the subcategories for the *Actors* category are: *Animal actors, Child actors, Actors with dwarfism, Fictional actors.*

Children. The Subcategories list is the initial list of candidates for Children. We can not include all the Wikipedia subcategories in our Children list as some of them are not semantically a subcategory, or not of the same type. As subcategories for a country we might have categories about presidents, movie stars, or other important persons for that country. This means that although we have as a starting category a *country* we end up having *people* as subcategories. The obvious solution to this is selecting only those subcategories having the same class as the initial category. As described in Section 3.1, YAGO contains also class information about categories. We will make use of this *subClassOf* information to identify suitable categories of the same kind. Thus, a Wikipedia subcategory is included in the Children list only if the intersection between its ancestor classes and the ancestor classes in YAGO (excluding top classes like *entity*) of the initial category is not empty. The final list of Children will therefore contain only subcategories of the same type as the given Category in the Topic, which specifies the type of entities the user is interested in.

Siblings. Using YAGO we can also retrieve categories of the same type as a given category, not restricting just to the Wikipedia subcategories. We first determine the type of the starting category using the *subClassOf* relation in YAGO. Knowing this type we construct a list of all categories of the same type and add them to the Siblings set. Siblings are thus all categories of the exact same type as the initial category.

Constructing the query is done similar to the *naïve approach* setting. The difference relies in the category matching part. In the *naïve approach* we had only one Category (given with the Topic) while here we have the additional lists of Subcategories, Children, and/or Siblings.

4.3 Using Semantics from the Topic: Extraction of Category Information

In the INEX Topics, we had extensive data about the information need. We used the Topic textual information (Title, Description, and Narrative) for building a query. In addition, the main Category to be searched was given with the Topic. But in a usual search environment people specify only a short unstructured

textual description as their information need (i.e., without category or information). To model such users and the lack of additional information, we created a method for automatically extracting a possible category list given only the Title of the Topic. As shown in Algorithm 1. we split up the Title of the Topic in a list of words[7]. Based on what words would be kept in this list, we have an additional filtering step using one of three approaches: (1) keep all words in the Title, (2) keep only the nouns, (3) keep only nouns in plural. For each of the remaining words in the list we then use WordNet to extract the lemma[8]. Each lemma is then matched against entity types that are present in the YAGO ontology. We use the lemma instead of the word itself, as the entity types in YAGO are also in the form of word lemmas. Given the retrieved YAGO entity type, we retrieve all Wikipedia categories using the *subClassOf* relation in YAGO, i.e. all Wikipedia categories of the same type that the lemma is describing. All Wikipedia categories extracted from all words in the Topic Title are then aggregated to a list of categories used for retrieval. Categories given with the Topic are ignored in this approach to ensure an objective comparison to the *naïve approach*.

Algorithm 1. Algorithm for extracting categories from the query

```
Create empty set of Wikipedia categories SWC
for all words W in Topic Title do
    if W does not satisfy filtering condition then
        Get next W
    end if
    Get lemma L of W
    Get entity type T from YAGO matching L
    Get list of Wikipedia categories C of type T
    Add C to SWC
end for
Return SWC
```

4.4 Using Wikipedia Links

Wikipedia, just like the Web, is highly interconnected. Search engines make use of link information for traditional Information Retrieval document ranking. Wikipedia pages, where each page represents an entity, has external links pointing to pages outside the Wikipedia corpus and internal links, which point to other Wikipedia entities. While external links are usually presented in a separate list at the end of the entity description, internal Wikipedia links appear inside the text. While indexing the entity pages, we have kept in the indexed text the names of the linked entities where they appear, and we have also indexed their titles in a separate field called *WikiLinks* to ensure that their importance is not lost in the entity text. In addition to the naïve approach, the contextual part of the query is searched also in the *WikiLinks* index field.

For example, some of the entities that *Nicolaas Bloembergen* links to are: *Dutch, physicist, American, Harvard University, 1948, University of Utrecht,*

[7] Stop words are not included.
[8] The canonical form of a word.

nuclear magnetic resonance, Lorentz Medal, Nobel Prize in Physics, laser spectrology, and others. There are many terms present in the list of linked entities. As the information in the linked entities field is more condensed than in the text field, linked entities matching will improve the ranking of the search results.

4.5 Using Lexical Compounds

Anick and Tipirneni [2] defined the *lexical dispersion hypothesis*, according to which an expression's lexical dispersion (i.e., the number of different compounds it appears in within a document or group of documents) can be used to automatically identify key concepts in the input document set. Although several possible compound expressions are available, it has been shown that simple approaches based on noun analysis are almost as good as highly complex part-of-speech pattern identification algorithms [1]. Lexical Compounds have been already used in different settings for refining web search queries [7]. We thus extract from simple text all the Lexical Compounds of the following form: { *adjective*? *noun*+ }. All such compounds could be easily generated for Title, Descriptions and Narratives in Topics using WordNet. Moreover, once identified, they can be further sorted depending on their dispersion within each Topic. We then use Lexical Compounds as search terms in the query, as they present the essential information in a more concise manner. We consider two approaches to using Lexical Compounds in constructing the query. The first uses only Lexical Compounds for constructing the textual part of the query, to search over all the textual index fields. In the second approach we use the text from the Topic to search over the *text* field and the extracted Lexical Compounds to search in the *wikiLinks* field, which has a structure more similar to the Lexical Compounds expression (the content of the field is basically an enumeration of entity titles).

5 Experimental Results

5.1 Experimental Setup

We performed evaluation experiments of our system using the 28 topics, used for the INEX 2007 Entity Ranking Track. The main evaluation metric used is Mean Average Precision which is defined as:

$$MAP = \frac{1}{|Q|} \sum_{i=1}^{|Q|} AP_i, \tag{1}$$

where $|Q|$ is the number of topics and AP is obtained averaging the Precision values calculated at each rank where a relevant entity is retrieved [3]:

$$AP = \frac{1}{|Rel|} \sum_{i=1}^{|Rel|} \frac{i}{rank(i)}, \tag{2}$$

where $rank(i)$ is the rank of the i-th relevant entity, and $|Rel|$ is the number of relevant entities. A score of 0 is assumed for any not-retrieved relevant document.

Table 2. Top 10 Results for Topic #67 ("Ferris and observation wheels") together with the containing Wikipedia category of each result

Rank	Entity	Most relevant Category
1	Ferris wheel	amusement rides ferris wheels
2	Tempozan Osaka Village Ferris	osaka ferris wheels
3	Sky Dream Fukuoka	ferris wheels
4	Roue de Paris	amusement rides ferris wheels
5	Helsinki pyörä	ferris wheels
6	Riesenrad	attractions in viena ferry wheels
7	Wheel	bus transit mass transit in california
8	Observation wheel	amusement rides
9	Singapore Flyer	amusement rides
10	Observation	observation

We evaluated our approaches against the *naïve approach* presented in section 4.1 which has a Mean Average Precision (MAP) value of 0.1049 and a Precision for the first 10 results (P@10) of 0.1393. Table 2 shows the first 10 results for Topic #67 ("Ferris and observation wheels") using the *naïve approach*.

5.2 Results for Categories Based Search

We performed the experiments on the evaluation dataset and we compared the algorithms which use category information (see section 4.2). The results (presented in table 3) show that using extra category information more than just the one present in the Topic improves the effectiveness (see, for example, the run using Subcategories). Moreover, the best performing approach, in terms of MAP, is obtained using the Subcategories. What we observed is that the use of Siblings did not improve the results. This result is explainable by the fact that in the used Topics, no Siblings categories were found.

Another observation of the results we can do is that the YAGO ontology is up-to-date and does not match some of the categories present in the XML Wikipedia dataset, which is a crawl of 2005, used in the experiments, and thus the evaluation assessments might not consider relevant information which is present today in YAGO.

Table 3. Mean Average Precision and Precision over the first 10 results for *Categories Based Search*

Algorithm	MAP	P@10
Naïve approach	0.1049	0.1393
Naïve approach + Subcategories	**0.1238**	**0.1393**
Children	0.1165	0.1321
Siblings	0.1049	0.1393
Children + Siblings	0.1165	0.1321

Table 4. Mean Average Precision and Precision over the first 10 results for *Categories from Query*

Algorithm	**MAP**	**P@10**
Title only search	0.0477	0.0679
Categories from whole Title	0.0278	0.0393
Categories from nouns in Title	0.0330	0.0500
Categories from plurals in Title	**0.0670**	**0.1071**

5.3 Results for Searching Using Categories from Query

Using the category information as extracted from the user keyword queries, we performed different runs using the combinations presented in table 4. We can see that using the information extracted from the user query as category information is, as expected, not as good as the category information that the user provides manually. If we compare the effectiveness with a run made using only the keyword query (i.e., the Title part of the Topic) we can see that the extraction of category information from the entire Title and that the use of nouns as category representative are not improving the results. Using this approach some non relevant categories could be added to the query. For example, in the topic *Countries where I can pay with Euros* the only term recognized as noun which is a YAGO type was *can* which pointed to the Wikipedia category *Beverage cans* which is obviously making the system retrieve non relevant entities. But we can see that the plurals is, instead, a better choice for extracting category information from a keyword query.

5.4 Results for Lexical Compounds and Links

In order to evaluate the approaches based on syntactic information we extracted the Lexical Compounds from the Topic and we performed several comparisons. The results are presented in table 5. The simple extraction of Lexical Compounds from the Topic does not show improvements. But it is possible to see that the combined usage of Lexical Compounds and the full Topic textual information (Topic, Description, and Narrative) and the exploitation of the link structure performs better than the baseline.

We also used the internal link structure within Wikipedia in order to improve the results of the search for entities. From the results presented in table 5 we can see that the simple search in the outgoing links of a page improves the effectiveness by 28% in terms of MAP. Combining the most promising approaches (i.e., searching in the outgoing links and using Lexical Compounds together with Topic terms) improves the results even more.

5.5 Results for the Combinations of the Best Approaches

After the lesson learned from the previous experiments about which approaches perform best, we evaluated combinations of those methods in order to see if the effectiveness improves.

Table 5. Mean Average Precision and Precision over the first 10 results for *Lexical Compounds* and Wiki *Links*

Algorithm	MAP	P@10
Naïve approach	0.1049	0.1393
Naïve approach + Wiki Links	**0.1342**	**0.1607**
Lexical Compounds	0.0601	0.0964
Lexical Compounds + Wiki Links	0.0701	0.1000
Lexical Compounds + Full Topic text	0.0633	0.0929
Lexical Compounds + Full Topic text + Wiki Links	**0.1157**	**0.1321**

Table 6. Mean Average Precision and Precision over the first 10 results for *Combinations of the best approaches*

Algorithm	MAP	P@10
Naïve approach	0.1049	0.1393
Naïve approach + Subcategories	0.1238	0.1393
Naïve approach + Lexical Compounds + Wiki Links	0.1157	0.1321
Naïve approach + Wiki Links + Subcategories	**0.1545**	0.1643
Naïve approach + Lexical Compounds + Wiki Links + Subcategories	0.1492	**0.2071**

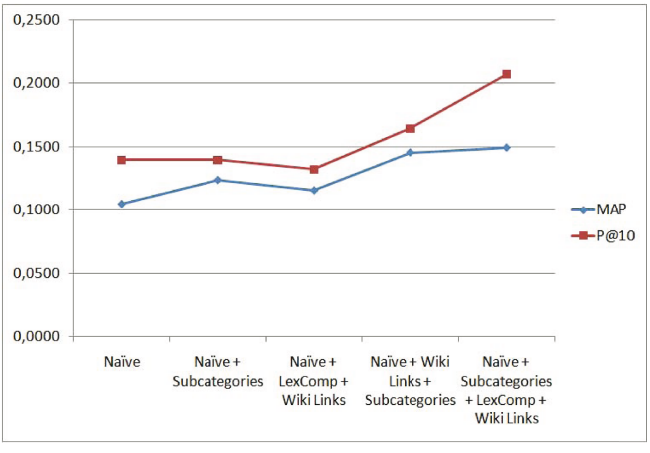


Fig. 2. Comparison to the baseline in terms of Mean Average Precision and Precision over the first 10 results for *Combinations of the Best Approaches*

The results presented in table 6 and figure 2 show that the use of Subcategories, Lexical Compounds, and internal Links can highly improve search, managing to profit from all the composing algorithms. In order to evaluate the impact on the users, we can look at the value of the $P@10$ which focuses on evaluating the performances of the system at the top of the retrieved list (i.e., the part

of the results that the user would care most about) while MAP evaluates the overall ranking generated by the system.

6 Conclusions and Future Work

In this paper we proposed several algorithms to rank entities in Wikipedia. Our approach uses a structured inverted index to represent the entities which are present in Wikipedia and uses the YAGO ontology in order to rewrite the user's query for improving the effectiveness of the results. Moreover, we evaluate the effectiveness of extracting category information from keyword queries. We investigated the usage of Lexical Compounds and link structure information. We have extracted Lexical Compounds from the Topics' textual information which we used in the Wikipedia link information due to their matching structure. The evaluation experiments show that the use of a highly accurate ontology for refining the category information in the query improves the effectiveness of the system. The *categories based search* obtains a 11% improvement in MAP over the *naïve approach*. In our experiments we have also shown that the most effective way to extract category information from a keyword query, among the tested ones, is to use plurals as category representatives. We have shown that enhancing the query with Lexical Compounds together with links and semantic category information gives the best results among the evaluated solutions with an improvement in MAP of 42% over the baseline. As main results of this paper we show that the use of simple NLP, Link Analysis and semantic techniques improves the effectiveness of entity ranking.

A first approach worth investigating is disambiguation of the query Topic with more advanced NLP techniques. Secondly, in the Wikipedia pages one can often find lists of other entities. Some Wikipedia pages are entirely structured as lists of entities (e.g., *List of European countries*). We could automatically enrich the category information by using these lists. Due to differences in used language models, Topics do not necessarily have the same vocabulary as Wikipedia. Therefore, a further step would be to extend Wikipedia page contents with linked Web pages. Finally, we will perform the evaluation on more extensive datasets, using entity classification and detection approaches.

References

1. Allan, J., Raghavan, H.: Using part-of-speech patterns to reduce query ambiguity. In: Proceedings of the 25th ACM SIGIR Conference (2002)
2. Anick, P.G., Tipirneni, S.: The paraphrase search assistant: Terminological feedback for iterative information seeking. In: Proceedings of the 22nd ACM SIGIR Conference (1999)
3. Baeza-Yates, R., Neto, R.: Modern Information Retrieval. ACM Press, New York (1999)
4. Bast, H., Chitea, A., Suchanek, F., Weber, I.: Ester: efficient search on text, entities, and relations. In: Proceedings of the 30th ACM SIGIR Conference (2007)

5. Cheng, T., Yan, X., Chang, K.C.-C.: Entityrank: Searching entities directly and holistically. In: Proceedings of the 23rd International Conference on Very Large Data Bases (2007)
6. Chernov, S., Iofciu, T., Nejdl, W., Zhou, X.: Extracting semantics relationships between wikipedia categories. In: First Workshop on Semantic Wikis, Proceedings, co-located with the ESWC 2006 (2006)
7. Chirita, P.-A., Firan, C.S., Nejdl, W.: Personalized query expansion for the web. In: Proceedings of the 30th ACM SIGIR Conference (2007)
8. Fu, L., Wang, H., Zhu, H., Zhang, H., Wang, Y., Yu, Y.: Making more wikipedians: Facilitating semantics reuse for wikipedia authoring. In: Proceedings of the 6th International Semantic Web Conference (2007)
9. Kasneci, G., Suchanek, F.M., Ifrim, G., Ramanath, M., Weikum, G.: NAGA: Searching and Ranking Knowledge. In: Proceedings of the 24th International Conference on Data Engineering (2008)
10. Kasneci, G., Suchanek, F.M., Ramanath, M., Weikum, G.: How naga uncoils: searching with entities and relations. In: Proceedings of the 16th International World Wide Web Conference (2007)
11. Pasca, M.: Weakly-supervised discovery of named entities using web search queries. In: Proceedings of the 16th Conference on Information and Knowledge Management (2007)
12. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th International World Wide Web Conference (2007)
13. von Brzeski, V., Irmak, U., Kraft, R.: Leveraging context in user-centric entity detection systems. In: Proceedings of the 16th Conference on Information and Knowledge Management (2007)

Discovering Pathways of Service Oriented Biological Processes

George Zheng[1] and Athman Bouguettaya[2]

[1] Virginia Tech, Blacksburg, Virginia, USA
gzheng@vt.edu
[2] CSIRO ICT Centre, Canberra, ACT, Australia
athman.bouguettaya@csiro.au

Abstract. Service oriented modeling and deployment of biological processes allow these processes to be published, discovered and, most importantly, invoked on the Web. This in-place invocation capability provides the basis for simulation based validation and predictive analysis of discovered pathways linking published services. We present a Web service mining framework that enables the discovery and validation of such pathways. In our experiment, we model the semantic interfaces of biological processes as WSML Web services and deploy them through the execution environment WSMX. We show how pathways involving these processes are discovered and simulated using our service mining approach.

1 Introduction

The Web is currently going through a transformation from a data-centric Web to a Semantic Web consisting of both self-describable data and a new type of *first class* objects called *Web services*. The Web service deployment of previously isolated applications allows such an application to be described and published by one organization (i.e., service provider), and discovered and invoked later by other *independently* developed applications (i.e., service consumers) [16, 17].

One use case of this new paradigm is the service oriented process modeling and deployment of biological processes. Biological entities such as cell, DNA, RNA and enzyme all have attributes and processes. These attributes and processes are currently stored in various databases (e.g., GenBank [32], DIP [37], KEGG [28, 30], Swiss-Prot [9], and COPE [27]) using predominantly free text annotations and narratives [21, 22] targeted towards human comprehension. Computer models of biological processes, on the other hand, have also been pursued (e.g., [3, 4, 20, 24, 29, 36]). However, they are usually constructed to simulate entities in an isolated local environment, limited to the study of known pathways (e.g., cell death, growth factor activated kinase in BPS [3]), and lack the ability to facilitate the discovery of new pathways. A service oriented modeling and deployment strategy would bridge the gap between these two representation approaches, fostering better sharing (i.e., discovery as well as execution) of biological processes originated from various sources as such sharing can now

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 189–205, 2008.

be made possible through standard Web service interfaces. Using this strategy, biological processes are modeled as Web service operations. An operation may consume some input substance meeting a set of preconditions and then produce some output substance as a result of its invocation. Some of these input and output substances may themselves carry processes that are known to us and thus can be also modeled and deployed as Web services. Domain ontologies containing definition of various entity types would be used by these Web services when referring to their operation inputs and outputs. This service oriented process modeling and deployment strategy not only allows for the identification of *pathways* [1] linking processes of biological entities, as do existing natural language processing approaches (e.g., [26, 33]), but would more importantly bring about unprecedented opportunity for validating such pathways right on the Web through direct invocation of involved services. When enough details are captured in these process models, this in-place invocation capability presents an inexpensive and accessible alternative to existing *in vitro* and/or *in vivo* exploratory mechanisms.

In [39], we presented a service mining framework that enables the discovery of pathways linking service oriented models of biological processes. Due to lack of execution environment of Semantic Web services at that time, we had to use a database of service metadata as a starting point with the assumption that such a database can be created by pulling information from registry of Web services. Consequently, we were only able to demonstrate in our experiment how biological pathways linking Web service models can be identified and partially verified. In this paper, we adapt our mining algorithms in the framework and apply them directly to real Semantic Web services. We show in our experiment not only how biological pathways can be identified, but also simulated through invocation of involved services. Such simulation allows for validation as well as 'what-if' analyses for testing out various hypotheses.

In the remainder of this paper, we first describe in Section 2 our mining framework used to discover biological pathways of service oriented processes. In Section 3, we introduce how we model biological processes as Web services and the runtime environments used to support their deployment and execution. In Section 4, we discuss adaptations made in our mining algorithms to work with WSML for pathway discovery purposes. We also introduce an improved rendering scheme using GraphML [5] and yEd [15] for displaying discovered pathways. In Section 5, we present our simulation algorithm used to invoke Web services involved in discovered pathways for validation and predictive analysis purposes. We then present and discuss our simulation results from applying this algorithm. We discuss related works in Section 6. We conclude the paper in Section 7 with discussion on future work.

[1] Pathways are represented as a network of interactions among biological entities such as cell, DNA, RNA and enzyme. Exposure of pathways are expected to deepen our understanding of how diseases come about and help expedite drug discovery for treating them.

2 Framework for Discovering Pathways of Service Oriented Processes

Our service mining framework for pathway discovery as shown in Figure 1 starts first with *scope specification*, a manual phase involving a domain expert defining the context of mining. The domain expert would use this phase to express an interest in mining pathways involving certain functional areas (e.g., cell enzyme and drug functions) and locales of these functions (e.g., heart, brain, etc. where these functions reside). Within this phase, a hierarchy of *domain ontology indices* is established to speed up later phases in the mining process.

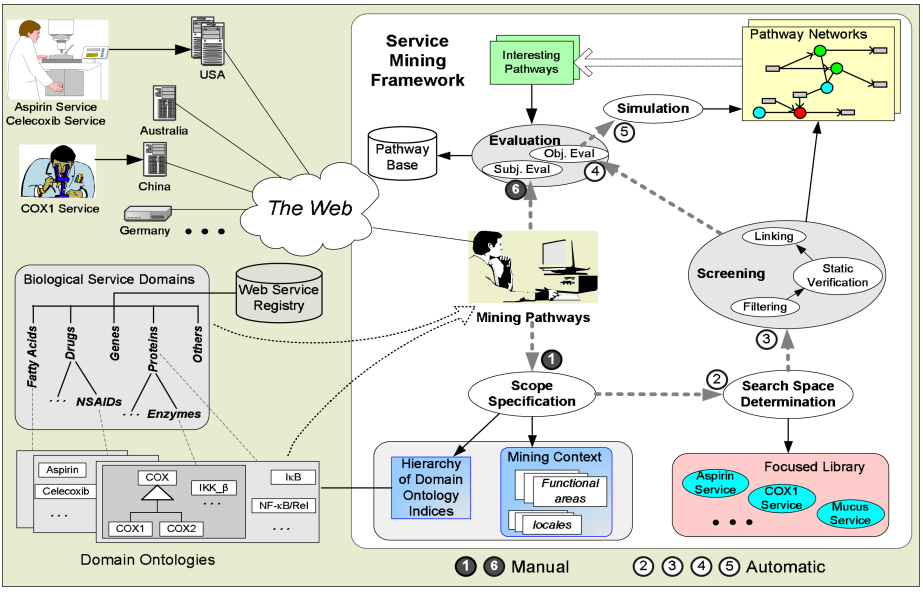


Fig. 1. Service Mining Framework

Scope specification is followed by several automatic phases. The first of these is *search space determination*, where the mining context is used to define a focused library of existing Web services as the initial pool for further mining. The next is the *screening* phase, where Web services in the focused library would go through filtering algorithms for the purpose of identifying potentially interesting leads of service compositions or pathway segments. This is achieved through establishing linkages between Web services via only a subset of matching Web service characteristics (e.g., operation parameters) that can be quickly processed. These pathway segment leads are then semantically verified based on a subset of operation pre- and post-conditions involving binary variables (e.g., whether the input to an operation is activated) and enumerated properties (e.g., the locale of an operation input). Finally, verified pathway segment leads are linked together

using our link algorithms for establishing more comprehensive pathway leads. Discovered pathways from the screening phase are then input to the *evaluation* phase, where two types of evaluation take place. The first type involves the use of an objective function. It checks whether any composition involved in a pathway has not been previously discovered, and whether the pathway itself is established in a surprising way (e.g., if the pathway links segments not previously known to be related). The *simulation* phase, which attempts to invoke involved Web services, is then applied to these potentially interesting pathways as identified by the objective function. Results from the simulation phase are expected to reveal hidden relationships among biological processes. Potentially interesting pathways and their corresponding results from executing the simulation phase are presented to the second portion of the evaluation phase, where the expert may rely on additional subjective knowledge to determine the interestingness of these pathways and devise 'what-if' scenarios to test out various hypotheses. At the end, the expert may want to introduce some of the discovered pathways to a pathway base for future references. One use of such references may be in the area of building models for biological entities at a more complex level.

Details of scope specification, search space determination, screening and objective function evaluation are covered in [39, 40]. In this paper, we describe how our adapted mining algorithms can be applied directly to real Semantic Web services described in WSML and deployed via WSMX. We focus on showing the simulation phase of our mining framework.

3 Web Service Description and Deployment of Biological Processes

To demonstrate the potential of our service mining framework, we need to first define biological processes, model and deploy them as Web services. We discuss in this section how biological processes are described in WSDL/WSML and deployed as Web services using WSMX. We start by describing the conceptual service oriented models of these processes.

3.1 Conceptual Service Oriented Models

We compiled a list of process models based on [2], [8], [18] and [38]. In addition to describing process models, these sources also reveal some simple relevant pathways that can be manually put together. We show examples of process models and corresponding simple pathways in Figure 2. Figure 2 (a) shows three types of service sources providing the functionality of activating IKK-β, which, once activated, provides a service as well (details not shown here). Figure 2 (b) shows that Aspirin can acetylate both COX1 and COX2, blocking their respective services. It can also bind IKK-β, consequently disabling its service. Figure 2 (c) shows that NF-κB/Rel when not phosphorylated can translocate from cytoplasm to cell nucleus, where it can stimulate proinflammatory gene transcription. This service, however, can be blocked by the IκB service if NF-κB/Rel is bound by its corresponding operation. Figure 2 (d) shows that PLA2

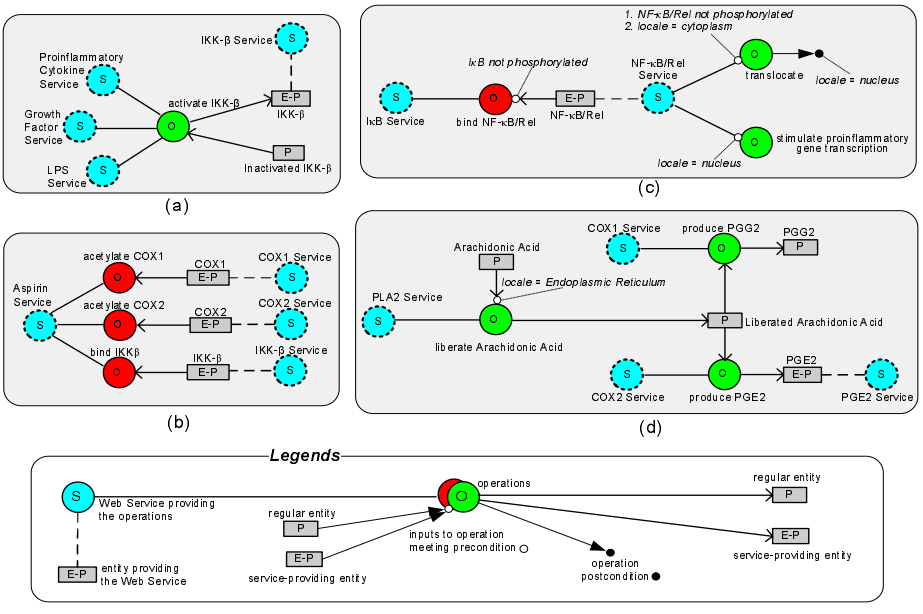


Fig. 2. Examples of Conceptual Process Model and Simple Pathway

service can liberate arachidonic acid, which can then be consumed by either COX1 to produce PGG2 or COX2 to produce PGE2. We use process models such as those in Figure 2 as references when developing WSDL and WSML Web services in Section 3. We also use simple pathways manually constructed here as references when we check the correctness of pathways automatically discovered in Section 4 using our mining algorithms.

We expect that biological processes modeled in this fashion will initially have incomplete details based on lab discoveries. As our knowledge increases and the modeling techniques continue to mature, the completeness of these models will also be increased accordingly. Instead of trying to solve the issue of model completeness, which is by itself an active research topic, we focus on how Web services can be used as a vehicle to describe and expose aspects of biological processes that we already know how to model.

3.2 WSDL/WSML Description and Deployment

In this section, we present an overview of our service representation strategy and deployment environment. Figure 3 depicts the runtime infrastructure used to support the execution of various components in our experiment. First, we model the actual process details for each type of biological entity as a WSDL [11] service and deploy these services using a Jetty Web server [7] (Section 3.3). Next, we expose the semantic interface (i.e., ontological types of operation input and output, pre- and post-conditions) of each WSDL service using Web Service

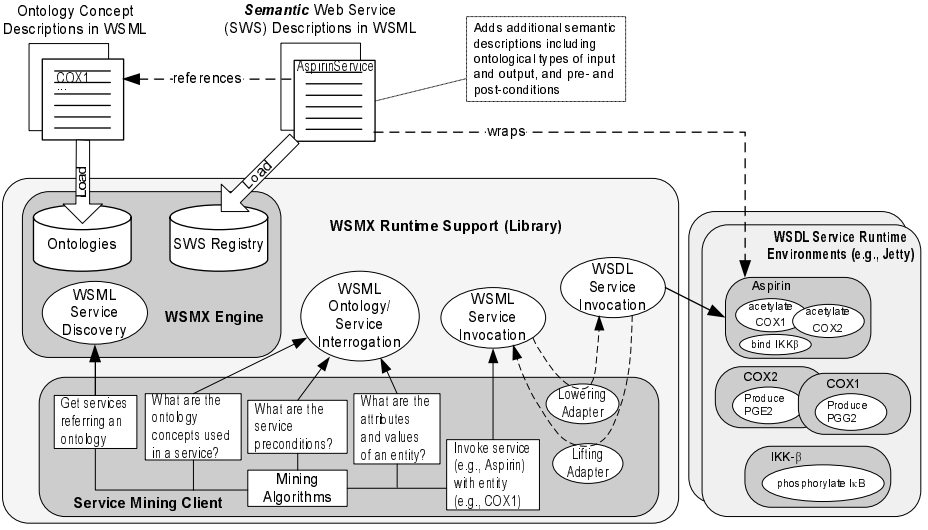


Fig. 3. Service Representation and Runtime Infrastructure

Modeling Language (WSML) [10] as a Semantic Web service (Section 3.4). These WSML services are then deployed and discovered at runtime using the Web Services Execution Environment (WSMX) [12]. A WSMX engine is used to host the WSML services. It provides runtime interfaces to our service mining client, which aims at the discovery of interesting pathways linking relevant WSML services. Both lowering and lifting adapters (Section 3.5) are provided by the mining client to convert ontological entity instances used by WSML services to/from SOAP messages used by WSDL services.

3.3 WSDL Service Modeling of Biological Processes

We first define an XML schema (Figure 4 (a)) containing generic types such as *InputSubstance*, *OutputSubstance* and *BooleanResponse*. These types are intended to be used by biological process models to represent their input and output substances. For example, type *OutputSubstance* contains information about an output substance type, location and an generic boolean flag that can be used for passing additional information (e.g., the output of *activate IKK-β* in Figure 2 (a) would have this flag set as activated). In some cases, an operation may simply return a boolean response indicating whether the corresponding process represented by the operation has been invoked successfully. The XML schema is then run through an *xjc* [6] compiler to generate corresponding bean classes. We define each process model with a Java class based on these bean classes. Using Axis2 [1] running inside a Jetty Web server [7], these Java classes are exposed as WSDL Web services at runtime.

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/
XMLSchema"
  xmlns="http://servicemining.org/"
  targetNamespace="http://servicemining.org/">

  <xsd:complexType name="InputSubstance">
    <xsd:sequence>
      <xsd:element name="type" type="xsd:string"
minOccurs="1" maxOccurs="1" />
      <xsd:element name="location" type="xsd:string"
minOccurs="1" maxOccurs="1" />
      <xsd:element name="flag" type="xsd:boolean"
minOccurs="1" maxOccurs="1" />
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="OutputSubstance">
    <xsd:sequence>
      <xsd:element name="type" type="xsd:string"
minOccurs="1" maxOccurs="1" />
      <xsd:element name="location" type="xsd:string"
minOccurs="1" maxOccurs="1" />
      <xsd:element name="flag" type="xsd:boolean"
minOccurs="1" maxOccurs="1" />
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="BooleanResponse">
    <xsd:sequence>
      <xsd:element name="result" type="xsd:boolean"
minOccurs="1" maxOccurs="1" />
    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```

(a) Schema

```
...
ontology AdapterOntology

concept xml2wsmlmapping
     instanceMappings impliesType  (1 *) _string
     valueMappings impliesType  (1 *) _string
     conceptOutput impliesType  (1 1) _string
     inputMessage impliesType  (1 1) _string

instance activateIKKBetaResponse memberOf
xml2wsmlmapping
     valueMappings hasValue { "//
location=locale(_string)", "//flag=activated"}
     conceptOutput hasValue
"ProteinOntology#IKK_beta"
     inputMessage hasValue "//ns1:IKKBeta"

instance acetylateCOX2Response memberOf
xml2wsmlmapping
     valueMappings hasValue { "//result=result"}
     conceptOutput hasValue "CommonOntology#Bool"
     inputMessage hasValue "//ns1:BOOL"

instance recruitNeutrophilResponse memberOf
xml2wsmlmapping
     valueMappings hasValue { "//
location=locale(_string)", "//
flag=localeInjured"}
     conceptOutput hasValue
"CellOntology#Neutrophil"
     inputMessage hasValue "//ns1:Neutrophil"
...
```

(b) Adapter Ontology

Fig. 4. Schema for WSDL Services and Adapter Ontology for Lifting Adapter

3.4 WSML Service Wrapping of WSDL Service

Although the internal details of biological processes can be modeled as WSDL Web services, WSDL itself does not provide elaborate mechanism for expressing the pre- and post-conditions of service operations. WSDL also lacks the semantics needed to unambiguously describe data types used by operation input and output messages. We choose WSML [10] among others (e.g., OWL-S [16], WSDL-S [14]) to fill this gap due to the availability of WSMX, which supports the deployment of ontologies and Web services described in WSML. We categorize biological entities within our mining context into several ontologies. These include *Fatty Acid*, *Protein*, *Cell*, and *Drug*. They would all refer to a *Common* ontology containing generic entity types such as *Substance*, the root concept of all entity types. We use *UnknownSubstance* as a placeholder for process inputs that are not fully described in the literature. For our experiment, we also create a *Miscellaneous* ontology capturing definitions of entity types found in the literature that don't seem to belong to any domain. Figure 5 lists snippets of three of the ontologies described in WSML.

Using these ontologies, we then wrap the semantic interfaces of existing WSDL services as WSML services. WSML supports the descriptions of pre- and post-conditions in the capability section and the ontological type description in the interface section. Figure 6 gives an example of each for the IκB service. The capability section states for the precondition that the input entity instance named $nfkbr$ should be of type NF_kappaB_Rel (defined in the protein ontology).

```
...
ontology CommonOntology
concept Substance
  locale ofType _string

concept UnknownSubstance subConceptOf Substance
  localeInjured ofType _boolean

concept Bool
  result ofType _boolean

instance uks memberOf UnknownSubstance
  locale hasValue "inflammation"
  localeInjured hasValue true
...
```

CommonOntology.wsml

```
...
Ontology CellOntology
concept Cell subConceptOf co#Substance
  nfp
    dc#description hasValue "concept of a cell"
  endnfp
  localeInjured ofType _boolean

concept Neutrophil subConceptOf Cell
  nfp
    dc#description hasValue "white blood cell"
  endnfp

concept Stomach_Cell subConceptOf Cell
  nfp
    dc#description hasValue "concept of a
StomachCell"
  endnfp
  coveredByMucus ofType _boolean
...
instance stomachCell memberOf Stomach_Cell
  coveredByMucus hasValue false
instance neutrophil memberOf Neutrophil
  localeInjured hasValue false
...
```

CellOntology.wsml

```
...
ontology ProteinOntology

concept Protein subConceptOf co#Substance
  nfp
    dc#description hasValue "concept of a Protein"
  endnfp

concept NF_kappaB_Rel subConceptOf Protein
  nfp
    dc#description hasValue "A heterodimer composed
of two DNA-binding subunits: NF-kappa B1 and relA."
  endnfp
  phosphorylated ofType _boolean

concept I_kappaB subConceptOf Protein
  nfp
    dc#description hasValue "Family of inhibitory
proteins which bind to the rel family of
transcription factors and modulate their activity"
  endnfp
  phosphorylated ofType _boolean

concept Enzyme subConceptOf Protein
  nfp
    dc#description hasValue "concept of an Enzyme"
  endnfp

concept COX subConceptOf Enzyme
  nfp
    dc#description hasValue "concept of a COX"
  endnfp
  acetylated ofType _boolean

concept COX1 subConceptOf COX
  nfp
    dc#description hasValue "concept of a COX1"
  endnfp

concept COX2 subConceptOf COX
  nfp
    dc#description hasValue "concept of a COX2"
  endnfp
...
```

ProteinOntology.wsml

Fig. 5. Example Ontologies of Entity Types

nfkbr's locale should be cytoplasm and it should not be phosphorylated. The interface section states that input entity *NF_kappaB_Rel* has grounding with the *translocate* operation of the corresponding WSDL service. The output from this service operation should be mapped to *NF_kappaB_Rel* as defined in the protein ontology.

3.5 Lowering and Lifting Adapters

When a WSML service is to be invoked, a *lowering adapter* (right end of service mining client at the bottom of Figure 3) is used to parse out the attribute values of the input entity and package them into a SOAP message, which is in turn used to invoke the corresponding WSDL service. Upon receipt of the return SOAP message from the WSDL service after its invocation, a *lifting adapter* is used to sift from it attribute values that are subsequently used to create an instance of an ontological entity type as specified in an adapter ontology (Figure 4 (b)), which provides the mappings for how such conversion can be made. For instance, the adapter ontology states that the *IKKBeta* field in the return SOAP message of the *activateIKKBeta* operation of the corresponding WSDL

```
capability translocate

  precondition
    definedBy
      ?nfkbr memberOf NF_kappaB_Rel[
      locale hasValue ?l,
      phosphorylated hasValue ?p] and
      (?l = "cytoplasm") and
      (?p = false).

  postcondition
    definedBy
      ?nfkbr memberOf NF_kappaB_Rel[
      locale hasValue ?l] and
      (?l = "nucleus").
```

```
interface NF_kappaB_RelServiceInterface

...
  importsOntology
  {
     po#ProteinOntology
  }

  in
    concept po#NF_kappaB_Rel withGrounding _"http://
servicemining.org:8001/
NFkappaBRel?wsdl#wsdl.interfaceMessageReference(NFkappaBRel/
translocate/in0)"

  out
    concept po#NF_kappaB_Rel
```

Fig. 6. Semantic Interface Description in WSML

service should be mapped to concept *IKK_beta* defined in the protein ontology. In addition, the *location* field should be mapped to the *locale* attribute, and the *flag* field should be mapped to the *activated* field. Using this mechanism, an entity instance (e.g., liberated arachidonic acid in Figure 2 (d)) created after invoking a WSML service operation (e.g., *liberateArachidonicAcid* of the PLA2 Service) can be instantiated accordingly and in turn used as an input to another WSML service operation (e.g., *producePGG*2 of the COX1 Service).

4 Pathway Discovery and Representation

The screening phase of our framework contains three steps [39]. The first involves the use of a publication and subscription based *filtering* algorithm to identify pathway segments. The next is a *static verification* algorithm, which is used for eliminating false segments based on binary (or boolean) variables (e.g., whether an entity is activated) and enumerated properties (e.g., cytoplasm, nucleus for the locale of an entity). The last is a *link* algorithm used to link pathway segments into pathway leads. We have made slight adaptations on these algorithms so they can be applied directly to WSML services. First, we have added a *provider* property in the non functional properties (nfp) section of each WSML service to indicate the corresponding ontological type of an entity that can provide the service. We use this information in our algorithms to establish the relationship between a service providing entity and the service it provides. Second, our validation algorithm has been customized to work with the service interrogation APIs of the WSMX runtime library (Figure 3) for determining the overlap between the postcondition of a source operation and the precondition of a target operation. Finally, WSML allows for the specification of pre- and post-conditions for only an entire service, but not its individual operations. Thus we have to split services that each originally has multiple operations into several services so that different conditions can be individually specified for these operations. We use the name of these services to keep track of their relationship and use that information to merge these services towards the end of the screening phase.

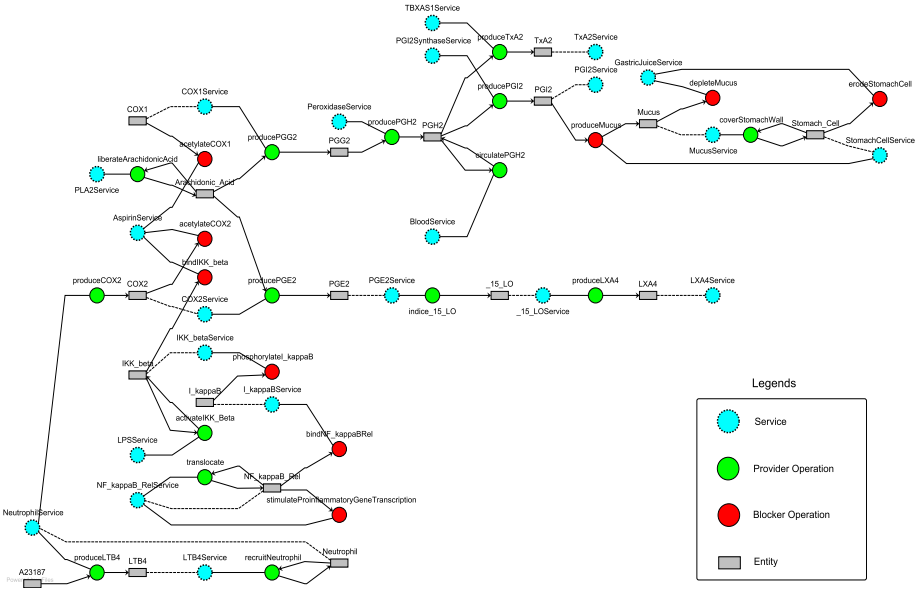


Fig. 7. Discovered Pathways Represented in GraphML and Rendered in yEd

In [39], we represented pathways discovered in the screening phase using the tree format due to its simplicity in implementation. However, this representation strategy has the inherent difficulty of merging potentially duplicate nodes in these pathways. We have since extended our rendering algorithms to represent pathways in GraphML [5], which can then be rendered and automatically arranged using yEd [15]. In Figure 7, we show pathways discovered using our adapted screening algorithms and then represented in the graph format. For brevity, we display only shortened names for nodes in the graph. We keep the full name containing either the ontological path for entity nodes or the WSML service path for both service and operation nodes in a separate description field (not shown in Figure 7). In addition, we omit pre- and post-condition details of operation linking edges such as the two forming a loop between operation *coverStomachWall* and entity *Stomach_Cell* [2]. However, we keep track of the pre- and post-conditions in our algorithm as such information along with the ontological entity paths and WSML service paths are needed when we try to invoke these services during simulation. To ensure the correctness of our algorithms, we compared segments within the automatically discovered pathway network with those constructed manually in Figure 2 and found them to be consistent in all cases. In addition, the graph shown in Figure 7 exposes previously hidden

[2] The precondition along the upper edge states that Stomach_Cell is not covered by Mucus and the postcondition along the lower edge states that Stomach_Cell is covered by Mucus.

linkages between individual services and pathway segments. For example, this graph shows that entity Neutrophil (bottom right) provides NeutrophilService (bottom left), which can produce COX2, an entity that in turn provides a service, which can produce PGE2, etc. In the meantime, COX2's service, however, can be blocked if it is acetylated by Aspirin. Aspirin can also block the service COX1 provides (top left). Such service includes the production of PGG2, which is an entity used in the production of PGH2. PGH2 is subsequently used in producing PGI2, which in turn is used in the production of Mucus, which helps protect our stomach by covering its surface. Thus it is conceivable to imagine that the blockage of COX1 by overdose of Aspirin might lead to the reduction of mucus protection, resulting in internal bleeding due to erosion from gastric juice.

5 Simulation of Pathways

In this section, we describe our simulation algorithm (Figure 8) used to enable the validation and predictive analyses on a chosen pathway network.

When an operation is to be invoked, the algorithm checks two factors. First, it examines whether all the pre-conditions of the operation are met. An operation that does not have available input entities meeting its preconditions should simply not be invoked. Second, it determines how many instances are available for providing the corresponding service. This factor is needed due to the fact that biological entities of the same type *each* has a discrete service process that

Input: Pathway Network PN, function $f()$ determining initial number of instances for an entity type, total number of iterations I, upper bound S for random number generator $random$ with uniform distribution
Output: Statistics $Stats$
Variables: entity type et, entity instance container $Container(et)$ of type et, operation op, input entity op_{in}, output entity op_{out} and precondition op_{pre} of op

```
(01) For each et ∈ PN
(02)    Container(et) ← create f(et) instances;
(03) EndFor
(04) Stats ← Tally entity quantities in each container;
(05) For each of the I iterations
(06)    For each op ∈ PN
(07)       s ← op.getProviderServce();
(08)       If random.nextInt(S) < s.getProviderEntityContainer().quantity()
(09)          If ∃op_in ∈ Container(et) : op_in matches op_pre
(10)             et ← op_in.getEntityType();
(11)             op_out ← invoke(op);
(12)             Container(et).remove(op_in);
(13)             et ← op_out.getEntityType();
(14)             Container(et).add(op_out);
(15)          EndIf
(16)       EndIf
(17)    EndFor
(18)    Stats ← Tally entity quantities in each container;
(19) EndFor
```

Fig. 8. Simulation Algorithm

deals with input and output of a finite proportion [3]. The available instances of a particular service providing entity will drive the amount of various other entities they may consume and/or produce. For this reason, the algorithm treats each entity node in a pathway network such as one shown in Figure 7 as a container of entity instances of the noted ontology type.

An initial number of instances for each entity type et are first generated based on function $f(et)$ (lines 01-03). For simplicity, we make $f(et)$ initially a constant N for all entity types in our experiment. In a more realistic setting, an expert may want to create different number of instances at the beginning for different entity types. Next, we conduct I iterations of operation invocations (lines 05-19). We take a snapshot of the quantities at the end of each iteration and before the very first iteration (lines 18 and 04). We use a random number generator to determine whether an operation should be invoked within an iteration (line 08). Using a uniform distribution, the chance of the operation getting invoked will be proportional to the quantity of the corresponding service providing entity. Upon invocation of the operation, we remove the input entity instance from the corresponding entity container (line 12) and add the output entity instance to its corresponding entity container (line 14). The simulation results are then compiled into a spreadsheet, which is used to generate a plot.

Figure 9 shows four such plots for respective values of initial number of instances N, total number of iterations I and upper bound S used in a random number generator for determining whether an operation in the iteration should be invoked. To reduce the overall crowdedness in these plots, we include only those entities that are mentioned in our subsequent discussion. Plots (a) and (b) show that the quantities of both LXA4 and 15_LO increase along iteration as they are both produced without any consumers. The increase in the quantity of PGE2 eventually plateaus as arachidonic acid, an entity needed for its production, dwindles. The same thing happens to LTB4 as A23187 is exhausted due to LTB4's own production. The quantity of stomach cell starts to reduce at the beginning due to erosion from gastric juice. That number eventually stabilizes as they are all covered and protected by mucus. [4]. Although no further reduction is happening with regard to the quantity of stomach cells, the earlier such reduction has tipped the balance of mucus production by the stomach cell service and the depletion by the gastric juice service. As a result, the quantity of mucus starts to decrease. In plots (c) and (d), we show how varying the quantity of Aspirin may have an impact on various entities involved in the pathway. When the quantity of Aspirin is low (plot (c)), the quantity of COX2 remains relatively steady as its production from the neutrophil service and consumption due to acetylation by Aspirin take place simultaneously at statistically the same

[3] This differs from traditional business service processes that are often represented as collective singletons for a given organization (e.g., credit check, loan approval).

[4] Our model assumes that a mucus covered stomach cell is immune from erosion by gastric juice. Since gastric juice can deplete mucus, such assumption may need to be revised as the process models become more complete.

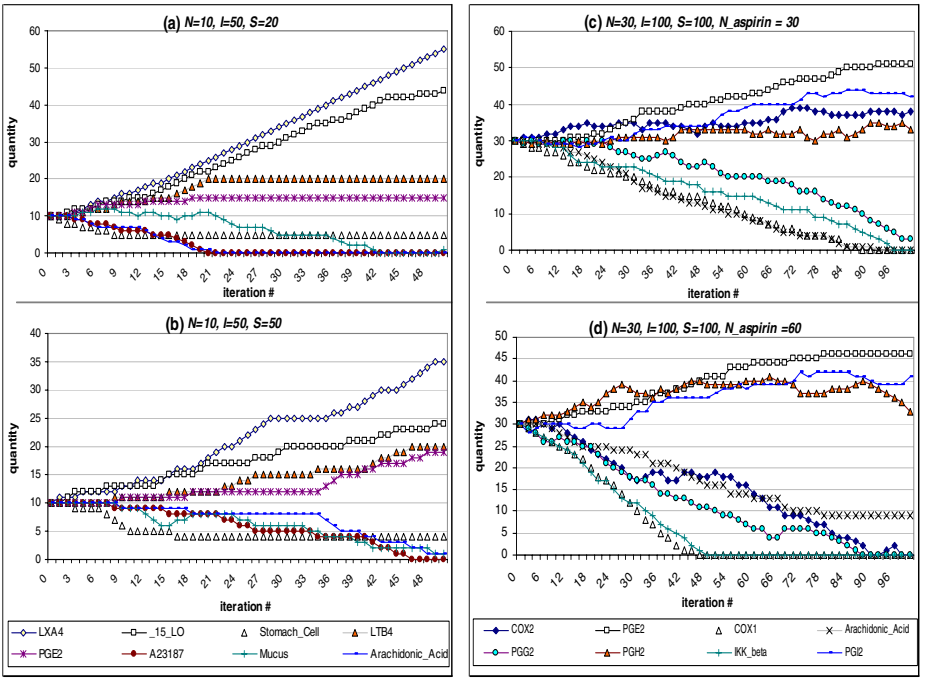


Fig. 9. Simulation Results

pace. As the quantity of Aspirin increases (plot (d)), such balance is broken. As a result, the quantity of COX2 quickly drops to zero. Since neither COX1 nor IKK_beta has any producer involved in this pathway, there is a significant increase in the rate of their reduction. However, a lag exists between the reduction of COX1 and the subsequent processes involved in the pathway as the earlier build-up of corresponding entities would still sustain them for a period of time. For example, the drop of PGH2 only starts to take place towards the end of the simulation. Another interesting observation is that as the quantity of both COX1 and COX2 approaches zero, the quantity of arachidonc acid starts to level off as it no longer has any consumers.

Simulation results such as these presented in Figure 9 provide additional information to a pathway analyst who would otherwise get such information from *in vitro* and/or *in vivo* exploratory mechanisms. The service-oriented simulation environment allows for 'what-if' analyses so that analyst can ask questions such as "what will happen if the initial quantity of a certain entity (e.g., Aspirin) is at a certain level than what has already been chosen?" and have them answered directory via service-oriented simulation. The interrelationships among various entities involved in the pathway network can now be exposed in a more holistic fashion than traditional text-based pathway discovery mechanisms, which inherently lack the simulation capability.

6 Related Work

In this section, we list several existing research areas and corresponding methodologies that are related to our work.

Web Service Composition. Web service composition has recently attracted much interest in government, business and the research community. Web service composition aims at providing value-added services through composing existing services. Governed by standards such as WSFL [13], XLANG [35], BPEL4WS [19], DAML-S and OWL-S [23], traditional Web service composition approaches are usually driven by a *top down* strategy, which first requires a user to provide a *goal* containing a fixed set of *specific* criteria. The strategy then uses these criteria to search for matching component Web services. Since the goal provided by the user already implies what type of compositions the user anticipates, the evaluation of composition *interestingness* is not a major concern in these approaches. Different from these traditional Web service composition approaches, Web service mining is driven by the desire to find *any* unanticipated and interesting compositions of existing Web services. In the absence of a top-down query, Web service mining techniques need to address how *interestingness* of service compositions can be determined. In [40], we introduced an interestingness measure used to evaluate composite services discovered via our service mining algorithms.

Log-based Interaction and Process Mining. In [25], Dustdar et. al. rely on analyzing Web service execution log data to discover potential workflow instances involving these services. While this approach may work well for business processes over time as more execution logs are expected to become available in this area, the challenge of identifying interesting workflows in the absence of such logs, especially at the time when component Web services are just introduced, is still real. Our Web service mining framework as described in [40] and this paper allows the mining of interesting composite services to be carried out in the absence of execution logs. When applied to the field of pathway discovery, where the expedience of such discovery is key to success, our approach enables the proactive discovery of interesting pathways upon the availability of these services.

Pathway Discovery via Natural Language Processing. Several natural language processing (NLP) approaches ([26, 31, 33, 34]) have been devised for the purpose of pathway identification. These approaches target free text annotations and narratives of biological entities. Since free text annotations and narratives are used mostly for human comprehension and lack the structure and interfaces required for a computer applications to "understand" what are being described, these approaches are limited to only the identification of pathways and all inherently lack the capability of verifying the validity of these pathways and 'what-if' analyses through computer-based simulation. Our approach aims at addressing all these aspects of pathway discovery.

7 Conclusion

We presented how biological processes can be modeled as Web services using WSDL and WSML. We introduced our service mining framework and how it can be applied to pathway discovery involving biological processes modeled as Web services. In particular, we described the runtime infrastructure for supporting the deployment and execution of such services. We presented our simulation algorithm aiming at exposing the potential interrelationships among processes involved in pathway networks and used to invoke these processes. The correctness of our service mining framework was checked against a relatively small set of Web service models of around 25 biological entities. We are currently working on improving the agility of our mining framework to accommodate for the dynamic expansion and evolution of WSML services. This would not only allow the framework to be checked against an expanding pool of Web services, but more importantly ensure that the results of the mining process are constantly updated to reflect the current availability and semantic description of service capabilities.

Acknowledgements

We would like to thank Maciej Zaremba from the National University of Ireland for his help on WSMX related issues.

References

1. Apache axis2/java - next generation web services, http://ws.apache.org/axis2/
2. Aspirin, http://www3.interscience.wiley.com:8100/legacy/college/boyer/0471661791/cutting_edge/aspirin/aspirin.htm
3. Bps: Biochemical pathway simulator, http://www.brc.dcs.gla.ac.uk/projects/bps/
4. Copasi, http://mendes.vbi.vt.edu/tiki-index.php?page=COPASI
5. The graphml file format, http://graphml.graphdrawing.org/
6. Java architecture for xml binding, binding compiler (xjc), http://java.sun.com/webservices/docs/1.6/jaxb/xjc.html
7. Jetty, http://www.mortbay.org/
8. Nf-kappab pathway, http://www.cellsignal.com/reference/pathway/NF_kappaB.html
9. Uniprotkb/swiss-prot, http://www.ebi.ac.uk/swissprot/
10. The web service modeling language WSML, http://www.wsmo.org/wsml/wsml-syntax
11. Web services description language (WSDL) 1.1, http://www.w3.org/TR/wsdl
12. Web services execution environment, http://sourceforge.net/projects/wsmx
13. Web Services Flow Language (WSFL), Technical report, IBM, http://xml.coverpages.org/wsfl.html

14. Web services semantics - WSDL-S, http://www.w3.org/Submission/ WSDL-S/
15. yed - java graph (ed.), http://www.yworks.com/en/products_yed_about.htm
16. Owl-s: Semantic markup for web services (November 2004), http://www.w3.org/Submission/OWL-S/
17. Web services architecture - w3c working group note (February 2004), http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/
18. Auyang, S.Y.: From experience to design - the science behind aspirin, http://www.creatingtechnology.org/biomed/aspirin.htm
19. BEA, IBM, and Microsoft. Business process execution language for web services (bpel4ws), http://xml.coverpages.org/bpel4ws.html
20. Bhalla, U.S., Iyengar, R.: Emergent properties of networks of biological signaling pathways. Science 283, 381–387 (1999)
21. Brent, R., Bruck, J.: Can computers help to explain biology? Nature 440(23), 416–417 (2006)
22. Cohen, J.: Bioinformatics: An introduction for computer scientists. ACM Computing Surveys 36(2), 122–158 (2004)
23. DAML (2004), http://www.daml.org/services/owl-s/
24. de Jong, H., Page, M.: Qualitative simulation of large and complex genetic regulatory systems. In: Proceedings of the 14th European Conference on Artificial Intelligence, ECAI, Amsterdam, pp. 141–145 (2000)
25. Dustdar, S., Hoffmann, T., van der Aalst, W.: Mining of ad-hoc business processes with teamLog, 2005. Data and Knowledge Engineering (2005)
26. Yao, D., et al.: Pathwayfinder: Paving the way toward automatic pathway extraction. In: Proceedings of the Second Conference on Asia-Pacific Bioinformatics, Dunedin, New Zealand, vol. 29, pp. 52–62. Australian Computer Society, Inc. (2004)
27. Ibelgaufts, H.: Cope - cytokines online pathfinder encyclopaedia, http://www.copewithcytokines.de/
28. Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K.F., Itoh, M., Kawashima, S., Katayama, T., Arki, M., Hirakawa, M.: From genomics to chemical genomics: new developments in kegg. Mucleic Acids Research 34, 354–357 (2006)
29. Karp, P.D., Paley, S., Romero, P.: The pathway tools software. Bioinformatics 18, S1–S8 (2002)
30. Kanehisa Laboratories. Kegg: Kyoto encyclopedia of genes and genomes, http://www.genome.jp/kegg/
31. McDonald, D.M., Chen, H., Su, H., Marshall, B.B.: Extractin gene pathway relations using a hybrid grammar: the arizona relation parser. Bioinformatics 20(182004), 3370–3378 (2004)
32. NCBI. Genbank, http://www.ncbi.nlm.nih.gov/Genbank/
33. Ng, S.-K., Wong, M.: Toward routine automatic pathway discovery from on-line scientific text abstracts, vol. 10, pp. 104–112 (1999)
34. Santos, C., Eggle, D., States, D.J.: Wnt pathway curation using automated natural language processing: combining statistical methods with partial and full parse for knowledge extraction. Bioinformatics 21(82005), 1653–1658 (2005)
35. Thatte, S.: XLANG - Web Services For Business Process Design. Technical report, Microsoft (2001), http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm
36. Tomita, M., Hashimoto, K., Takahashi, K., Shimizu, T.S., Matsuzaki, Y., Miyoshi, F., Saito, K., Tanida, S., Yugi, K., Venter, J.C., Hutchison III, C.A.: E-cell: software environment for whole-cell simulation. Bioinformatics 15(1), 72–84 (1999)

37. UCLA. Database of interacting proteins, http://dip.doe-mbi.ucla.edu/
38. Yin, M.-J., Yamamto, Y., Gaynor, R.B.: The anti-inflammatory agents aspirin and salicylate inhibit the activity of IκB kinase-β. Nature 369, 77–80 (1998)
39. Zheng, G., Bouguettaya, A.: Mining web services for pathway discovery. In: 2007 VLDB Workshop on Data Mining in Bioinformatics, Vienna, Austria (September 2007)
40. Zheng, G., Bouguettaya, A.: A web service mining framework. In: 2007 IEEE International Conference on Web Services (ICWS), Salt Lake City, Utah (July 2007)

Supporting Judgment of Fact Trustworthiness Considering Temporal and Sentimental Aspects

Yusuke Yamamoto[1], Taro Tezuka[2], Adam Jatowt[1], and Katsumi Tanaka[1]

[1] Graduate School of Informatics
Kyoto University
Kyoto, Japan
{yamamoto,adam,tanaka}@dl.kuis.kyoto-u.ac.jp
[2] College of Information Science and Engineering
Institute of Science and Engineering
Ritsumeikan University
Kyoto, Japan
tezuka@media.ritsumei.ac.jp

Abstract. We have developed a system for helping users to determine the trustworthiness of uncertain facts based on sentiment and temporal viewpoints by aggregating information from the Web. Our goal is not to determine whether uncertain facts are true or false, but to provide users with additional data on which the trustworthiness of the information can be determined. The system shows with what sentiment and in what context facts are mentioned on the Web and displays any temporal change in the fact's popularity. Furthermore, the system extracts counter facts and analyzes them in the same way. The majority of 1000 users who evaluated our system found it to be a useful tool for helping to determine the trustworthiness of facts from various viewpoints.

1 Introduction

Nowadays, much information is available from various types of media, such as newspapers, magazines, and television. Furthermore, as the Web has grown and become very popular, people have become more easily and freely able to obtain information. However, along with this facilitated access to large amounts of data, users encounter more and more untrustworthy information. It is especially evident in uncontrolled environments like the Web, where information is often created by anonymous authors. Users often encounter uncertain statements on the Web, such as "soybeans are effective for weight loss." or "Pluto is a planet." Therefore, efficient methods for checking the trustworthiness of information on the Web are becoming necessary.

Perhaps the most common and easiest way to check the information trustworthiness is to use Web search engines such as Google [1]. By inputting questionable statements into search engines, users can examine their popularity on the Web

[1] Google, http://www.google.com/

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 206–220, 2008.

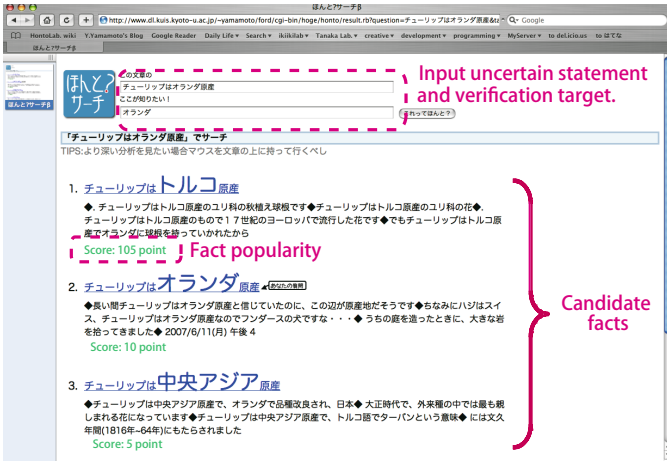


Fig. 1. System overview

and, at the same time, find out any contradictory information. However, this process is usually time consuming. To accommodate this need to check information, we have developed a system called "Honto? Search" ("Honto" means "is it true?" in Japanese) that helps users to judge the trustworthiness of statements whose reliability seems disputable [1]. When using our system, users can determine the popularity of questionable statements and their alternatives or counter examples on the Web (Fig.1).

However, in some cases a straightforward approach for estimating fact popularity produces incorrect results. This is because the context of information is not properly taken into account. For example, some pages may claim "soybeans are effective for weight loss; it is a great food!" while other pages may contain the following text "a TV station said that soybeans are effective for weight loss, however, this is a lie..." Obviously, these are counter opinions and should not be agglomerated in a straightforward way. Therefore, we extended our method in order to consider the context of encountered statements and, thus, to be able to more accurately evaluate the results. This extension involves sentiment analysis of information contained in Web pages in order to classify it as supporting or rejecting given questionable facts input by users. Except for sentiment-type context, there is also another important context of information that needs to be taken into account. While some facts are valid at any time point (i.e., "Albert Einstein was born in 1879"), many statements may be true only in certain time frames. For example, "the president of China is Hu Jintao" is a time-sensitive statement. In order to be able to precisely estimate the trustworthiness of such statements, one needs to consider temporal aspect of Web content.

In general, we believe that naive frequency-based approaches are insufficient and may judge correct facts extracted from the Web as wrong and wrong ones as correct. This drawback is especially problematic when contradicting facts have

similar levels of popularity on the Web. In such a case, their sentiment and temporal analysis would provide more insight into the characteristics and trustworthiness of these facts. It would be possible not only to more accurately select correct answers but also to provide additional contextual information related to answers, sentiment to answers, and their dynamics in society.

We have thus developed an extended search system to help users more accurately determine the trustworthiness of facts on the Web. Honto? Search system has been expanded so that it considers information from sentimental and temporal viewpoints. The system has three key factors: counter examples extraction, sentiment distribution analysis, and popularity evolution analysis of facts. Honto? Search proposes counter examples to the input fact and provides a framework for their temporal and sentimental analysis. Sentiment analysis is carried out in order to categorize Web pages containing information about a doubtful fact as positive or negative about the fact and to present final sentiment distribution. This approach is augmented with the prior construction of a large-scale sentiment term dictionary from the Web. A temporal approach is also applied to analyze changes in popularity of facts over time and to visualize them to users.

The remainder of the paper is as follows. In the next section, we discuss related work. Section 3 provides the background description of Honto? Search system. Sections 4 and 5 discuss sentiment and temporal analysis of information on the Web, respectively. In the next section, we demonstrate the experimental results conducted using our system. The last section concludes the paper and outlines our future research directions.

2 Related Work

2.1 Trustworthiness of Web Information

The problem that users have in the evaluation of Web resources from the viewpoint of trustworthiness has been recently studied by Nakamura et al. [2]. The authors have undertaken a large-scale analysis of user behaviors and expectations by conducting an online survey of Internet users. The results indicated that average Web users have difficulty in correctly estimating trust levels of encountered information. In addition, users often tend to trust the information without further analysis of its credibility. A study aiming at similar objective was made by Fogg et al. [3].

Many researchers have proposed effective ways of evaluating the trustworthiness of Web pages. PageRank is a well-known algorithm for estimating the trustworthiness of Web documents by considering their relative popularities [4]. Although the PageRank algorithm was quite effective in the past, its efficiency has been recently undermined due to the increase in link spam on the Web. Therefore, several approaches have been proposed to propagate trust among Web pages with the purpose of combating Web spam [5]. In addition, with the proliferation of user-generated content, some researchers aimed at evaluating the trustworthiness of social content. For example, by focusing on editors' activities,

trust and controversy levels of Wikipedia [2] articles were estimated [6]. In this research, we approach the problem of information trustworthiness in a more general way by agglomerating it from the Web and by considering its temporal and sentiment aspects.

2.2 Sentiment Analysis

Sentiment analysis of documents is an attractive, yet, at the same time, quite challenging task [7,8]. The potential benefits that would result from an effective opinion mining of large text collections like the Web cannot be underestimated. For example, companies would be very interested in the feedback from users of their products or services if it was automatically collected from large collections of blog data. In [8], the sentiment of content related to objects is determined using approaches borrowed from natural language processing. In [7], the authors calculate beforehand the probability that any term appears in a given sentiment class based on mining a large news corpus. By using the calculated information, sentiment levels of whole documents are then determined. Although our approach is similar to their method, it is however more general from a theoretical perspective and, also, more accurate. The target of [7,8] is documents or objects, while our system analyzes sentiments about facts.

2.3 Term Dynamics

Our system uses temporal changes in the occurrence frequency of phrases to filter out obsolete information. Kleinberg's "word burst" is a well known method for examining changes in word frequencies over time [9]. It is a state-based approach that measures term dynamics characterized by transitions between two states: low- and high-frequency. That work, however, was intended to detect significant bursts of terms in text streams, whereas in our system, we compare differences between the frequencies of phrases and their duration over time on the Web. In the area of blog mining, Mei et al. [10] proposed using a Topic-Sentiment Mixture Model to model generation of Web users' positive and negative opinions on a certain subject over time.

3 Honto? Search

In this section, we describe WebQA system called Honto? Search [1] that we developed to help users judge the trustworthiness of given information. Users input a phrase representing a fact whose trustworthiness they doubt into the system and indicate the suspicious part of the phrase. The system then provides users with a popularity estimation for the fact and for alternative or counter examples to it that occur on the Web. For example, if the user inputs "Tulips are native to the Netherlands" as an uncertain fact and "the Netherlands" as a verification target into Honto? Search, our system returns several facts. The basic

[2] http://www.wikipedia.org/

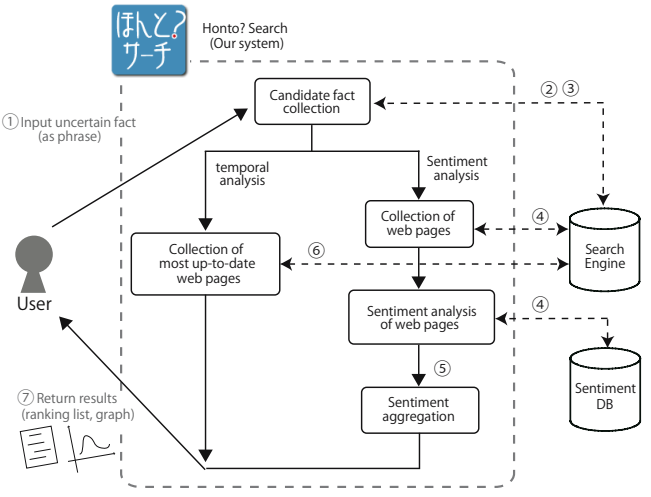


Fig. 2. System flow

idea is to extract the patterns which match "*tulips are native to *(wildcard)*" in snippets returned from a Web search engine. From the result ranking, the user can find that the most popular fact is "tulips are native to Turkey (this is the correct answer)" [3].

Fig.2 summarizes the approach in which extended Honto? Search estimates the trustworthiness of facts. The system works as follows.

1. The user inputs a questionable fact (as a phrase) and its doubtful part into the system.
2. The system divides the phrase given by the user into parts, and constructs a query that is then sent to a Web search engine.
3. The system extracts alternative or counter facts to the original phrase from the search results.
4. For sentiment analysis, the system inputs each fact into a Web search engine and collects search results.
5. The distribution of sentiment on each fact is estimated by aggregating the search results of each fact (Section 4).
6. The system sends the original fact and the counter facts to the Web search engine in order to collect time information (Section 5).
7. Using the time information, the system evaluates the original fact and candidate facts from a temporal viewpoint (Section 5).

If the user wants to know how popular a fact is, he or she can easily check the total number of Web pages that include it as a phrase by using a Web search

[3] Some people have the wrong idea that tulips are native to the Netherlands according to the English Wikipedia article on tulips (http://en.wikipedia.org/wiki/Tulip)

Table 1. Estimation of simple fact popularity

"There are 15 countries in the European Union."	
counter terms	fact popularity
25	187
15	156
10	141

"The President of China is Hu Jintao."	
counter terms	fact popularity
Hu Jintao	589
Jiang Zemin	574

engine. Our system does so automatically by sending all candidate facts as a phrase to a search engine and obtaining the total number of Web pages that include the fact. We call this number the *fact popularity*. Each fact popularity estimation is then presented to the user. By comparing the fact popularity of the original fact with the popularity of alternative facts, the user can get an idea of how strongly the fact is supported on the Web.

Table 1 shows the results of simple fact popularity analysis. In these examples, we input two facts, "there are 15 countries in the European Union" (Example 1) and "the President of China is Hu Jintao" (Example 2) to Honto? Search. Verification targets were "15" and "Hu Jintao", respectively.

Table 1 lists the frequencies of the original and counter facts in the Web search results. For example, for the fact "there are 15 countries in the European Union", we got two counter terms,"25" and "10". The most frequent one was "25", which is the correct answer. The counter term "15" also produced many results, since it was true until 2004. Additionally, the counter term "10", was also frequently reported on the Web, which must have come from expressions such as "10 new countries in the European Union". The user can judge that the original fact may not be trustworthy, since it is not the most frequent one. For the fact, "the President of China is Hu Jintao", we got an counter fact "the President of China is Jiang Zemin", which was actually true until 2003. From the table, the user can judge that the fact "the President of China is Hu Jintao" is reliable, since it is the most frequent one. These are simple estimations which do not consider the temporal aspect.

In addition to calculating simple fact popularity, Honto? Search estimates the sentiment behind the facts by analyzing context in the aggregated Web pages. The system shows the ratio of positive to negative sentiment to users. Fig.3 illustrates an example of sentiment aggregation. In this example, the fact "soybeans are effective for weight loss" was input into the system. The system extracted multiple candidate facts from the Web and performed a sentiment aggregation on them. Two interesting results are illustrated in the figure concerning two facts: "walking is effective for weight loss" and "soybeans are effective for weight loss". From the result, we can find that (1) the fact "soybeans are effective for weight loss" has more negative factors than the fact "walking is effective for weight loss", and (2) opinions on the fact about soybeans are not consistent and so we should not accept the fact.

Fig. 3. Example of sentiment aggregation. Result no. 2 is "walking is effective for weight loss" and result no. 7 is "soybean is effective for weight loss". Green and orange colors in bar graphs denote positive and negative sentiments, respectively, while purple color means "unspecified".

As we have mentioned before, the extended system estimates also fact popularity from the temporal aspect. Our system describes the evolution of fact popularity on a graph like Fig.4. In this example, the user input "Chiba Lotte became Pacific League champions" as a doubtful fact and compared the evolution graph of the fact with that of the fact, "Nippon Ham became Pacific League champions" (Chiba Lotte and Nippon Ham are Japanese baseball teams). Both facts are true, but the result graph shows how the popularity of each team differs over time.

4 Sentiment Analysis

In this section, we describe the method to aggregate sentiment about a fact on the Web. To analyze sentiment about a fact, the system operates as follows:

1. The system inputs a fact as a phrase query to a Web search engine, such as Yahoo! [4] and gets top N search results.
2. It analyzes the sentiment contained in each result page and categorizes them as one of the following: "positive", "negative" or "unspecified".
3. It illustrates the ratio of three groups to users as a bar graph.

4.1 Sentiment Analysis of Pages

Our method uses the Naive Bayes Classifier in order to categorize content as "positive" or "negative".

[4] http://www.yahoo.com/

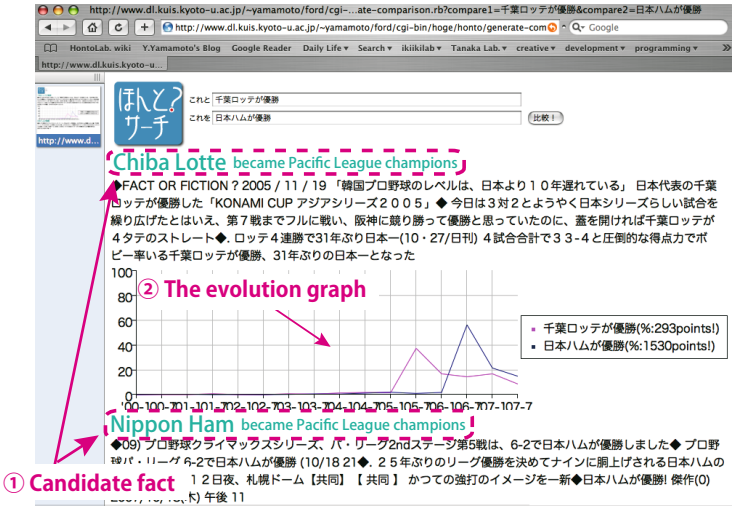


Fig. 4. Example of evolution graph. In the graph, blue line means the popularity change of "Chiba Lotte" and pink line indicates that of "Nippon Ham".

Naive Bayes Classifier. To statistically analyze the sentiment level of a text, we use a well-known statistical classifier, *naive bayes classifier*. This classifier is based on the Bayes theory and it can be speedily trained and applied. We use it on a multinomial model.

In order to use this classifier, the following assumption is made: all terms that appear in documents are independent of each other. The probability $Pr(d|c)$ that document d belongs to class c is formulated as follows:

$$Pr(c|d) = \frac{Pr(c)Pr(d|c)}{\sum_\gamma Pr(\gamma)Pr(d|\gamma)} \tag{1}$$

$$= \frac{Pr(c)Pr(L = l_d|c)Pr(d|l_{d,c})}{\sum_\gamma Pr(\gamma)Pr(d|\gamma)} \tag{2}$$

$$= \frac{Pr(c)Pr(L = l_d|c)\begin{pmatrix} l_d \\ n(d,t) \end{pmatrix}\prod_{t\in d}\theta_{c,t}^{n(d,t)}}{\sum_\gamma Pr(\gamma)Pr(d|\gamma)} \tag{3}$$

where $\begin{pmatrix} l_d \\ n(d,t) \end{pmatrix} = \frac{l_d!}{n(d,t_1)!n(d,t_2)!\ldots}$ is the multinomial coefficient. $\theta_{c,t}$ means the probability that term t appears in a document in class c. Let term t occur $n(d,t)$ times in document d, which is said to have *length* $l_d = \sum_t n(d,t)$. Using this formulation, we have to estimate $Pr(d|c)$ from training sets and prior probability. Here, class c is either "positive" or "negative". In order to estimate $Pr(d|c)$,

Table 2. Parts of sentiment term seeds

Sentiment	Term seeds
Positive	fair, moderate, appreciate, comfortable
	reliable, happy, enjoyable, honest, useful
Negative	unfair, despite, useless, uncomfortable
	unhappy, trivial, dishonest, unreliable

we use the training data sets that are collected using the method which is described later. We then smooth the parameter of $\theta_{c,t}$ in order to estimate $Pr(d|c)$ as accurately as possible [11]. Finally, in order to classify a document into the two classes, we evaluate the logarithmic likelihood ratio LR. LR is defined as

$$LR = log\frac{Pr(pos|d)}{Pr(neg|d)} \tag{4}$$

$$= log\frac{Pr(pos)}{Pr(neg)} + \frac{\sum_{t\in d} n(d,t) log\theta_{pos,t}}{\sum_{t\in d} n(d,t) log\theta_{neg,t}} \tag{5}$$

If LR is much greater than 0, document d has positive sentiment, whereas if LR is below 0, document d has negative sentiment. When the value of LR is around 0, it means that the sentiment contained in document d cannot be specified. When our system classifies documents, the threshold $\alpha_{pos/neg}$ is used, and if $|LR| >= \alpha_{pos/neg}$, our system classifies documents into positive/negative sentiment.

Training data. To make training data sets, we use the following approach. First we prepare positive term seeds and negative term seeds using 31 terms (see Table 2). These seeds are subjectively constructed. Next, we input the terms from each sentiment term seed as a query into a Web search engine and obtain search results. Then, the snippets collected with positive seeds are regarded as positive sentiment training sets and the snippets collected with negative seeds are regarded as negative sentiment training sets. This approach is based on the assumption that the prepared positive (or negative) seeds frequently appear in positive (or negative) documents. This method is a simple way to make training sets, and the Naive Bayes classifier, which is based on this approach, works well, as described in Section 4.1. Therefore, we use the approach above.

After collecting the training data sets, we count the terms' frequencies in positive and negative training data sets. In this step, only nouns, verbs, adjectives and adverbs are extracted. Finally, we calculate the probability $\theta_{pos,t}$ ($\theta_{neg,t}$) that each extracted term t appears as positive sentiment (or negative sentiment). Through the above procedure, we calculated $\theta_{pos,t}$ ($\theta_{neg,t}$) of 85,776 terms using 62,000 Web search results as training data sets.

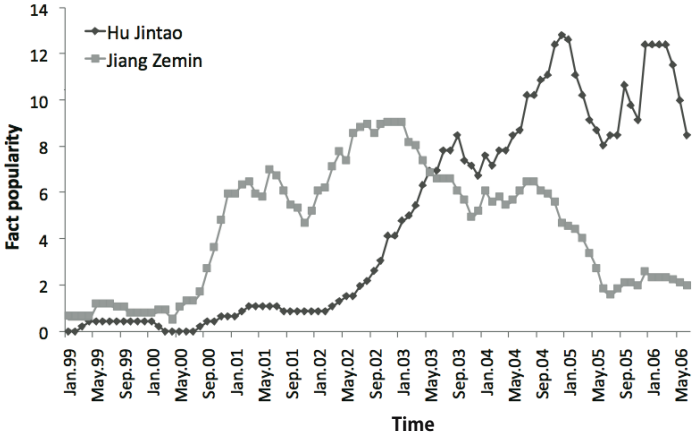


Fig. 5. Example of temporal evolution

5 Visualization of Temporal Evolution

Since a simple fact popularity does not take temporal factors into account, it is often not appropriate to use it for trustworthiness judgement. For example, consider the fact, "the Chinese president is Mr. Jiang Zemin". This fact was correct only until 2003. This example shows that the trustworthiness of a fact is strongly dependent on time. Our system helps users to temporally judge the trustworthiness of a fact by portraying its temporal evolution. Here, we define the popularity of fact A, FP_A, at time period t (FP is fact popularity).

Fact popularity of fact A during time period t

$FP_A(t)$: the number of Web pages that refer to fact A and were last modified during time period t

Using FP, we can estimate which fact out of several alternative facts is the most reliable for a certain period of time. That is, if we want to estimate which fact is more reliable, fact A or fact B in time period t, we only have to compare $FP_A(t)$ with $FP_B(t)$. If $FP_A(t)$ is greater than $FP_B(t)$, we can estimate that fact A was truer in period t than fact B. We calculate popularity for the fact, which is the fact the user inputs into our system, and $FP(t)$ for alternative facts, which our system generates from the original fact. We then identify the fact for which $FP(t)$ has the greatest score to be the most reliable fact during period t.

To collect Web pages with timestamps, we use the Ask.jp[5] search engine. This search engine allows the use of a temporally structured query such as "keywords between: date1, date2" for which relevant Web pages modified between date1 and date2 are returned. The time periods for which the support of facts is to be estimated can have an arbitrary length.

[5] http://ask.jp/

Fig.5 shows that although FP for the fact "the President of China is Jiang Zemin" is higher than that of "the President of China is Hu Jintao" at the beginning, they reversed around 2003. In fact, Jiang Zemin was the President of China until March 2003 and was later substituted by Hu Jintao.

6 Evaluation

In this section, we present the results of our experiment. The experiment had two main parts. The first was to estimate the efficiency of sentiment analysis by evaluating its precision and recall levels. The second part was a user test. The goal of the user test was to confirm whether or not our system is useful when users use it for checking the trustworthiness of a fact.

6.1 Precision and Recall of Sentiment Analysis

Sentiment aggregation of Web pages strongly depends on the ability to estimate the sentiment contained in a text. In this section, we evaluated the capability to estimate sentiment contained in a text using our system by comparing our sentiment analysis method with the one described in [7], which uses a preconstructed sentiment dictionary. We made test sets from 196 randomly collected blogs. They were manually categorized into three groups: "positive (58)", "negative (59)", and "unspecified (79)" (the numbers in brackets are the numbers of blogs that were categorized into each sentiment group). We evaluated the efficiency of the two sentiment analysis methods using the test sets with different levels of threshold $\alpha_{pos/neg}$. We have found the optimum values of this threshold to be between 8 and 17. For using a Naive Bayes Classifier, we set the prior probability $P(pos)$ as 0.5. That is, we assumed that positive pages and negative pages were uniformly distributed on the Web. The result is given in Table 3. $Prec_{pos}$($Prec_{neg}$) means the precision of positive (negative) sentiment decisions. $Recall_{pos}$($Recall_{neg}$) means the recall of positive (negative) sentiment decisions.

Table 3. Evaluation of sentiment decisions using our approach vs. the baseline method

$\alpha_{pos/neg}$	$Prec_{pos}$	$Recall_{pos}$	$Prec_{neg}$	$Recall_{neg}$	$Prec_{pos'}$	$Recall_{pos'}$	$Prec_{neg'}$	$Recall_{neg'}$
8	0.464	0.224	0.455	0.612	1.000	0.542	0.732	1.000
9	0.462	0.207	0.452	0.571	1.000	0.545	0.737	1.000
10	0.478	0.190	0.441	0.531	1.000	0.550	0.743	1.000
11	0.524	0.190	0.456	0.531	1.000	0.579	0.765	1.000
12	0.556	0.172	0.434	0.469	1.000	0.556	0.742	1.000
13	0.625	0.172	0.449	0.449	1.000	0.556	0.733	1.000
14	0.615	0.138	0.455	0.408	1.000	0.533	0.741	1.000
15	0.615	0.138	0.462	0.367	1.000	0.571	0.750	1.000
16	0.667	0.138	0.457	0.327	1.000	0.571	0.727	1.000
17	0.727	0.138	0.438	0.286	1.000	0.571	0.700	1.000
Baseline	0.342	0.224	0.364	0.735	0.813	0.406	0.655	0.923

These values are calculated using all the test sets, thus, also the ones whose sentiment cannot be specified (unspecified sentiment category). In order to better gauge the performance of our method, we have also evaluated the efficiency by removing test sets whose sentiment could not be specified. $Prec_{pos'}$, $Prec_{neg'}$, $Recall_{pos'}$ and $Recall_{neg'}$ mean the precision and recall of positive (negative) sentiments on the test datasets composed of only positive and negative categories (i.e. blogs whose sentiment is unspecified are not taken into account).

From Table 3 we can see that our method classifies sentiment contained in documents more accurately than the baseline method. In particular, if we do not take unspecified sentiment into account, the performance of the Naive Bayes classifier is very high (the precision is about 100%(positive)/74%(negative) and the recall is about 56%(positive)/100%(negative)). That is, there were few wrong decisions. From these results, we can notice that the problem of our system is to correctly classify ambiguous content. Therefore, we have to (1) make more exact and (2) larger test sets in order to evaluate the performance more precisely. Improving the method to select sentiment term seeds and the algorithm to estimate sentiment is a part of our future work.

6.2 User Test

In this section we report on the results of the user test we did regarding the effectiveness of our system. The user test was online and it was done between 7th and 12th February 2008 by a group of 1000 Internet users in Japan. Subjects were divided into four categories depending on their age: 20-29, 30-39, 40-49 and 50-59 years old. Each category consisted of 250 respondents, where half were males and half females. We asked the users to judge the trustworthiness of facts using Honto? Search and to give their impressions about the system in a free form. The given uncertain fact was "soybean is effective for weight loss". We did

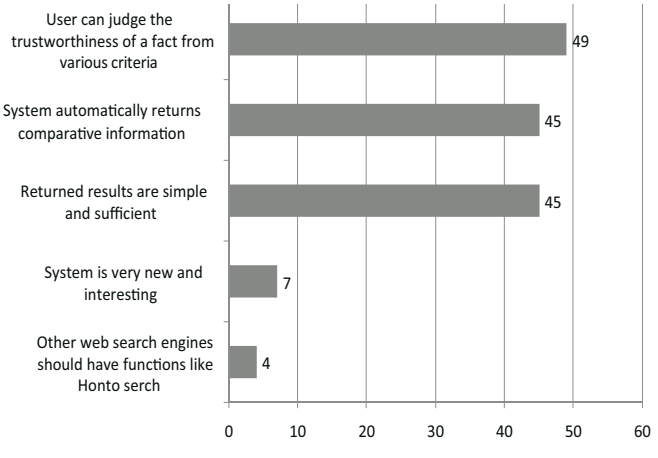


Fig. 6. Main reasons for judging our system as good

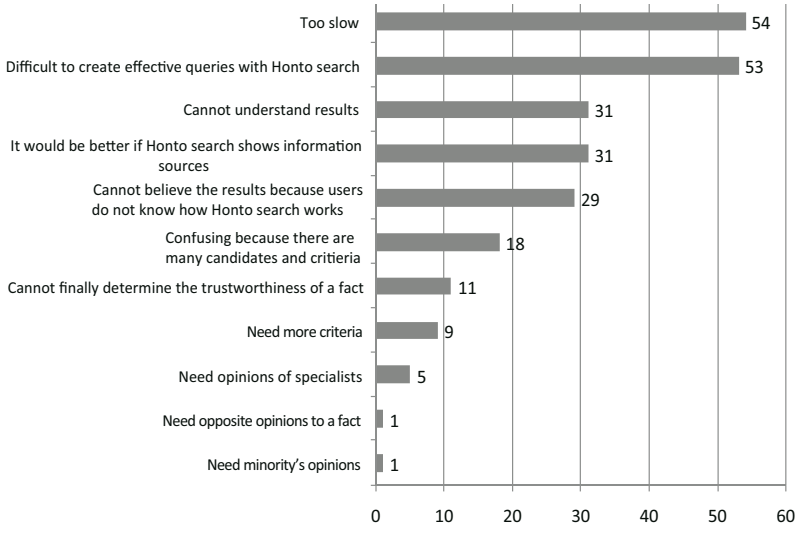


Fig. 7. Main reasons for judging our system as bad

not limit the time to make the judgement and asked the users to use the system until they could finally confirm the trustworthiness of the fact. The survey was done in the Japanese language (we show here translated results).

In summary, 44.1% of users had a good impression of Honto? Search, 31.1% had a bad impression, and 24.8% had a neutral impression. Although, more users evaluated our system as good than as bad, the number of users who evaluated our system as good is still not enough. Fig.6 summarizes the aspects that users evaluated as good and Fig.7 indicates the aspects that users evaluated as bad in our system. From Fig.6, we can see that some functions that we proposed were useful because users could judge the trustworthiness of facts quickly and concisely and make necessary comparisons. If users do the same thing using conventional Web search engines, it is time consuming. The most negative thing, on the other hand, was that our system works very slowly. It takes about 10 seconds to return results. Thus, we have to improve the response speed of our system. Focusing on the answers related to the trustworthiness, we can see that some users do not understand the results generated by the system and the mechanism in which trustworthiness of facts is estimated. It is, thus, important to provide users with the information that is easy to be understood. We plan to improve the interface to make it easier and more transparent to users.

7 Conclusion

In this paper, we have proposed a method to estimate the trustworthiness of a fact from sentimental and temporal viewpoints. We implemented a trustworthiness judgment support system, Honto? Search, based on the proposed method.

Our approach uses (1) candidate fact extraction, which enables users to compare facts, (2) sentiment aggregation, which is the summarization of sentiment about a fact on the Web, and (3) visualization of temporal evolution about a fact, which enables users to know how a fact's popularity has changed over time. With Honto? Search users can compare their query with automatically collected similar facts in order to have more confidence in their judgment. Furthermore, users can estimate sentimental and temporal characteristics of the fact, which cannot be obtained using conventional naive frequency-based approaches.

Nevertheless, Honto? Search system has some problems. In the process of extracting candidate facts, a rather naive natural language technique is employed; therefore, if a fact's meaning is the same as another fact's meaning and their spellings are different, the system returns different results. Moreover, the interface and the time required to generate the results are not yet satisfactory and therefore should be optimized. In the future, we plan to work on improving the above drawbacks and we would like also to estimate reliability of Web pages and Web sites based on a sentiment and temporal analysis.

Acknowledgement

This work was supported in part by a MEXT Global COE Program "Informatics Education and Research Center for Knowledge-Circulating Society" (Project Leader: Katsumi Tanaka), a MEXT Grant for "Development of Fundamental Software Technologies for Digital Archives", Software Technologies for Search and Integration across Heterogeneous-Media Archives (Project Leader: Katsumi Tanaka), a MEXT Grant-in-Aid for Scientific Research on Priority Areas "Cyber Infrastructure for the Information-explosion Era", Planning Research "(BContents Fusion and Seamless Search for Information Explosion" (Project Leader: Katsumi Tanaka, A01-00-02, Grant#: 18049041), the National Institute of Information and Communications Technology, a MEXT Grant-in-Aid for Young Scientists (B) "Trust Decision Assistance for Web Utilization based on Information Integration" (Project Leader: Taro Tezuka, Grant#: 18700086), and a MEXT Grant-in-Aid for Young Scientists(B) "Information Discovery Using Web Archives" (Project Leader: Adam Jatowt, Grant#: 18700111).

References

1. Yamamoto, Y., Tezuka, T., Jatowt, A., Tanaka, K.: Honto? Search: Estimating Trustworthiness of Web Information by Search Results Aggregation and Temporal Analysis. In: Dong, G., Lin, X., Wang, W., Yang, Y., Yu, J.X. (eds.) APWeb/WAIM 2007. LNCS, vol. 4505, pp. 253–264. Springer, Heidelberg (2007)
2. Nakamura, S., Konishi, S., Jatowt, A., Ohshima, H., Kondo, H., Tezuka, T., Oyama, S., Tanaka, K.: Trustworthiness Analysis of Web Search Results. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 38–49. Springer, Heidelberg (2007)

3. Fogg, B., Swani, P., Treinen, M., Marshall, J., Laraki, O., Osipovich, A., Varma, C., Fang, N., Paul, J., Rangnekar, A., et al.: What makes Web sites credible?: a report on a large quantitative study. In: Proceedings of the SIGCHI conference on Human factors in computing systems (CHI 2001), pp. 61–68 (2001)
4. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web (1998)
5. Gyongyi, Z., Garcia-Molina, H., Pedersen, J.: Combating web spam with trustrank. In: Proceedings of the Thirtieth international conference on Very large data bases. VLDB 2004, vol. 30, pp. 576–587 (2004)
6. Vuong, B., Lim, E., Sun, A., Le, M., Lauw, H.: On ranking controversies in wikipedia: models and evaluation. In: Proceedings of the international conference on Web search and web data mining (WSDM 2008), pp. 171–182 (2008)
7. Kumamoto, T., Nadamoto, A., Tanaka, K.: Design and Evaluation of News Reader "wEE" Expressing Emotional Aspects of News Articles. Trans. IEICE of Japan J90–D(2), 185–195 (2007)
8. Yi, J., Niblack, W.: Sentiment Mining in WebFountain. In: Proceedings of the 21st International Conference on Data Engineering (ICDE 2005) (2005)
9. Kleinberg, J.: Temporal Dynamics of On-Line Information Streams. Data Stream Management: Processing High-Speed Data Streams. Springer, Heidelberg (2005)
10. Mei, Q., Ling, X., Wondra, M., Su, H., Zhai, C.: Topic sentiment mixture: modeling facets and opinions in weblogs. In: Proceedings of the 16th international conference on World Wide Web (WWW 2007), pp. 171–180 (2007)
11. Chakrabarti, S.: Mining the Web: Discovering Knowledge from Hypertext Data. Morgan Kaufmann, San Francisco (2003)

Improving Mobile Web-IR Using Access Concentration Sites in Search Results

Masaya Murata, Hiroyuki Toda, Yumiko Matsuura, and Ryoji Kataoka

NTT Cyber Solutions Laboratories, NTT Corporation,
1-1 Hikari-no-oka, Yokosuka-Shi, Kanagawa, 239-0847, Japan

Abstract. Effective ranking algorithms for mobile web search are being actively pursued. Due to the peculiar and troublesome properties of mobile contents such as scant text, few outward links, and few input keywords, conventional web search techniques using bag-of-words ranking functions or link-based algorithms are not good enough for mobile web search. Our solution is to use click logs; the aim is to extract only access concentrated search results from among the many search results. Users typically click a search result after seeing its title and snippet, so the titles and snippets of the access concentrated sites must be good relevance feedback sources that will greatly improve mobile web search performance. In this paper, we introduce a new measure that is capable of estimating the degree of access concentration and present a method that uses the measure to precisely extract the access concentration sites from many search results. Query expansion with terms extracted from the access concentration sites is then performed. The effectiveness of our proposal is verified in an experiment that uses click logs and data from a real mobile web search site.

1 Introduction

Web access from mobile devices is increasing and the volume of accessible contents is exploding. Given this background, demand for more effective retrieval of mobile web sites is rising and studies on effective ranking techniques are being actively pursued.

However, due to the following properties of mobile sites, good ranking is not always assured if we simply apply conventional ranking techniques to mobile web search.

1. For mobile devices with a small screen, mobile contents are limited to small amounts of text. This implies that the volume of information is little,
2. The number of links is few and, unlike regular web sites, the links are to navigate the home site, and not point at useful sites.

Characteristic 1 implies the difficulty of ranking sites by only using bag-of-words ranking functions such as $tf \cdot idf$[1] or Okapi BM25[2]. The characteristic 2 implies that link-based ranking such as PageRank[3][4] are not likely to work well.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 221–234, 2008.

Table 1. The number of query keywords

	One	Two	Three+
Ratio(mobile goo)	77.5%	20.0%	2.5%
Ratio(AlltheWeb)[5]	53%	18%	29%

Queries for mobile web search also have the special property that few query keywords are input. In fact, our investigation of the query logs of a Japanese mobile search service site, "mobile goo"[1], showed that 80% of the queries were uniterms(single query). This result differs considerably from the query logs analysis of a conventional web search engine, "AlltheWeb", performed by Jansen[5]. It is very hard to find the information desired by users since the single word queries, along with the properties of mobile web sites stated above, makes ranking extremely difficult.

To solve these problems, we consider a method that uses the titles and snippets of search results which gathered up many users' clicks to clarify the user's intention implied in the search keywords. We define such search results as access concentration sites(ACS) and the titles and snippets as TS of ACS. Users typically click search results after seeing the titles and snippets, therefore it is thought that the ACS contains expressions that users found useful. These are our basic ideas.

Consequently, the problems that should be resolved are as follows;

1. Precise extraction of ACS from many search results,
2. Effective way of applying them to improve search accuracy.

Our solution to problem 1 is analyzing the click logs for the targeted query and plot the search result ranks vs click number. On the resulting curve, ACS must be represented by a strong peak because they attracted many users. The degree of ACS is thought to correspond to how steep the peak is. By estimating the degree, we can select the top k ACS candidates from among many search results. Against problem 2, we consider that sites that include many terms extracted and selected from TS of ACS have a large possibility of being user-desired sites. Therefore we adopt the query expansion method with these terms as our approach.

We summarize related works in the following chapter. In Chapter 3, we describe preliminary experiments (Section 3.1-3), the evaluation method used (Section 3.4) and the results of preliminary experiments(Section 3.5). We explain our new method in Chapter 4, along with the results of the experiments in Chapter 5. Chapter 6 concludes this paper.

2 Related Works

Our proposed method aims to precisely choose only ACS candidates from many search results. That is, we consider some search results as potential feedback

[1] We used query logs of mobile goo(http://mobile.goo.ne.jp) from May 1, 2007 to June 18, 2007.

sources. This point of view is similar to a local feedback technique(pseudo feedback technique) which considers the whole top K search results as feedback information[6].

Feedback sources are often used to extract terms relevant to the input query. Query expansion with the number of T terms is performed to offset the mismatch between the original query and the documents. The term selection technique was first researched by Robertson who introduced the Robertson selection value(RSV)[18][19] which is intended to rank possibly effective expansion terms in decreasing order. Claudio et al. used the Kullback-Leibler distance to rank and weight expansion terms instead of RSV[7]. As there are normally both relevant and non-relevant sites in the top K search results, many researchers have also focused on how to precisely extract only the relevant ones. Attar et al. used clustering techniques to identify eligible local feedback and they found their method to be a practical tool for improving the overall performance of a retrieval system[8]. Mitra et al. utilized term cooccurence information to estimate word correlation, aiming to refine the local feedback[9]. They demonstrated that using the refined feedback sources to query expansion effectively prevents query drift, which is likely if local feedback is used as is.

Sakai et al. consider that the main problem is due to the use of the same parameter values of K and/or T for all queries. Their solution is a technique, called flexible pseudo-relevance feedback, that varies K and/or T across queries to enhance its reliability[10]. Recently Tao et al. presented a robust method for pseudo feedback based on statistical language models, which has no parameter to tune such as K or relative weight of original query[11]. Their method was tested and shown to be significantly more robust than existing language models with regard to feedback sources.

We expect the incorporation of click logs will yield more precise selection of the relevant search results because the clicks on each search result represent conscious user decisions. Our proposing method utilizes relative click numbers to rank the whole search results in a decreasing order of their access concentration degree, not by absolute click numbers.

For an example of a research using absolute click numbers, Cui et al. constructed probabilistic correlations between query terms and terms in the titles and snippets of clicked search results according to the absolute click numbers[12]. Their approach is to select expansion terms according to the correlation measures and then perform query expansion. Tests showed a large improvement in search precision. Another example of extracting useful information from click logs, Shen et al. also paid attention to re-formulation processes for query expansion[13]. According to their paper, the series of past re-formulations are expressed as probability distributions and the current query are expanded by combining it with the distributions. They estimated the search precision and concluded that using clicked site was more effective than re-formulation processes in improving search result quality.Zhuang et al. and Parikh et al. also devised similar methods[14][15]. Their techniques differed from that of Shen et al. in the point that they use query re-formulation to re-rank search results, not to expand queries.

As shown above, many methods using click logs are based on the assumption that the absolute click numbers obtained from click logs are effective. However Joachims et al. investigated the users' decision process and concluded that clicks are informative but somehow biased[16]. They stated that relative preferences derived from clicks are reasonably accurate on average and utilized them as training data to train retrieval functions[17]. Experiments showed that their method performs well in practice and yields successful retrieval functions. Their method is intuitive and demonstrates that the use of relative click numbers is quite effective. However the problem is that only partial relative preferences are derived from click logs over the whole search results, which means the whole search results can not be ranked uniformly. We can not get enough relative preferences usually and if there are intervals between the preferences, it becomes impossible to connect them.

3 Preliminary Experiments

In this chapter, we tested four different methods to grasp the relative precision scores and create baselines as follows;

- BM25 (baseline 1),
- Re-ranking search results by click number order(baseline 2),
- Query expansion by pseudo-feedback(baseline 3),
- Query expansion by click numbers from click logs(baseline 4).

Baseline 1 is a bag-of-words ranking function without any feedback, described in Section 3.1. Section 3.2 describes baseline 2, a method to re-rank search results in decreasing order of click numbers. Section 3.3 describes baseline 3 and 4. Both baselines use the same query expansion technique but acquire the expansion terms differently. The former selects expansion terms from the top k search results and the latter uses the top k click number search results. In Section 3.4, we explain our evaluation method and show the evaluation results of each method in 3.5.

3.1 BM25 (Baseline 1)

As the baseline 1 without any feedback, we adopted BM25, which was widely used as a strong full-text retrieval function. If the query q consists of words t_i, $(i = 1, 2, 3, \cdots, n)$ and we express the document as d. The score of d is given by

$$BM25(q,d) = \sum_{i=1}^{n} w_d(q) \tag{1}$$

$$w_d(q) = \frac{(k_1+1)tf(q,d)}{K + tf(q,d)} \times log(\frac{|N| - df(q) + 0.5}{df(q) + 0.5})$$

$$K = k_1(1 - b + b\frac{dl(d)}{avdl})$$

Here, $BM25(q,d)$ represents a similarity score between q and d. $tf(q,d)$ is a term frequency of q in d, N is a number of total documents in our index, and $df(q)$ is a document frequency which includes q within N. $dl(d)$ is a document length of d and $avdl$ is an average document length over our index. b and K are parameters which are both empirically set as $b = 0.75$ and $k_1 = 1.2$ The documents are ranked in decreasing order of their scores and presented as the search result.

3.2 Re-ranking Search Results by Click Number Order(Baseline 2)

We can order sites according to the click numbers by analyzing the click logs. The search results were re-ranked in decreasing order of the absolute click numbers and we examined how much this improved the precision.

3.3 Query Expansion Method

Query expansion is a method, which we first acquires words(called expansion terms) from possible relevant documents to the input query, second adds them to the query to offset the mismatch between the query and documents. As for query expansion method, the important goals are to identify from where we get the expansion terms and which words we actually use among the many candidates.

In the following two query expansion methods, the obtained expansion terms are ranked by the Robertson selection value (rsv)[18][19] which is widely used as the selection criteria. The rsv is defined as,

$$\begin{aligned} rsv(i) &= r(i) \times rw(i) \\ rw(i) &= log\frac{(r(i)+0.5)(N-n(i)-R+r(i)+0.5)}{(n(i)-r(i)+0.5)(R-r(i)+0.5)} \end{aligned} \tag{2}$$

where $r(i)$ is the number of relevant documents containing term i, and $rw(i)$ is the standard Robertson/Sparck Jones relevance weight[2]. R is the total number of relevant documents for this query and N is the total number of documents. The selected expansion terms are only used for reranking the original search results.

Query expansion by pseudo-feedback(baseline 3). We used the whole top K search results of each query to acquire expansion terms. We devolved the titles and snippets into words t_i, $(i = 1, 2, 3, \cdots, n)$ and adopted expansion terms in decreasing order of the rsv(equation 2).

In calculating rsv, we considered the top K search results as relevant documents to the query. That is to say, we set $R = K$ and r_i was the number of documents over K, which contained term i. We then performed query expansion and retrieved search results based on equation(1).

Query expansion by click numbers from click logs(baseline 4). Click logs consist of query and URLs actually clicked. Therefore, by collating input query

q against the click logs we can clarify the top K click number search results. We used the top K click number search results to acquire expansion terms. The titles and snippets are then devolved into words t_i, $(i = 1, 2, 3, \cdots, n)$ and adopted as expansion terms in decreasing order of the rsv(equation 2).

As well as the pseudo-feedback in calculating rsv, we set $R = K$ and r_i was the number of documents over K, which contained term i. We then performed query expansion and retrieval based on equation(1).

3.4 Evaluation Method

We conducted experiments using index data for each query, about 1.5 months click logs and 45 queries from Japanese mobile search service site, "mobile goo", which is introduced in Chapter 1. The 45 queries in our data were randomly chosen from common queries submitted to mobile goo.

Retrieval is performed by exact matching of the words in the query to those in the titles and bodies. The sum of word scores is the score of the document(see equation (1)). We display the retrieved documents in decreasing order of their score as the search results.

The relevance judgment data was collected by judging every site of search results retrieved by the query as correct or not by hand. The metric used to judge the precision of search results was Mean Average Precision(MAP).

3.5 Evaluation Result

We plot the results of the preliminary experiments in Figure 1. The X axis is the number of expansion terms and the Y axis is MAP.

As for the two query expansion, that is baseline 3 and baseline 4, we tested them using various top K values, such as 3, 5 and 10. Since baseline 3 and

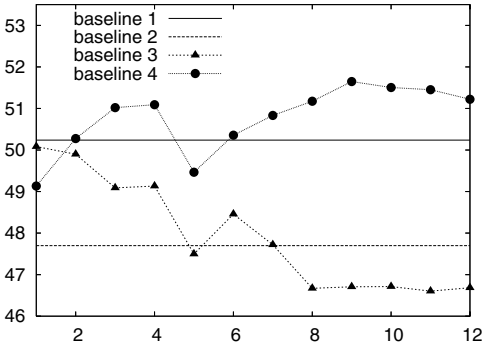


Fig. 1. X axis is the number of expansion terms and Y axis is MAP. Straight line plots our baseline 1, dotted line plots the method that re-ranks search results according to their click numbers(baseline 2), line plots the query expansion by pseudo-feedback(baseline 3) with the top 3 search results and line plots the query expansion by the click numbers of click logs(baseline 4) with the top 3 click number sites. When the expansion terms became insufficient, we reverted to the MAP of the previous number of expansion terms.

baseline 4 recorded their best performance with $k = 3$, only those scores are plotted in Figure 1.

The MAP of our baseline 1 was 50.24% while that of the method that reorders search results according to their click numbers(baseline 2) was 47.70%. This result indicates that the re-ranking search results simply based on their click numbers does not improve the precision. Therefore some techniques applying the click numbers effectively are necessary.

Query expansion by pseudo-feedback(baseline 3) with the top 3 search results achieved its highest MAP 50.08% with 1 expansion term. From this result, we understand that query expansion by pseudo-feedback does not improve the

Query expansion by the click numbers of click logs (baseline 4) with the top 3 click number sites recorded its highest MAP 51.65% with 9 expansion terms. At this time, the query expansion by the click numbers of click logs improved the search precision with 2-4 and 6+ expansion terms. Although its highest MAP was the best among other baselines, the change against BM25 was about 1.4%.

From this overall preliminary experiment, we find that some improvement can be achieved by using click logs(absolute click numbers) as the selection criteria for expansion terms. However there is an obvious problem in only using absolute click numbers, that is, considering merely clicked-well sites as relevant sites, because higher rank sites are much more clicked than lower sites[5]. In a background, the click numbers decrease exponentially along with an increase in search result ranks, so simply using the absolute click number includes a significant bias. It seemed that no further improvement can be achieved until we solve or alleviate this troublesome background problem. That consideration led us to the idea of using relative click numbers of adjoining ranks to extract only relevant search results.

4 Proposed Method

Considering the results of the preliminary experiment, we decided to extract only those sites whose titles and snippets really attracted many users and as a result, were accessed frequently. We defined such sites as access concentration sites(ACS) and the titles and snippets as TS of ACS. It is clear that TS of ACS are effective for improving search accuracy because users typically click a search result after seeing the title and snippet, which implies that TS of ACS are quite suitable for the input query. The precise extraction of ACS is realized by analyzing click logs from a different point of view.

Our proposed method is applied to each query in two stages as detailed below;

1. ACS extraction from search results using click logs,
2. Query expansion with terms extracted from TS of ACS.

4.1 ACS Extraction from Search Results Using Click Logs

In order to precisely extract ACS from many search results, we assumed that if we plot click number versus results rank, ACS must be represented by a strong peak on the curve because they attracted many users.

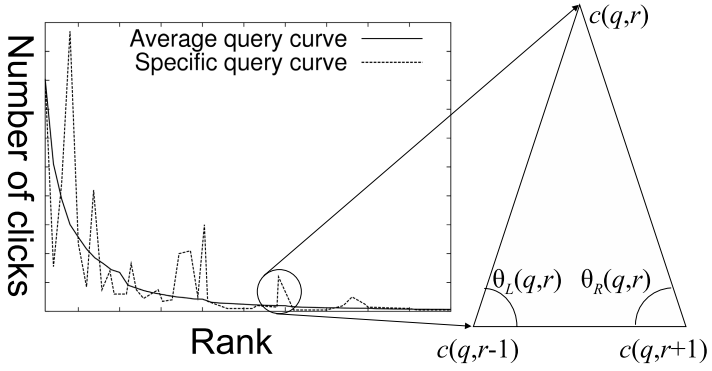


Fig. 2. Click number versus search result rank. Sites that yield strong peaks are taken as ACS. The left and right angles are used for calculating ACD.

We explain this idea in detail by using Figure 2. The left part of this figure was plotted by analyzing our click logs. The X axis is the rank of search results and the Y axis is the click number. The solid curve drawn by analyzing click logs for a certain query is called the specific query curve. The dotted curve yielded by analyzing click logs for all queries and normalized by the highest click number of the specific query curve is called the average query curve. If the specific query curve shows peaks at certain ranks whereas the average query curve shows no corresponding peak, we assume that the sites there attract many more users than the usual sites. Such curves can be measured using the angles of the peak's triangle, which is shown at the right part of the Figure 2. Our method identifies the subtle difference of the left and right angles, and the sites that have large total angles are taken as ACS candidates. Therefore, the sum of the angles is considered to express the degree of ACS so we define it as access concentration degree(ACD).

ACD is capable of distinguishing whether a sites can become an ACS candidates or not. We formulate ACD by utilizing a function $arctan(x)$(see Figure 2). $arctan(x)$ also has the good property that it rapidly saturates with an increase in x. Note that there is a strong tendency that higher rank sites are more often clicked than lower sites, which implies that the inclinations(that is the angles) of higher rank sites are large. It follows that by using $arctan(x)$ to estimate ACD will greatly suppress the effect of the background bias.

For a certain query, we pay attention to each rank and its immediate neighbors, that is r, $r-1$, $r+1$. Each click number is thus expressed by $c(q, r_d)$, $c(q, r_d - 1)$, $c_(q, r_d + 1)$. We first estimate the slopes of the both side curves;

$$\text{slope}_\text{L} = (c(q, r_d) - c(q, r_d - 1))/1$$
$$\text{slope}_\text{R} = (c(q, r_d) - c(q, r_d + 1))/1$$

We then define ACD as;

$$ACD(q, r_d) = \theta_L(q, r_d) + \theta_R(q, r_d)) \tag{3}$$

$$\theta_L(q, r_d) = \begin{cases} \arctan(\alpha \times \text{slope}_\text{L}) & (\text{if } c(q, r_d) - c(q, r_d - 1) > 0) \\ 0 \text{ (otherwise)} \end{cases}$$

$$\theta_R(q, r_d) = \begin{cases} \arctan(\beta \times \text{slope}_\text{R}) & (\text{if } c(q, r_d) - c(q, r_d + 1) > 0) \\ 0 \text{ (otherwise)} \end{cases}$$

Here, α and β determine the degree of emphasis we place on the two angles. In the experiment, we varied α, β values to understand the role of these parameters.

However there are two problems in calculating ACD by using click logs that we should note. First, ranks to which the sites belong change such as 3 to 2 or 9 to 10. This means that the ranks next to each other change, as a result it becomes difficult to identify ACD in a stable manner. Thus, in calculating the angles, we need an appropriate method that is capable of handling the change well. Our solution is to estimate each average rank of the search result during the movement. We use the estimated average rank to reorder the search results, and then calculate ACD.

To precisely calculate the average ranks, we require the occupancy durations at each rank. Therefore, we extract not only the click numbers but also the occupancy durations from click logs. With these factors, if a certain rank belongs to ranks r_d and occupancy duration is expressed as $t_j(r_d)$, the average rank is as follows;

$$cr_d = \sum_j (t_j(r_d) \times r_d) / \sum_j t_j(r_d) \tag{4}$$

We then assign integer ranks such as $1, 2, 3, ...$ in increasing order to the average ranks to normalize them and yield r_d. Click number $c(q, r_d)$ in equation(3) is replaced by the total click number for the site over n days. We explain the parameter n in next paragraph.

Second, the angles highly depend on the click numbers of both sides. This implies if the click numbers being compared are always high, even though the site in the center has quite good titles and snippets and is likely to become an ACS candidate, the calculated ACD is always low. Thus we also need to devise a method that is capable of offsetting the unfairness of the comparison sites. Our solution is to make use of the rank changes. We calculate the average ranks cr_d and the ACD_i every n days by using click logs, the intention is to rate a site against many sites, not just one site. In other words, owing to the brisk changes in ranks, it becomes possible to rate a site against many sites.

Finally the total angles(that is, the total of ACD_i) are used to calculate the final ACD. In our experiment, we used 1.5 months of click logs and set n to 15 so that the summation was repeated $45days/15days = 3times$.

$$ACD(q, r_d) = \sum_i ACD_i(q, r_d) \tag{5}$$

We ordered the sites in decreasing order of their ACD. Then top K sites were then adopted as ACS to improve the search result quality.

4.2 Query Expansion with Terms Extracted from TS of ACS

Here, we restate our hypotheses about TS and ACS again. We believe that when the user decides to click a search result, the user must have judged that the document would be useful after seeing the TS. This means that the TS of ACS are quite suitable for expanding input queries and they are expressions that users have found useful.

Consequently, we treat the TS of ACS as highly refined local feedback and perform query expansion with terms extracted from them. We used the top K ACD(equation(5)) search results of each query to acquire expansion terms.

Query expansion is performed in the same with Section 3.3. In calculating the rsv, we considered the top K ACD search results(that is top K ACS) as relevant documents to the query. Therefore, we set $R = K$ and r_i was the number of documents over K, which contained term i. We then performed query expansion and retrieved search results based on equation(1).

5 Experiments and Result

We show the overall result in Figure 3. The evaluation of our method was conducted using the same manner described in Section 3.4. We used 1.5 months of click logs and set n to 15, which means we calculated the ACD every 15 days. We varied (α, β) values in equation(3), using $(1.0, 1.0)$, $(1.0, 0.5)$, $(1.0, 0.2)$, $(0.5, 0.5)$, $(0.5, 0.25)$, $(0.5, 0.1)$, and adopted top $K = 3, 5, 10$ ACD sites as ACS.

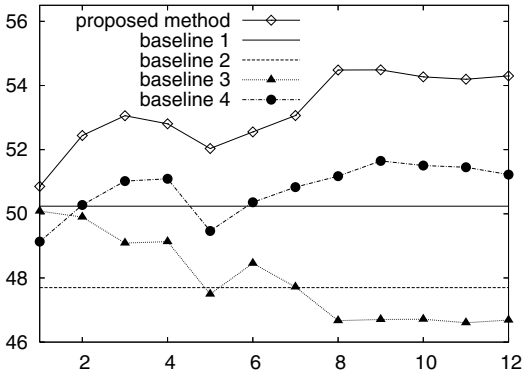


Fig. 3. X axis is the number of expansion terms and Y axis is MAP. Straight line plots our baseline 1, dotted line plots the method that re-ranks search results according to their click numbers(baseline 2), ▲ line plots the query expansion by pseudo-feedback(baseline 3) with the top 3 search results and ● line plots the query expansion by the click numbers of click logs(baseline 4) with the top 3 click number sites. ◇ line plots the query expansion with the top 3 access concentration sites. When the expansion terms became insufficient, we reverted to the MAP of the previous number of expansion terms.

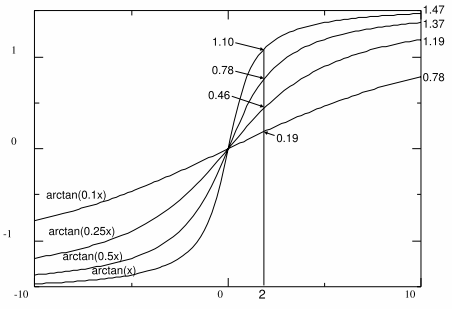


Fig. 4. The curves of $arctan(x)$, $arctan(0.5x)$, $arctan(0.25x)$, and $arctan(0.1x)$

Our proposed method recorded its best performance with $K = 3$ and $\alpha = 0.5, \beta = 0.5$, which is plotted in Figure 3. The highest MAP was 54.49% with 9 expansion terms. All MAP values of the expansion term number were much higher than that of all baselines recorded in the preliminary experiments.

We further investigated the difference between the highest MAP of our proposed method and the highest MAP of the preliminary experiments recorded by query expansion based on click numbers from click logs(baseline 4). Statistically-significant differences in performance were determined according to the two-sided wilcoxon signed rank test($0.01 < p < 0.05$). This result implies that we have succeeded in extracting more relevant feedbacks from many search results than the baselines did.

The reason why $\alpha = 0.5, \beta = 0.5$ yielded the best performance is considered as below. We plot the curves of $arctan(x)$, $arctan(0.5x)$, $arctan(0.25x)$, and $arctan(0.1x)$ in Figure 4. $arctan(x)$ rapidly increases until about 2 and then dramatically saturates. This means that if we use just $arctan(x)$, we cannot attach a large difference between very few click numbers and more click numbers. For example, the ratio of $arctan(x), x = 2$ to $arctan(x), x = 10$ is about 0.75, which implies that the importance is not so different. However, the change of click numbers such as 2 may be accidental, and it is better to drop the importance a little more. On the other hand, $arctan(0.25x)$ and $arctan(0.1x)$ suppress the relative importance too much. The each corresponding ration is about 0.39 and 0.25, which leads to inaccurate ACD values. $arctan(0.5x)$ provides a better assessment of relative importance. Indeed, the ratio of $arctan(0.5x), x = 2$ to $arctan(0.5x), x = 10$ is about 0.57, which implies that the importance is almost halved. Consequently we can understand that the function $arctan(0.5x)$ has superior in discriminating between accidental click numbers and normal click numbers.

We present three examples of queries and the expansion terms in Table 2.

“高橋大輔”(Daisuke Takahashi) is a famous Japanese figure skater. He won a silver medal at the 2007 world championship and played an active part in the Turin Olympics. Therefore, the extracted expansion terms were“日本のメダル”(medal in Japan), “外国メディア”(foreign media), “メダル”(medal) and “トリノ”(Turin). On the other hand, there is another “高橋大輔”(Daisuke Takahashi) in Japan; he is a soccer player and his alma mater is “福岡大”(Fukuoka Univ.).

Table 2. Examples of queries and the expansion terms

query	expansion terms
高橋大輔	日本のメダル,外国メディア,メダル,トリノ,福岡大
Daisuke Takahashi	medal in Japan,foreign media,medal,Turin,Fukuoka Univ.
つわり	不快 症状 色々 栄養不足
morning sickness	indisposition,symptom,various,poor nutrition
トイザらス	日本トイザらス,トイザらスギフトカード,日本,こどもの日
Toys"R"Us	Toys"R"Us-Japan,Toys"R"Us gift card,Japan,Children's Day

He married a graduate of the same university. So, the expansion term 福岡大"(Fukuoka Univ.) is also related to the query "高橋大輔"(Daisuke Takahashi). The MAP of baseline(BM25) scored 25.75%, while the query expansion with these terms achieved 28 50%.

The extracted expansion terms for "つわり"(morning sickness) were"不快"(indisposition),"症状"(symptom),"色々" (various) and "栄養不足" (poor nutrition). "色々"(various) is an adjective and thought to modify a terms such as "症状"(symptom). From these expansion terms, users appear to search for the symptoms and causes of morning sickness. Its MAP of baseline(BM25) was 43.34%, while the query expansion with these terms was 49.32%.

"トイザらス"(Toys"R"Us) is a major global toy company."日本トイザらス"(Toys"R"Us-Japan), "トイザらスギフトカード"(Toys"R"Us gift card),"日本"(Japan) and "こどもの日"(Children's Day) were extracted for the expansion terms. "日本"(Japan) was seen often. We considered that the reason was that Japanese users searched sites on "日本トイザらス"(Toys"R"Us-Japan), not on U.S. Toys"R"Us and others. "こどもの日"(Children's Day) is a national holiday in Japan and as the word suggests, the holiday is to wish child's happiness. Therefore, many parents in Japan buy toys for their children in Toys"R"US. The MAP of baseline(BM25) was 33.23%, while the query expansion with these terms achieved 47.94%.

The overall results confirm that our technique for extracting ACS from search results is successful and that query expansion based on the TS of ACS can significantly improve the search accuracy. With regard to implementation concerns, real search engines demand fewer expansion terms that still offer a large improvement to lower calculation costs.

We verified that our method offered much higher precision with a few expansion terms than all baselines, thus our proposed method is also promising in terms of practicality.

6 Conclusion

In this paper, we introduced a new metric, access concentration degree(ACD), which is capable of evaluating the degree of access concentration for each search result. Our aim is to extract only access concentrated sites(ACS) from many search results, with the understanding that the title and snippet(TS of ACS) are quite suitable for expanding the input query.

We tested the proposed method using the click logs and data from an actual mobile web search site, "mobile goo", and demonstrated that it offered much higher precision than all baselines considered in this paper. This results support our hypotheses of ACS and TS of ACS, and also imply the strong potential of our proposal.

Our basic innovations are summarized below. We define ACS as the sites that have useful titles and snippets because they attract many users to click on the sites. These sites should show up as peaks in the curve of click number versus result ranks. Next, we use the left and right side inclinations of the peaks (suitably weighted) to generate ACD. Then top K ACD sites are taken as ACS.

However there are two problems in calculating ACD: rank movement and unfairness in site comparison. To resolve these problems, we calculate the average rank during the movement and the ACD in 15 day periods. The grand total of ACD becomes the final ACD of the site. After extracting ACS and related TS, we apply them to improve precision. Based on our hypotheses, TS of ACS are quite suitable for enhancing input query through query expansion. Consequently, sites that frequently include the same words as TS of ACS must be query candidates that will find user-desired sites.

We intend to ascertain in more detail the problems and characteristics of mobile web search and develop solutions using various data sets. Because our ACS method is not limited to a mobile web search, we are also looking at the possibility of applying this method to a web search or other tasks, such as keyword recommendation.

References

1. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24(5), 513–523 (1988)
2. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M., Gatford, M.: Okapi at TREC-3. In: Proc. of TREC-3, pp. 109–126 (1995)
3. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project (1998)
4. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: Proc. of WWW7, pp.107–117 (1998)
5. Jansen, B.J., Spink, A.: An Analysis of Web Documents Retrieved and Viewed. In: International Conference on Internet Computing, pp. 65–69 (2003)
6. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press Series. Addison-Wesley Pub.(Sd), Reading (1999)
7. Carpineto, C., De Mori, R., Ropano, G., Bigi, B.: An information-Theoretic Approach to Automatic Query Expansion. ACM Transactions on Information Systems 19(1), 1–27 (2001)
8. Attar, R., Fraenkel, A.S.: Local Feedback in Full-Text Retrieval Systems. Journal of ACM 24(3), 397–417 (1977)
9. Mitra, M., Singhal, A., Buckley, C.: Improving Automatic Query Expansion. In: Proc. of SIGIR 1998, pp. 206–214 (1998)
10. Sakai, T., Robertson, S.E.: Flexible Pseudo-Relevance Feedback Using Optimization Tables. In: Proc. of SIGIR 2001, pp. 396–397 (2001)

11. Tao, T., Zhai, C.: Regularized Estimation of Mixture Models for Robust Pseudo-Relevance Feedback. In: Proc. of SIGIR 2006, pp. 162–169 (2006)
12. Cui, H., Wen, J., Nie, J., Ma, W.: Probabilistic Query Expansion Using Query Logs. In: Proc. of WWW 2002, pp. 325–332 (2002)
13. Shen, X., Tan, B., Zhai, C.: Context-Sensitive Information Retrieval Using Implicit Feedback. In: Proc. of SIGIR 2005, pp. 43–50 (2005)
14. Zhuang, Z., Cucerzan, S.: Re-Ranking Search Results Using Query Logs. In: Proc. of CIKM 2006, pp. 860–861 (2006)
15. Parikh, J., Kapur, S.: Unity: Relevance Feedback using User Query Logs. In: Proc. of SIGIR 2006, pp. 689–690 (2006)
16. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Gay, G.: Accurately Interpreting Clickthrough Data as Implicit Feedback. In: Proc. of SIGIR 2005, pp. 154–161 (2005)
17. Joachims, T.: Optimizing Search Engines using Clickthrough Data. In: Proc. of SIGKDD 2002. pp. 133–142 (2002)
18. Robertson, S.E.: On Term Selection for Query Expansion. Journal of Documentation 46(4), 359–364 (1990)
19. Robertson, S.E., Jones, K.S.: Relevance Weighting of Search Terms. Journal of the American Society for Information Science 27(3), 129–146 (1976)

Can Social Tagging Improve Web Image Search?

Makoto Kato, Hiroaki Ohshima, Satoshi Oyama, and Katsumi Tanaka

Department of Social Informatics, Graduate School of Informatics, Kyoto University, Yoshida-honmachi, Sakyo, Kyoto 606-8501, Japan

Abstract. Conventional Web image search engines can return reasonably accurate results for queries containing concrete terms, but the results are less accurate for queries containing only abstract terms, such as "spring" or "peace." To improve the recall ratio without drastically degrading the precision ratio, we developed a method that replaces an abstract query term given by a user with a set of concrete terms and that uses these terms in queries input into Web image search engines. Concrete terms are found for a given abstract term by making use of social tagging information extracted from a social photo sharing system, such as Flickr. This information is rich in user impressions about the objects in the images. The extraction and replacement are done by (1) collecting social tags that include the abstract term, (2) clustering the tags in accordance with the term co-occurrence of images, (3) selecting concrete terms from the clusters by using WordNet, and (4) identifying sets of concrete terms that are associated with the target abstract term by using a technique for association rule mining. Experimental results show that our method improves the recall ratio of Web image searches.

1 Introduction

The continuing growth of the Internet and advances in Web search engines have made it easy to search for documents, images, videos, and other types of media. Most of the search engines for documents, images, and videos use keyword-matching between the terms in the queries and the terms in documents, the terms around images, and the terms in tags attached to videos. However, this is not always an effective approach. For example, a user searching for springlike images would likely input "spring" or "springlike" as a search term. The results for such a search would likely be poor because images are generally not indexed using abstract terms like these and because the abstract terms in text surrounding images do not always represent the contents of the images. In contrast, the results of an image search using concrete terms usually have more precision, as demonstrated in section 2.

Abstract terms include terms representing concepts such as "spring," "peace," and "kyoto" and sentimental terms such as "happy," "sad," and "gloom." Although such terms are not often used in document searches, they are frequently used in multimedia searches. Multimedia search engines should be able to comprehend the search terms and return relevant images.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 235–249, 2008.

To improve the results of Web image searches made using abstract terms, we have developed a method of extracting sets of concrete terms from social tagging information and using them in queries in place of the abstract terms. The social tagging information used for this transformation is extracted from a system like Flickr that uses "social tagging," which is a form of classification in which content posters and viewers collaborate in tagging the contents. This method is based on the assumption that the tags attached to an image contain information that is more relevant to the image than the text surrounding it on the Web page and that the information is clearly related to a certain type of content. This is helpful in discovering text-based knowledge about images. By making use of this accumulated knowledge, we can improve Web image search.

In section 2, we evaluate current Web image search. In section 3, we describe our approach to improving it. In section 4, we explain how we extract a set of feature terms from Flickr[1], and, in section 5, we discuss our assessment of the effectiveness of our approach. In section 6, we discuss related work, and, in section 7, we summarize the key points, discuss the need for improvement, and mention other possible applications of this approach.

2 Evaluation of Current Web Image Search

Before discussing the problem with current Web image search methods, we classify the types of image searches on the basis of the relationship between the query and the desired images. We define the input term as q, the set of images the user considers to be relevant as $I(= \{i_1, i_2, \ldots, i_k, \ldots\})$, and the set of objects in i_k as O_{i_k}.

We categorize image searches into three types.

2.1 Type 1: Search Item Often Completely Appears in Photos

When a concrete term is submitted as a query, and it represents an item that often completely appears in a photo, such as an automobile, a bird, or a butterfly, the relation between query q and a certain object in O_{i_k} in relevant image i_k is "instance-of" or "itself." This type of image search is often done when users want to know about q visually. It generally results in relatively high precision and recall. When a query term is the name of a class, for example, q is "apple" and O_{i_k} has "apple" as a object, the relation is "instance-of." When a query term is the name of an instance, for example, q is "Bill Gates" and O_{i_k} has "Bill Gates" as a object, the relation is "itself." Example terms in this type of search and their precisions are listed in Table 1. To determine the precisions, we used the Google Image Search Engine[2] to search for images.

2.2 Type 2: Search Item Rarely Completely Appears in a Photo

When a concrete term is submitted as a query, and it represents an item that rarely completely appears in a photo, like a lake, a forest, or Paris, the relation

[1] http://www.flickr.com/
[2] http://images.google.com/

Table 1. Precisions for each image search type

Type 1			Type 2			Type 3		
Query	Top 20	Top 100	Query	Top 20	Top 100	Query	Top 20	Top 100
moon	1.00	0.82	sea	0.95	0.45	summer	0.40	0.25
bird	0.95	0.68	sky	0.90	0.47	winter	0.90	0.54
rose	1.00	0.55	forest	0.90	0.52	justice	0.45	0.25
butterfly	0.80	0.70	lake	0.95	0.67	love	0.25	0.17
bear	0.95	0.75	tokyo	0.65	0.47	sad	0.95	0.37
cat	1.00	0.86	kyoto	0.75	0.47	powerful	0.35	0.10
dog	1.00	0.76	paris	0.70	0.41	america	0.25	0.10
pigeon	1.00	0.77	kyoto-university	0.45	0.22	japan	0.45	0.23
ipod	1.00	0.79						
headphone	1.00	0.77						
Avg.	0.97	0.74	Avg.	0.78	0.46	Avg.	0.50	0.25

between query q and a certain object in O_{i_k} in relevant image i_k is "part-of." The precision of this type is a little lower than those of "instance-of" and "itself," as shown in Table 1. In most cases, when an item does not completely appear in the photo, it is because it is too large, and users consider a partial image of the item to be sufficient. For example, when q is "Japan" and O_{i_k} has "Mt. Fuji" or "kinkakuji temple" as an object, the relation is "part-of." This type of image search task is performed when the query term represents an object that is too large for one image, for example, names of places, the sea, and the sky. The "part-of" relation is similar to the "instance-of" and "itself" relations in that the set of objects in a relevant image includes a "part-of" an "instance-of" a query. At the same time, it is also similar to a relation described below, "associated-with," because in the "part-of" relation, "part-of" an object in O_{i_k} is "associated-with" the whole object.

2.3 Type 3: Search Item Cannot Directly Appear in a Photo

When an abstract term is submitted as a query, it represents an item that cannot directly appear in a photo, and the relation between query q and a certain object in O_{i_k} in relevant image i_k is "associated-with." As shown in Table 1, this type achieves lower precision than those in which the search item can appear in a photo. For example, if q is "spring," and O_{i_k} has "tulip" or "crocus" as a object, the relation between "tulip" or "crocus" and "spring" is "associated-with." The members of the set of objects in relevant image i_k are not instances or parts of a query; they are simply associated with the query.

This typing scheme is complicated by the fact that some query terms can be either concrete or abstract. Consider "Japan" for example. When it is considered to be the name of a place, it is a concrete query term and at least part of the search item such as "Kyoto" or "Mt. Fuji", can appear in a photo. When it is regarded as a concept, a culture, a characteristic, etc., it is an abstract query term and cannot appear in a photo. The search item is not "Japan" itself but

things associated with it, such as “geisha” or “sushi”. Whether a term is concrete or abstract depends on the user’s intention. Therefore, with only the term, it can be difficult to determine whether it is a concrete or abstract search term without considering the user’s intention.

2.4 Preliminary Experiment for Each Type of Image Search

In the preceding section, we compared the precisions for each type of image search and found that the precisions for “instance-of (itself)” searches are much higher than that for “associated-with” searches. In this section, we discuss these differences in more detail using the results of a preliminary experiment.

The query terms used in the experiment were terms randomly selected from headlines on Yahoo! News[3] that appeared from January to February 2008. They were categorized into “instance-of,” “itself,” “part-of,” or “associated-with” type queries and input into Google Image Search. We evaluated the top 100 images retrieved for each type of query. For example, if “kinkakuji temple” was selected from the headline “Kinkakuji temple covered with snow—Snow in Kyoto,” it was regarded as an “itself” query and, as relevant images for “kinkakuji temple,” we selected images in which “kinkakuji temple” appears. In this paper, we exclude the “part-of” type query and focus on the “instance-of (itself)” and “associated-with” type queries. We used 68 queries for “instance-of,” 32 for “itself,” and 100 for “associated-with.” Figure 1 shows the (11-pt average) precision curves. The recall ratio was calculated assuming that all relevant images appeared in the top 100 retrievals.

At each recall level, the precisions for “instance-of” and “itself” were far higher than those for “associated-with.” When terms used as queries for “instance-of”

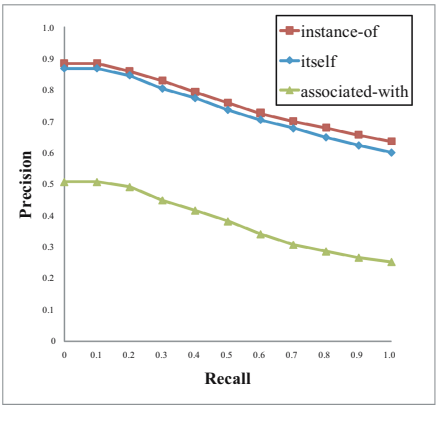


Fig. 1. Recall-Precision curve for each image search type

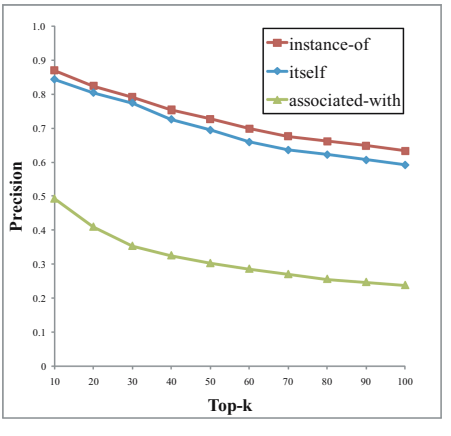


Fig. 2. Top-k precision curve for each image search type

[3] http://headlines.yahoo.co.jp/

or "itself" appear around Web images, they often indicate objects in the images. In contrast, terms used for "associated-with" often refer only to the image context and are sometimes not descriptive of the images because they were subjectively written by the Web page creator. Therefore, even if a query matches the image context, it may not match the image contents. This is why "instance-of" and "itself" queries have better precision than "associated-with" queries.

Figure 2 shows the precisions for every k, the number of images returned. For $k = 10$ to 30, the slope for "associated-with" queries is steeper than those for "instance-of" and "itself" queries. That is, the low precisions for "associated-with" queries at $k = 10$ are much lower at $k = 30$.

These results show that the average precision of "associated-with" queries is low. However, it is not always necessarily low. While some queries may return only a few relevant images, others may return many relevant images. We thus grouped "associated-with" type queries into four categories ("action," "attribute," "condition," and "concept") to investigate the reason for the low precision.

For an "action" type query, relevant images should have an object performing the action. For example, if the action is "running," relevant images should show runners. If it is "baseball," relevant images should show people playing baseball or objects associated with baseball (gloves, bats, etc.). For an "attribute" type query, relevant images should have an object with that attribute. For example, if the attribute is "transparent," the relevant images should show something transparent such as a glass. For a "condition" type query, relevant images should have an object in that condition. We regard "condition" as a temporal feature, while "attribute" is a characteristic that is generally well known, and these are not always visible. Therefore, although an image may contain an object in the condition of interest, the image may not be relevant because the condition may not be recognized by the viewer. For a "concept" type query, relevant images may not have clearly recognizable objects although there should be an object representing or associated with the concept.

As shown in Figure 3, the precisions for the four categories decreased in the order of condition > concept > action > attribute.

The differences in precision between the four categories is attributed to differences in awareness between people who insert images into Web documents and those who view them. For the "action" and "attribute" categories, there is a stronger likelihood that the words placed on the page by the creator to describe an image will be used by someone looking for such an image in his or her query, and they would likely consider it relevant. For example, if the page creator regards an image as a "baseball" image and thus includes the term "baseball" in text surrounding the image, people who search with "baseball" in their query and find that image would likely consider it relevant. For the "condition" and "concept" categories, there is less likelihood of this shared awareness of suitable terms. This is because the words used by the page creator to describe such images often simply indicate the context of the image, which is unknown to viewers.

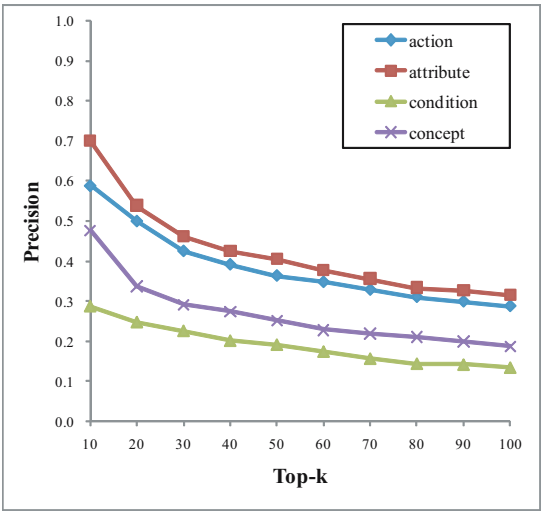


Fig. 3. Top-k precision curve for each "associated-with" category

3 Our Approach

Our goal is to improve Web image searches when abstract terms are used. As shown by the results of our preliminary experiment, Web image search engines work much better with concrete term queries than with abstract term queries. Our approach is to transform an abstract term into sets of concrete terms

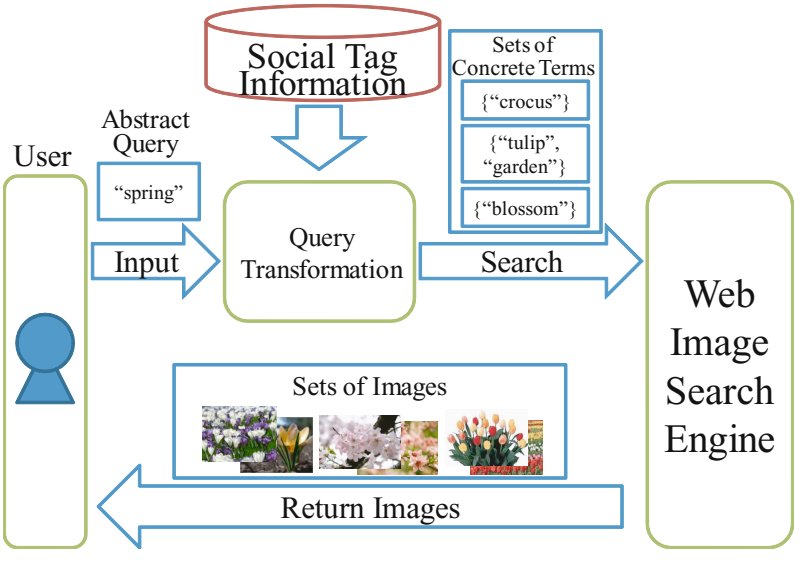


Fig. 4. Overview of our approach

representing the abstract term and to search for images using these concrete terms. The aim is to perform image searches using abstract terms that are close in performance to those using concrete terms. This transformation from an abstract term to sets of concrete terms is done using social tagging information obtained from a photo sharing system such as Flickr. Figure 4 shows an overview of our approach.

3.1 Definition of a Set of Feature Terms

A set of terms A for abstract term x should to be able to return results containing images relevant to x. That is, it must satisfy two conditions:

1. A has the features of x.
2. A is associated with x.

We define such a set as a "set of feature terms." As an example of the first condition, if x is "spring," A must have a term representing a feature of "spring" such as "warm" or "butterfly." As an example of the second condition, if x is "spring," A must have a term associated with "spring" such as "cherry blossom" or "crocus." These two components are not independent: almost all of the features are associated with an abstract concept at each level of association.

If a set of terms does not satisfy both conditions, it is not a set of feature terms. For example, a set containing "summer," "autumn," and "winter" satisfies the second condition for "spring," but not the first, so it is not a set of feature terms. In contrast, "strong wind" satisfies the first, but not the second. Although strong wind is a feature of spring, it is not associated only with "spring" because a strong wind can also blow in other seasons, so the level of the association is relatively low. Hence, this set of terms is not a set of feature terms.

4 Extracting a Set of Feature Terms

In this paper, we extract a set of feature terms for an abstract term from social tagging information found in Flickr. We use Flickr for two reasons. First, the tags in Flickr indicate the objects in the tagged photo more frequently than the text around Web images. Second, viewers as well as the poster can tag photos. If a tag representing an abstract concept is added to a photo by a viewer, it is reasonable to assume that the viewer associates objects in the photo with the abstract concept. This is useful for finding sets of concrete terms corresponding to an abstract term.

Each photo in Flickr can have several tags, so there is generally a one-to-many relation. To extract a set of feature terms for abstract term x, we first extract several sets of concrete terms that represent the features of x and from them select sets of terms that are associated with x to some extent.

4.1 Extracting Feature Clusters

The first step in this process is to obtain a set of feature clusters representing features of the given x. A feature cluster consists of feature vectors for photos.

A feature is represented not as a term but as a set of feature vectors because a term is not enough to identify a feature. For example, "sea" can represent different features in some cases. Specifically, the sea in warm weather countries may represent "vacation," while a raging sea with dark clouds may represent "the threat of nature." Since a term cannot represent a specific feature, we use feature vectors, which have several terms and values for the terms.

Flickr hosts more than 2 billion photos, and almost every photo has multiple tags. We defined variables for the photos and tags:

$P = \{p_1, p_2, \ldots, p_i, \ldots\}$: the set of all photos
$T = \{t_1, t_2, \ldots, t_i, \ldots\}$: the set of all tags
T_{p_i}: the set of tags for photo $p_i \in P$
N: the total number of photos ($\simeq$2,000,000,000)

A feature vector for a photo is weighted using the term frequency-inverted document frequency (tf/idf) method in which we equate a tag with a term and a photo with a document. The dimensions of the feature vector correspond to T.

Our aim is to create a set of concrete terms, but T contains both concrete and abstract terms. Given only a term, we cannot be certain whether it is a concrete one or an abstract one because of possible synonyms and various possible meanings. Therefore, we define the probability that tag t_j is used for the concrete meaning. This probability, conc_j, is calculated using WordNet [1], "a lexical database for the English language." Given a term, WordNet shows all of the meanings for it, its inherited hypernyms, and whether each meaning is a descendant of a physical entity or an abstract entity. Assuming that the meanings are used at the same rate,

$$\mathrm{conc}_j = \frac{\text{The Number of } t_j \text{ Meanings for a Physical Entity}}{\text{The Number of } t_j \text{ Meanings}}. \tag{1}$$

This probability is used in feature vectors to give less weight to the abstract terms in T. If it equals zero, term t_j is considered to be an abstract term and is excluded from T. In this way, separating abstract terms and concrete terms is done automatically.

For each photo $p_i (i = 1, 2, 3 \ldots)$ with x as a tag, we compute a feature vector:

$$V_{p_i} = (\mathrm{tf}_{p_i,t_1} \cdot \mathrm{ipf}_{t_1} \cdot \mathrm{conc}_1, \mathrm{tf}_{p_i,t_2} \cdot \mathrm{ipf}_{t_2} \cdot \mathrm{conc}_2, \ldots, \mathrm{tf}_{p_i,t_n} \cdot \mathrm{ipf}_{t_n} \cdot \mathrm{conc}_n), \tag{2}$$

where

$$\mathrm{tf}_{p_i,t_j} = \begin{cases} 1 \ (p_i \text{ has a tag } t_j) \\ 0 \ (otherwise) \end{cases},$$

pf_{t_j} is the number of photos that have tag t_j, and

$$\mathrm{ipf}_{t_j} = \log \frac{N}{\mathrm{pf}_{t_j}}.$$

To classify the photos, we use a complete linkage method, which is a hierarchical clustering method. The similarity between two photos is represented by the cosine similarity between their vectors. Centroid vectors for feature clusters of "spring" are shown as an example in Table 2.

Table 2. Centroid vectors for "spring"

Tag	Value
peony	12.11
ant	11.26
insects	9.76
bugs	8.15
garden	4.93

Tag	Value
blueberries	12.46
kiwi	11.36
salad	11.01
honey	10.72
raspberries	8.60

Tag	Value
petals	10.48
daisy	10.17
centerre	7.77
sky	6.98
stem	5.96

Tag	Value
branches	6.97
plants	6.36
blossoms	5.33
trees	5.02
bravo	5.02

Tag	Value
flower	4.65
flowers	4.61
change	3.37
roses	3.20
world	3.09

Tag	Value
tulip	10.11
bravo	5.02
garden	4.93
drops	4.65
nature	2.62

4.2 Evaluating Feature Clusters by Association Rule Mining

The feature clusters obtained by hierarchical clustering represent the features of x, but they may not always be associated with x. Therefore, we evaluate the terms in each feature cluster in terms of their association with x and select those terms with a score exceeding a certain threshold. We do this using two values, "confidence" and "support," which are used in association rule mining [2]:

$$\mathrm{conf}(A, x) = \frac{\mathrm{pf}(A \cup \{x\})}{\mathrm{pf}(A)} \tag{3}$$

$$\mathrm{sup}(A, x) = \mathrm{pf}(A \cup \{x\}), \tag{4}$$

where $\mathrm{pf}(A)$ is the number of photos that contain all the tags in A. We extract the set of tags from the clusters and evaluate the relationship between each tag and x. Using centroid vector m_i for cluster c_i, we select those tags with the k highest values and make a power set, T_i, containing them. The power set thus contains candidate feature terms. Centroid vector m_i for cluster c_i is defined as

$$m_i = \frac{1}{n_{c_i}} \sum_{k=1}^{n_{c_i}} v_k \tag{5}$$

where v_k is a k-th feature vector in c_i,and n_{c_i} is the number of them in c_i.

For each A of T_i, we evaluate the relation

$$A \Rightarrow x. \tag{6}$$

There is a potential problem with this. If the value of $\mathrm{sup}(A, x)$ is relatively low, i.e., the number of photos that contain objects corresponding to all the terms in $A \cup \{x\}$ is low, it is possible that the values for $\mathrm{sup}(A, x)$ and $\mathrm{conf}(A, x)$ are unduly high because a user has uploaded many photos with the same set of tags. To avoid this problem, we weight the users equitably by using uf(user frequency) instead of pf(photo frequency), where $\mathrm{uf}(A)$ is the number of users who have uploaded photos that contain all the tags in A.

4.3 Extracting Sets of Feature Terms from Feature Clusters

After we evaluate each member A of T_i that is a candidate set of feature terms, as described above, we select the set with the highest "confidence" value that also has a "support" value higher than min_sup. The reason we select the set with the most terms associated with x is to remove the redundancies and optimize the size of this set of feature terms. Figure 5 diagrams the extraction from tag clusters using association rule mining.

Ten sets of feature terms for "spring" are listed in Table 3. The number of photos returned was 400, and min_sup was 100. The value of uf was estimated due to the tremendous number of photos in Flickr. It was obtained by using the fraction of unique posters for the top 500 photos instead of the fraction of unique ones for all photos.

5 Evaluation

To assess the effectiveness of our method, we compared the results between image searches using an abstract term and ones using a set of feature terms for the abstract term. We used 20 abstract terms: africa, cute, fly, future, gloom, happy, heaven, hell, japan, jump, love, music, peace, powerful, scary, sing, spring, summer, sweet, and warm.

First, for each abstract term x, we retrieved 400 photos from Flickr, clustered them, and extracted from them 10 sets of feature terms with the highest confidence values. The number of terms extracted from the feature clusters, k, was 5, and min_sup was set to 50. The sets of feature terms extracted for "peace" and

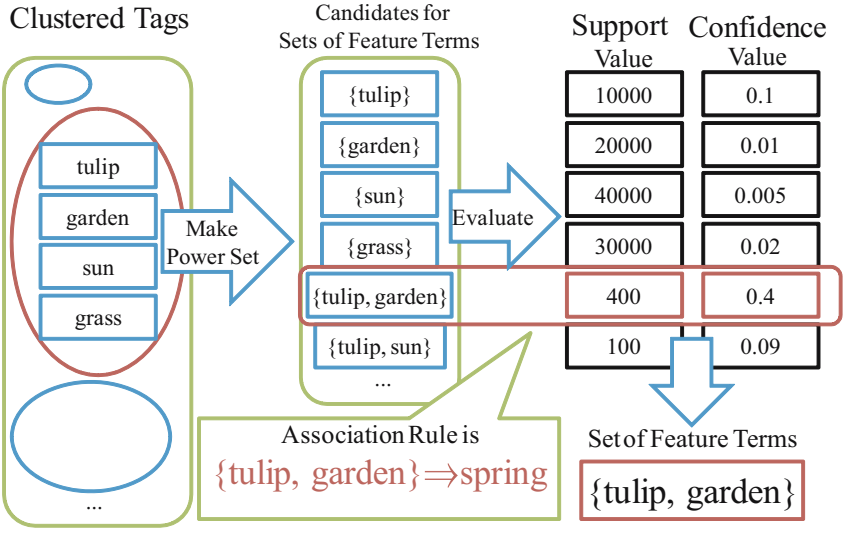


Fig. 5. Extraction from tag clusters using association rule mining

Table 3. Sets of feature terms for "spring"

Set	Support	Confidence
crocus	7013	0.46
garden tulip	3269	0.35
flowers daffodils	5194	0.35
blossoms	15730	0.34
forsythia	1460	0.34
tree spain	5313	0.26
nature flower japan	1446	0.22
bluebells	5403	0.19
lilacs	3032	0.09
dandelions	3410	0.08

"sweet" are shown in Tables 4 and 5. The feature terms in each set were joined with "AND" and used as an image search query in Yahoo! Developer Network[4].

Then, for each abstract term, we searched for 500 images, and for each set of feature terms for the term, we searched for 50 images, which were then merged into a set of 500 images. We manually checked each of these 1000 images for relevance to the query.

Table 4. Sets of feature terms for "peace"

Sets	Sup	Conf
lebanon israel palestine	267	0.13
calm	16890	0.07
iraq	48085	0.07
lebanon	23733	0.05
activists	821	0.05
buddha	70878	0.04
dove	16871	0.04
bush	73686	0.03
christian father	88	0.03
hope	31723	0.02

Table 5. Sets of feature terms for "sweet"

Sets	Sup	Conf
tender	4471	0.11
food dessert	12314	0.11
muffin dessert	73	0.10
blue candy	1664	0.10
dessert	29160	0.09
food chocolate	11867	0.09
baby eyes girl	1687	0.09
tart	3467	0.09
dogs baby puppies	112	0.08
face cat ears	106	0.08

Recall-precision curves with 11-pt average precision are shown in Figure 6 for the abstract terms and the sets of feature terms. We estimated the recall by assuming that all of the relevant images were included in the 1000 images. At recall = 0.0, there was no difference in the precisions. From 0.1, the precision for the sets of feature terms was higher at every recall level. Particularly from 0.1 to 0.5, the sets maintained higher precision.

The recall-precision curves for six of the abstract terms individually and the corresponding sets of feature terms are shown in Figure 7. In almost every case, both precision and recall were higher when the set was used.

[4] http://developer.yahoo.co.jp

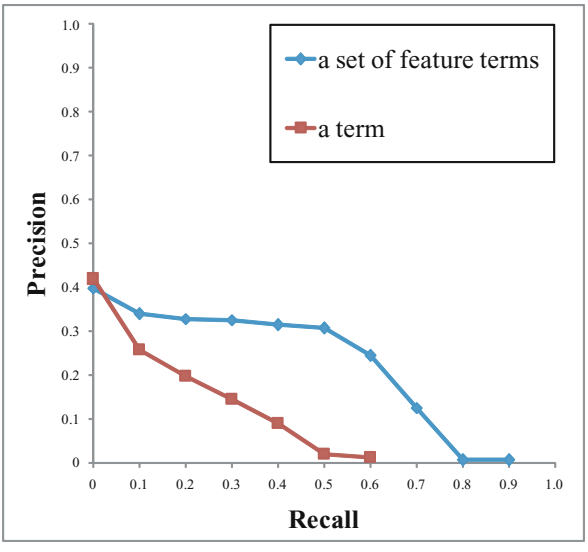


Fig. 6. 11-pt average precision for abstract terms and sets of feature terms

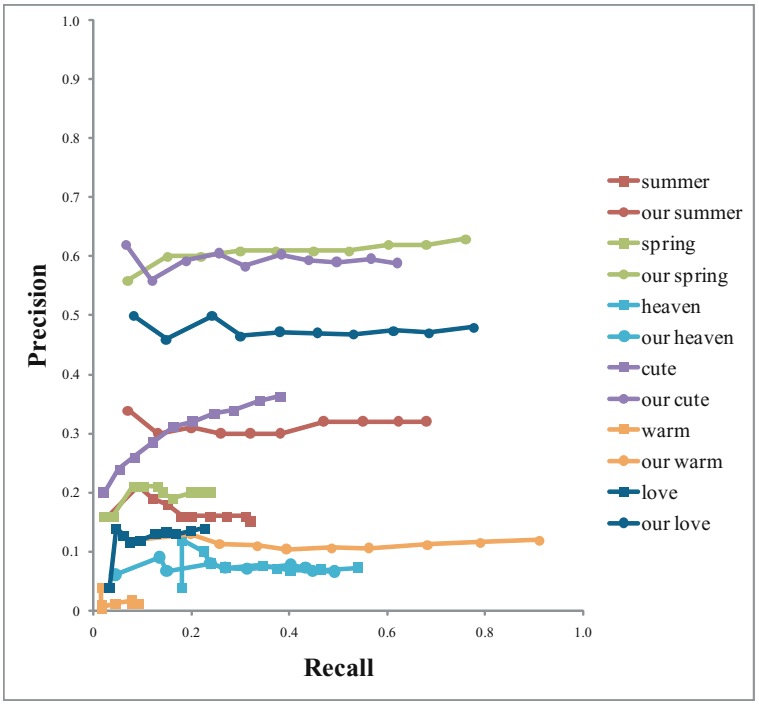


Fig. 7. Recall-precision curves for six abstract terms and corresponding sets of feature terms

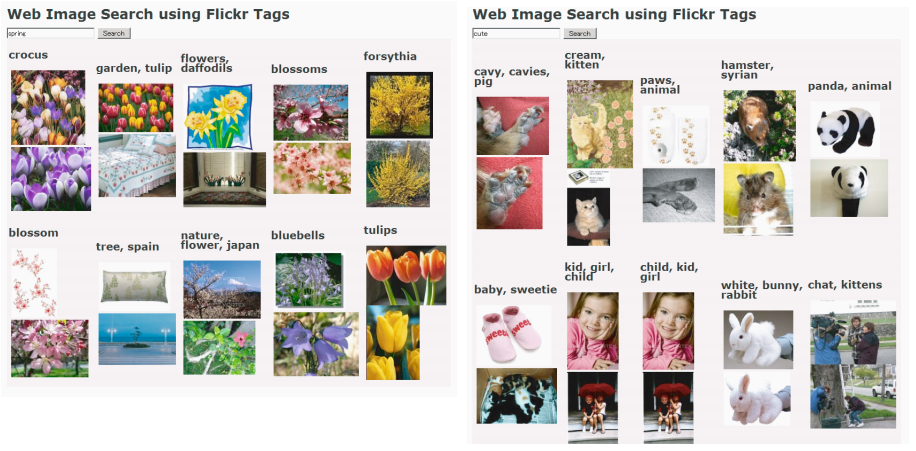


Fig. 8. Screenshots of results for "spring" and "cute" queries

In short, using sets of feature terms for an abstract term results in better identification of relevant images for the abstract term and using sets extracted from social tagging information improves Web image search.

6 Related Work

Kuwabara et al. used not only Web image data but also Web page data to improve Web image search [3]. They searched collections of Web pages with various search engines dedicated to searching a particular media type, took the intersection of the collection, and extracted images from it.

Social information has been used to improve Web search in many recent works [4][5]. Yanbe et al. analyzed social bookmark information and proposed a method that reranks results returned by a Web search engine by using social bookmarks as votes for pages[6]. Semantics has also been extracted from social tagging information in Flickr [7][8], and there have been analyses using Flickr regarding user behavior and the number of photos and tags [9][10].

Classification of queries for images was suggested by Hollink et al [11]. They identified a mismatch between user needs and current image search methods, that is, perceptual term queries are not used more frequently than conceptual queries by users. Although "content-based" image retrieval systems [12] have been developed, it is necessary to bridge the gap between the queries users input and the ones systems expect.

One approach to searching for images using conceptual term queries is to separate each image into segments and index each segment using a term representing the segment [13][14]. Another approach is to create a high-dimensional visual vocabulary that represents the image documents [15]. Even if these methods are applied, however, a problem remains: it is difficult to index images using abstract terms simply by analyzing the image contents. Our approach towards

discovering the relationship between an abstract term and concrete terms can contribute to the solution of the problem.

7 Conclusion

We have described a method that improves Web image searching with abstract term queries. It represents abstract terms with sets of concrete terms extracted from social tagging information. These sets of concrete terms that describe features of the abstract term and are associated with it. To extract these terms from the social tagging information, we cluster concrete terms taken from Flickr photo tags on the basis of image co-occurrence and evaluate them by applying an association rule for power sets, which comprise several of the tags in each cluster. We select as sets of feature terms those terms in each power set that have the highest confidence value and a support value above a threshold. We assessed the effectiveness of our method by comparing the results of image searches using only an abstract term with those for searches using corresponding sets of feature terms. The results show that our method improves the recall ratio while maintaining the precision ratio.

Although using sets of feature terms improves performance, there is room for further improvement. We did not consider the similarity between tags. Some feature vectors for the images were sparse, so the similarity between the vectors could not be calculated effectively. The dimensions could be compressed by using a method such as latent semantic indexing. We used WordNet for filtering out the tags with abstract terms; however, it does not work with proper names such as "sky tower" or "kinkakuji temple." In addition, it considers names of places to be only concrete terms. However, names of places sometimes represent concepts such as a culture, characteristic, or style. We plan to analyze social tags in Flickr and develop a method that distinguishes abstract terms from concrete ones by using social tagging information.

The sets of feature terms extracted in this work can also be used for other purposes. We are currently considering using them for image query modification using an abstract term. For example, users often want to modify the results of an image search by using an intuitive term such as "more springlike." In addition, we will focus on multi-keyword queries such as "peace spring" or "cute cat" and apply our approach to them.

In this work, we focused on improving Web image search by using social tagging information. Our approach can also be applied to other types of searches and should be especially effective for searching multimedia contents such as videos. We found that using concrete query terms is effective for image searches; for video and music searches, we should consider their unique properties when applying our approach.

Acknowledgements

This work was supported in part by the MEXT Grant-in-Aid for Scientific Research on Priority Areas entitled: "Contents Fusion and Seamless Search

for Information Explosion" (#18049041, Representative: Katsumi Tanaka) and "Design and Development of Advanced IT Research Platform for Information" (#18049073, Representative: Jun Adachi), by the MEXT project "Software Technologies for Search and Integration across Heterogeneous-Media Archives", and by the Kyoto University Global COE Program: "Informatics Education and Research Center for Knowledge-Circulating Society" (Representative: Katsumi Tanaka).

References

1. Miller, G., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.: Introduction to WordNet: An On-line Lexical Database. International Journal of Lexicography 3(4), 235–244 (2004)
2. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. Processings of the ACM SIGMOD 22(2), 207–216 (1993)
3. Kuwabara, A., Tanaka, K.: RelaxImage: A Cross-Media Meta-Search Engine for Searching Images from Web Based on Query Relaxation. In: Proceedings of the 21st ICDE, pp. 1102–1103 (2005)
4. Heymann, P., Koutrika, G., Garcia-Molina, H.: Can social bookmarking improve web search? In: Proceedings of the international conference on WSDM, pp. 195–206 (2008)
5. Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: Proceedings of the 16th international conference on WWW, pp. 501–510 (2007)
6. Yanbe, Y., Jatowt, A., Nakamura, S., Tanaka, K.: Can social bookmarking enhance search in the web? In: Proceedings of the conference on JCDL, pp. 107–116 (2007)
7. Schmitz, P.: Inducing ontology from flickr tags. In: Collaborative Web Tagging Workshop at WWW (2006)
8. Rattenbury, T., Good, N., Naaman, M.: Towards automatic extraction of event and place semantics from flickr tags. In: Proceedings of the 30th annual international ACM SIGIR, pp. 103–110 (2007)
9. Marlow, C., Naaman, M., Boyd, D., Davis, M.: HT06, tagging paper, taxonomy, Flickr, academic article, to read. In: Proceedings of the 17th conference on HT, pp. 31–40 (2006)
10. Lerman, K., Jones, L.: Social Browsing on Flickr. In: Proceedings of ICWSM (2006)
11. Hollink, L., Schreiber, A., Wielinga, B., Worring, M.: Classification of user image descriptions. International Journal of Human-Computer Studies 61(5), 601–626 (2004)
12. Veltkamp, R., Tanase, M.: Content-based image retrieval systems: A survey. Ultrecht, Netherlands: Department of Computing Science, Utrecht University (2000)
13. Duygulu, P., Barnard, K., de Freitas, N., Forsyth, D.: Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 97–112. Springer, Heidelberg (2002)
14. Lavrenko, V., Manmatha, R., Jeon, J.: A model for learning the semantics of pictures. In: Proceedings of Advance in Neutral Information Processing (2003)
15. Magalhaes, J., Rueger, S.: High-dimensional visual vocabularies for image retrieval. In: Proceedings of the 30th annual international ACM SIGIR, pp. 815–816 (2007)

Mashing Up Context-Aware Web Applications: A Component-Based Development Approach

Florian Daniel[1] and Maristella Matera[2]

[1] University of Trento, Via Sommarive 14, 38100 Povo (TN), Italy
daniel@disi.unitn.it
[2] DEI - Politecnico di Milano, Via Ponzio 34/5, 20133 Milano, Italy
matera@elet.polimi.it

Abstract. Context-awareness and adaptivity in web applications have been gaining momentum in web engineering over the last years, and it is nowadays recognized that, more than a mere technology aspect, they represent a first-class design concern. This acknowledgment has led to a revision of existing design methods and languages, finally resulting in runtime adaptation being considered a cross-cutting aspect throughout the whole development process. In this paper, we propose a radically new view on context-awareness and show how a well-done component-based development may allow the fast mashup of context-aware and adaptive web applications. The proposed approach comes with an intuitive graphical development environment, which will finally enable even end users themselves to mash up their adaptive applications.

1 Introduction

The current technological advances are shaping up various scenarios where people with different (dis)abilities interact with web applications through multiple types of mobile devices and in a variety of different contexts of use. That is, we are moving toward accessing at any time, from anywhere, and with any media customized services and contents. In such scenarios, the need for effective methodologies for the fast development of adaptive and context-aware web applications arises.

Adaptivity is intended as the autonomous capability of the application to react and change in response to specific events occurring during the execution of the application, so as to better suit dynamically changing user profile data. *Context-awareness* is intended as adaptivity based on generic context data, not limited to user profile data.

Some works already addressed adaptivity and context-awareness, spanning a number of perspectives: from the representation of context properties through formalized context models [1,2], to the definition of high-level modeling abstractions for the conceptual design of adaptive behaviors [3,4,5,6] (in Section 2 we discuss the cited approaches in detail). Typically, all these approaches share the same view over context-awareness and consider it an *explicit* design dimension, to be addressed with specific (sometimes complex) design artifacts.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 250–263, 2008.

In this paper we break with this interpretation and show how a *mashup approach* to the development of web applications may *implicitly* provide support for context-awareness. More specifically, we propose an innovative framework for the development of context-aware web applications, which tries to hide the complexity of adaptivity design, also fostering reuse. In line with the current trend supported by the recent Web 2.0 technologies [7], our approach indeed aims at empowering the users (both developers and end users) with an easy-to-use tool for mashing up applications by integrating ready (adaptive) services or application components.

The framework leverages our previous results in the design of context-aware web applications [6,8] and in the component-based development of web applications [9,10]. In particular, we exploit an event-driven paradigm, which enables the composition of self-contained, stand-alone applications (components) equipped with their own user interface [10,9], and show how context-aware applications can be composed by integrating *context components* with generic components. Context components are in charge of monitoring the context and generating *context events* when the context changes [8]. Such context events are then mapped onto operations of the generic components, which in this way are enabled to change their internal state to adapt to context changes. Provided that context components and generic components are respectively able to generate context events and react to such events, the adaptivity logic simply resides in the composition logic, which defines the synchronization of components without requiring any "ad-hoc" extension of the composition framework for the specification and implementation of adaptive behaviors.

To further facilitate the application development, the proposed approach is also complemented with an intuitive visual development environment, which will eventually enable even end users to mash up their adaptive applications.

1.1 Motivating Scenario

As a reference example throughout this paper, we have implemented a location-aware tourist guide through Trento, Italy. Trento is particularly suited to our application, as the city center, where all the interesting sights are located, is covered with free wireless Internet connection, providing for the necessary connectivity.[1] The application is a location-aware mashup of Google Maps and a tourist information system. Besides explicit user navigations and selections, the application also reacts to location changes tracked by means of a GPS device. Figure 1 shows a screen shot and explanations of the application.

1.2 Contributions

Besides introducing a novel development paradigm, this paper provides an innovative vision on context-aware web applications. More specifically, we provide the following contributions:

[1] `http://www.wilmaproject.org/`

Fig. 1. Screen shot of the location-aware tourist guide based on GPS position data

- We show how the event-driven paradigm of our component-based web application development approach [9,10] (Section 3) nicely suits the needs of context-aware applications. The resulting development approach (Section 4) hides most of the complexity of adaptivity design and fosters reuse.
- Consistently with the component-based approach, we show how also the context model, underlying all adaptive behaviors, can easily be mashed up, starting from so-called *context components*, generating context events (Section 4). Each context component is indeed in charge of monitoring and managing a separate context domain; therefore the "global" context model, needed to support adaptivity, is simply mashed up by composing context components.
- We equip the described approach with a Web 2.0 development and execution environment (Section 5), in order to enable also end users to mash up context-aware web applications. More precisely, we describe our easy-to-use, graphical editor and the light-weight, client-side execution environment.
- We finally summarize the main novelties of the proposed development method and provide an outlook over current and future works (Section 6).

2 Current Approaches to Context-Awareness

Context-awareness has been mainly studied in the fields of ubiquitous, wearable, or mobile computing. Several applications have been developed [11,12], and context abstraction efforts have produced platforms or frameworks for rapid prototyping and implementation of context-aware software solutions [13]. However, recently some efforts have also been devoted to the Wed domain; they principally deal (i) with the gathering of context data and their representation as proper *context models*, and (ii) with the identification of modeling abstractions, able to support the design of adaptive behaviors.

Belotti et al. [14] address the problem of the fast and ease development of context-aware (web) applications and propose the use of a universal context engine in combination with a suitable content management system [15]. In such a framework, context affects the actual web application indirectly by altering the state of the database and is not able to trigger autonomously application functionalities. Also, developers have to deal with a centralized context model, possibly integrating the heterogeneous data coming from different context sources.

At a more conceptual level, some well known model-driven methodologies (such as Hera [16], UWE [5], and WebML [6]) aid developers in the design of adaptive web information systems, by extending design models with concepts and notations to specify adaptive behaviors. For example, Ceri et al. [6] propose an extension of the WebML model, in which adaptive pages are associated with a chain of operations that implement the page's adaptivity logic and are executed each time a context monitor [8] (which restricts the analysis of the context model to the only properties relevant to the currently viewed page) demands for adaptation. The modeling of adaptive actions leverages a set of adaptivity-specific units, especially related to the acquisition and management of context data and the enactment of adaptation actions. In [17] and [18,19], the authors propose the use of event-condition-action rules for adaptivity specification and management.

The modeling approaches proposed so far allow developers to reason at a high level of abstraction. However, in all these approaches adaptivity design is strictly coupled with application design and requires the explicit and detailed specification of adaptivity rules and actions. The framework that we propose in this paper goes beyond these limitations and allows developers, or even end users, to concentrate on the global adaptive behavior of the application in an event-driven fashion, hiding the complexity of how single adaptation actions are executed – also fostering reuse, a typical feature of component-based development.

Finally, a family of approaches to personalization or adaptation is based on algebraic specifications and formal reasoning. For example, in [2] the authors extend SiteLang, a process algebra developed by the authors to express so-called application "stories". User preferences are specified by means of algebraic pre- and post-conditions that act as filters over a web information system's story space and tailor the algebraic expression of the story space to an individual user. Unfortunately, despite their effectiveness, such approaches make the implementation less intuitive, compared to both model-driven approaches and the component-based development proposed in this paper.

3 The Mixup Approach to Component-Based Web Application Development

In [10,9] we have shown an approach to the fast development of web applications based on the composition of components that are equipped with own User Interfaces (UIs), i.e., we have introduced a systematic approach to mash up web applications. The distinguishing characteristic of the proposed approach is that

it focuses on the integration of components at the presentation layer, leaving application and data management logic inside components. As illustrated in the rest of this section, component and composition models are inspired by research in the field of web services and the service-oriented architecture (SOA).

3.1 Component Model

A so-called *UI component* is characterized by an abstract external interface that enables its integration into composite applications and the setup of inter-component communications. UI components are proper stand-alone applications, whose *state* is represented by the portion of UI published by the component and by the value of some component-specific *properties*. Components may generate *events*, which communicate to the outside world changes in the internal state; events are component-specific and typically follow a high-level semantics, e.g. a component that provides tourist information may publish an event `sightSelected` to notify other components of the selection performed by the user.[2] Components may have *operations*, which enable the outside world to modify the internal state of a component; operations are component-specific and follow the semantics of events, e.g. the tourist information component may have a `showSight` operation, which allows one to emulate a user selection from the outside.

Similarly to WSDL for web services, components are abstractly described via so-called UISDL (UI Service Description Language) descriptors. The `tinfo.uisdl` file in Figure 2 shows for example the UISDL descriptor of the tourist information component used in our case study (see Figure 1); for presentation purposes, the example is kept simple (e.g. we omit data types and technology bindings). After a mandatory header, the descriptor specifies the actual component: its identifier, its location, and its operations and events with possible parameters.

3.2 Composition Model

Given the described abstract UI component model, the composition model can be kept simple. Indeed, we propose an *event-driven* model, where events from one component may be mapped to operations of one or more other components; mappings are expressed by means of so-called *listeners*. In addition to the direct mapping of events to operations, listeners also support *data transformations* in form of XSLT transformations, and the specification of more complex mapping logics via inline JavaScript. The definition of listeners represents the composition logic, while the *layout* of a composite application is specified by means of a suitable HTML template that contains placeholders, which can be used at runtime to embed and execute components, thereby re-using their UIs.[3]

[2] Low-level events such as mouse clicks or keystrokes do not represent meaningful events in the context of the proposed approach.

[3] In our current implementation, we focus on web technologies, but conceptually our framework may also span other UI technologies.

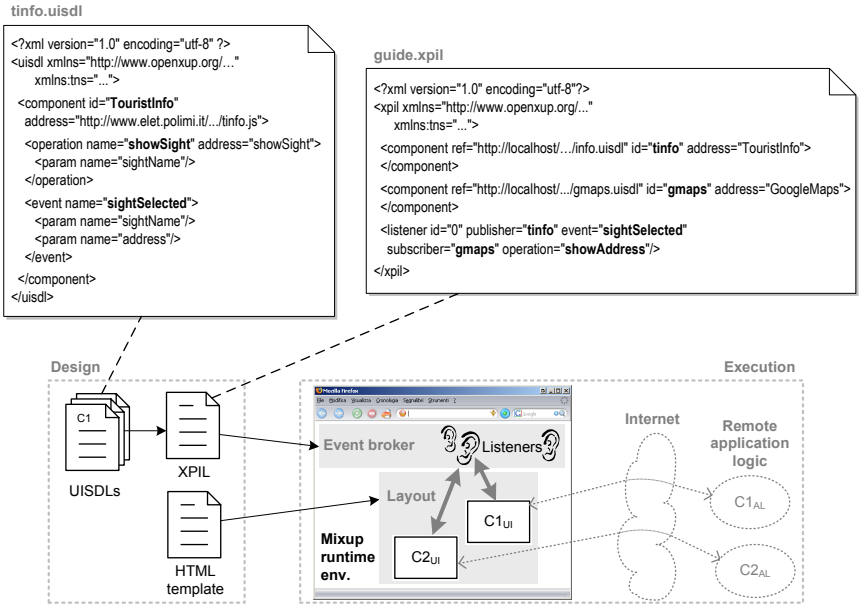


Fig. 2. The Mixup framework with UISDL descriptor and XPIL composition

The composition logic is expressed in XPIL (eXtensible Presentation Integration Language), an XML-based language specifying how UISDL-based components are integrated within single pages. The file `guide.xpil` in Figure 2 shows a simplified XPIL composition for our reference example (we omit complex data mappings or scripting). The composition refers to the two components in Figure 1: it references the UISDL descriptors of the tourist information component and of the Google Maps component and assigns unique identifiers (`tinfo` and `gmaps`). The identifiers are used in the specification of the listener that updates the map according to the user's selection of a sight: the `sightSelected` event of `tinfo` is mapped to the `showAddress` operation of `gmaps`. The HTML template corresponds to the HTML page shown in Figure 1 without the rendering of the two components.

3.3 The Mixup Framework

The lower part of Figure 2 summarizes the overall framework. At design time, we mash components up starting from their UISDL documents; the mashup is stored as XPIL composition equipped with an HTML template for the layout. At runtime, a JavaScript/AJAX environment running in the client browser parses the XPIL file, instantiates the components, and places them into the layout for the rendering of the composite page. As shown in the figure, UI components may internally communicate with their own remote application logic necessary for the

execution of the component; however, such communications are transparent to the designer who only focuses on the composition and layout logic.

4 Mashing Up Context-Aware Web Applications

Leveraging our experience on context-aware applications [6,18,19], in this section we show how the described mashup approach naturally lends itself to the development of context-aware and adaptive web applications. The proposed approach is characterized by an adaptation specification logic that is very simple and consistent with the previously described composition logic, thus elevating the abstraction level at which developers deal with adaptivity compared to the approaches discussed in Section 2.

4.1 Adaptivity Layers in Mixup

Context-awareness means runtime adaptation, i.e. *adaptivity*, in response to changes of context data, whose structure is expressed by means of some kind of context model. Let's first consider the typical adaptivity features of context-aware web applications, i.e. *what* we adapt. Considering the works discussed in Section 2, we can categorize the typical adaptivity actions supported in current adaptive/context-aware web applications into the following six features:

- *Adaptation of contents*: contents/data published in pages may be changed;
- *Enactment of operations*: external operations or services may be invoked;
- *Adaptation of style*: properties like colors and font sizes may be changed;
- *Adaptation of layout*: the arrangement of page contents may be re-organized;
- *Hiding/showing of links*: hyperlinks may be dynamically hidden or shown;
- *Automatic navigation actions*: hyperlinks may be automatically navigated on behalf of the user.

If we now consider our mashup approach, we can easily identify *two layers* at which adaptivity features may operate. The two layers are characterized by different adaptivity features:

1. *Component adaptations*: Components may internally support one, more, or all of the above features. Components are indeed small stand-alone applications, and as such they may be developed according to the approaches discussed in Section 2 and based on own context-data. Alternatively, they may expose operations to be invoked from the outside to trigger the supported features; in this paper we will focus on such kind of operations. Well-developed components are key for the success of mashups in general; here we build atop of current best practices and experience.
2. *Composition adaptations*: also the composite application may support one, more, or all of the above features (a composite application may have its own business logic, in addition to the components it integrates). But the composite application may also support adaptivity features over its composition logic, which are new with respect to the above features:

- *Hiding/showing of components*: components in the composition may be hidden or shown dynamically;
- *Re-configuration of listeners*: new listeners may be added or existing listeners may be dropped at runtime;
- *Selection of components*: new components may be dynamically selected (e.g. from a component registry) and added to the composition.

4.2 Enabling Context-Awareness through Context Components

But *to what* and *when* do we adapt our application? In short, we adapt to changes of context data, typically structured according to an application-specific context model, and we adapt as soon as such changes are registered by our application. It is exactly the triggering of adaptivity actions where the proposed composition approach shows its full power: instead of having a centralized context model that captures all context data and the respective changes, we use dedicated *context components*, which are in charge of observing a particular piece of context and of generating events in response to changes thereof. Such *context events* are then mapped to the operations of the components or of the composite application that are able to trigger changes to the internal state, i.e. to *adapt* the component or the whole application to changing context conditions.

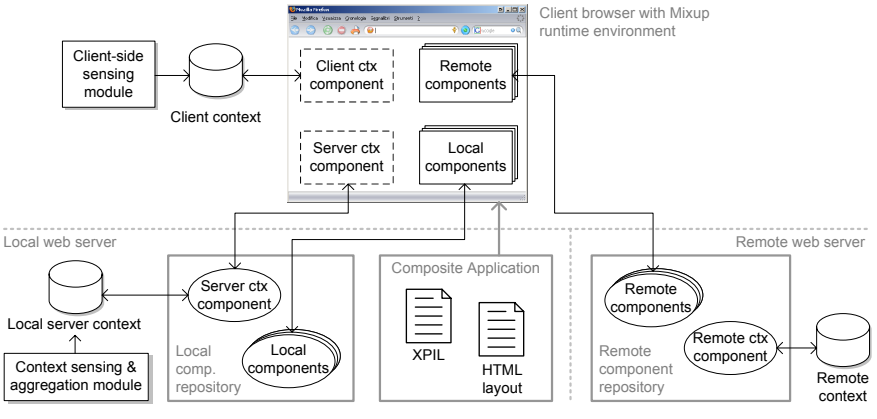


Fig. 3. The use of dedicated context components enables the development of context-aware web applications

Figure 3 shows how context components seamlessly extend the Mixup framework depicted in Figure 2, effectively enabling context-aware mashups. We distinguish between client-side and server-side context components. *Client-side* components fully run on the customer's device and enable the communication of *client context* data sensed via dedicated sensing modules. *Server-side* components run their business logic (i.e. the specific context management logic) on the server side, thus enabling the communication of context data representing

centrally sensed or aggregated data. There may also be *remote* context components, which enable the access to context data that are outside of the control of the developer.[4] For instance, the following represent a few possible and reasonable context components:

- *Location component*: a component, implemented at the client- or at the server-side[5], which may fire "low-level" position events, for example regarding longitude and latitude, or "high-level" events, for example regarding cities, streets, or countries;
- *User Model component*: a server-side component in charge of monitoring changes to a user model typically stored at the server side. The component could be used to provide advanced (e.g. adaptive) personalization features;
- *Time component*: a client-side component that may fire whatever event (specified at design time) at given time instants or intervals. The component could expose a set of configuration operations that allow the designer to set up the events to be emulated;
- *System component*: a server-side component that exposes runtime data about the health or performance of the running application;
- *Shared context component*: a server-side component based on context data aggregated at the server side, which e.g. may allow the generation of events that express the presence of other people in a same physical location;
- *Weather component*: a remote component that accesses weather forecasts provided by third-party service providers and supplies users with forecast data depending on their current GPS position.

As can be seen in the above examples, context components typically concentrate on one specific *context domain*. Therefore, we do not have any centralized reference *context model*. Just like the whole application itself, also the context model is simply "mashed up" by putting together the context components that are needed. We are hence in presence of a modular approach to construct the context model, which transparently integrates context data from disparate sources, possibly distributed over the Web.

It is finally worth noting that context components that generate context events are per definition *active* components that are able to communicate context data without an explicit user intervention, thus solely based on the dynamics of context. This therefore leads us to interpret context as an *independent actor*, working on the same application as users do [8].

[4] The described use of context components is fully in line with the idea of *context monitor* discussed in [8].

[5] The logic of this component depends on the mechanism adopted for capturing context data. For example, components implementing proprietary sensing mechanisms can be required on the client side, as it happens for GPS-equipped devices, or server-side components can be used to communicate to the client context data collected through centralized sensing infrastructures, such as those based on infrared signals or RFID.

4.3 Mashing Up Context-Aware Web Applications

As explained above, context components generate events that can easily be mapped onto other components' operations (just like with conventional UI components) in order to trigger adaptivity features. When developing context-aware composite applications, the XPIL composition therefore also reflects the *adaptivity logic*, which is expressed by means of the listeners that specify how context events influence other components. The *layout* of context-aware applications is instead slightly different from non context-aware applications, as context components typically come without own UI, reason for which they are typically invisible in the composite application. Using context components in the Mixup framework does therefore not imply the need for any extension to the existing runtime environment nor to the composition model.

The adaptive features that can be used when developing a context-aware mashup very much depend on the capabilities of the components to be integrated. Well-designed components come with a rich set of operations, which may enable a wide spectrum of adaptive behaviors. As already discussed in earlier, *component adaptations* depend on the internal implementation of components (which typically is out of the control of the mashup developer), while *composition adaptations* can be freely defined by the developer.

It is worth noting that from a technical perspective the distinction between context components and conventional components is very blurred. Whether a component is a context component or not rather depends on the overall *semantics* of the application to be developed. As a matter of fact, each component that generates events may be considered a context component, as its events may be used to trigger adaptivity features in other components.

To equip our example application of Figure 1 with the necessary location-aware behavior, we use a location context component that tracks the user's position in terms of street name and number. Also, the operation `showSight` of the tourist information component accepts addresses in input and, if a corresponding sight can be found, publishes the respective details; otherwise, no changes are performed. In Figure 2 we introduced the XPIL logic for the integration of the UI components only. In order to achieve context-awareness, the following code lines need to be added to the XPIL composition of Figure 2 (`guide.xpil`):

```
<component ref="http://localhost/.../gps.uisdl" id="location"
   address="Location_Context">
</component>
<listener id="1" publisher="location" event="streetChanged"
                 subscriber="gmaps" operation="showAddress"/>
<listener id="2" publisher="location" event="streetChanged"
                 subscriber="tinfo" operation="showSight"/>
```

The first three lines import the location context component. The remaining lines define two listeners, one that couples the `streetChanged` event of the `location` component with the `showAddress` operation of the `gmaps` component,

and one that couples the same event with the `showSight` operation of the `tinfo` component. The two listeners specify the adaptivity logic of the context-aware application.

4.4 Termination and Confluence of Adaptive Behaviors

Although in the proposed approach we allow developers to specify cyclic dependencies among listeners, at runtime the execution environment does not allow for cycles. More precisely, at runtime possible events generated by a component in response to the invocation of one of its operations (the invocation is triggered by another event) are neglected, thus effectively preventing the cascaded or cyclic invocation of listeners. This implies that possible multiple dependencies from one and the same event must be explicitly modeled through suitable listeners (one for each dependency). Listeners reacting to the same event are activated concurrently, and the final result of their execution depends on the order of the invocation of the listeners. Therefore: *termination* of page computation is guaranteed, as there are no cyclic runtime dependencies among listeners; and *confluence* depends on the order in which events are generated and – for listeners activated on a same event – on the order in which listeners over a same event are specified in the XPIL composition. Consequently, there are no indeterministic behaviors in the evaluation of listeners, and the designer has full control over the runtime behavior of the composite context-aware application.

5 Implementing Context-Aware Mashups

The development approach described in this paper is assisted by a visual development environment, which allows for the drag-and-drop composition of context-aware web applications [9]. Both composition logic and layout can be easily designed; for the layout, it is also possible to upload an own HTML template with placeholders. Figure 4 shows the graphical composition of the logic of our reference application: the three components (Google Maps, tourist information, and location context component) are represented by the icons in the modeling canvas; the listeners are represented by the connections among the components. A double click on components and listeners allows one to set the necessary parameters.

The editor is an AJAX application that runs in the web browser. Compositions can be stored on our web server and executed in a hosted fashion through the Mixup runtime environment, a JavaScript runtime library that parses the XPIL file and sets up the necessary communication among the components.

The location context component used in the case study is based on previous work [8]: it leverages a client-side Bluetooth GPS device, interfaced via the Chaeron GPS Library[6], and is wrapped by means of Flash (to exchange position data between the component and the GPS library) and JavaScript (to provide a UISDL-conform interface). The other two components are conventional UI components.

[6] http://www.chaeron.com/gps.html

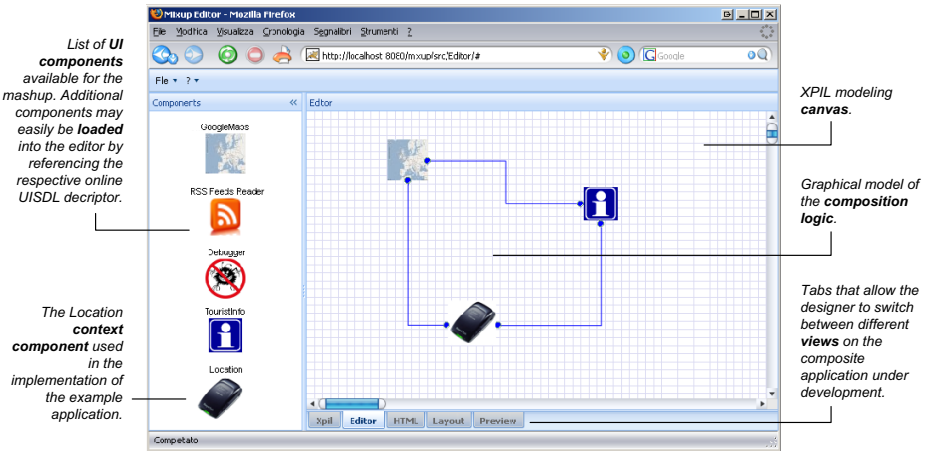


Fig. 4. The Mixup editor for the drag-and-drop mashup of XPIL and layout logic

6 Discussion and Outlook

Runtime adaptivity and context-awareness are relevant aspects in the design of modern web applications. Indeed, if we consider how they have been addressed so far by the most prominent conceptual design methods, we note that they have been treated like an explicit, first-class design concern. The fashion in which we develop context-aware or adaptive web applications in this paper allows us to make the most of such approaches, but it also goes *beyond* what has been done so far in this area. Provided a set of ready components, with the described mash up approach we indeed enable developers (or even end users) to focus on adaptive behaviors only, completely hiding the complexity of how adaptations are actually carried out.

Adaptivity is enabled and supported by self-contained stand-alone applications, i.e. UI components. The actual adaptivity logic is represented by means of a simple composition language, such as XPIL, interpreted during runtime by a light-weight runtime environment.[7] The simplicity of composing adaptive web applications, however, does not come for free: complexity resides *inside* the components, which provide for the necessary business logic to generate events and enact operations. In this setting, the design of the components (both UI and context components) becomes *crucial*, as the events they generate and the operations they support build up the expressive power of the adaptivity logic,

[7] Due to the highly UI- and event-based logic of our composition approach, we have opted for UISDL/XPIL, instead of for instance standard web service languages such as WSDL/BPEL, which do not natively support UIs. Although theoretically WSDL/BPEL could be used for similar purposes, this would however require extending the two languages with new features, which would only get the already complex languages more complex. We instead believe that our specific context demands for languages that are easily intelligible and easily executable (e.g via client-side code).

which, hence, varies from application to application, depending on the components that are adopted. It is exactly the design of UISDL-compliant components that benefits most from existing approaches to the development of adaptive or context-aware web applications, since, as a matter of fact, components can be considered "traditional" web applications equipped with a UISDL API.

The described composition model based on XPIL does not only enable the mashing up of context-aware and adaptive applications, it also allows for the easy introduction of adaptivity features into already *existing* applications. It suffices to add a respective context component, generating suitable context events (e.g. changes in user characteristics, preferences, locations, etc.), to the composition and to map its events to the components of the composite application, in order to obtain adaptive behaviors. The graphical XPIL editor described in this paper also supports the modification of an application's adaptivity logic during *runtime*, thus paving the road (i) for *fast prototyping* and easy testing of otherwise complex application features during application design and (ii) for the efficient and consistent *evolution* of an application after its deployment.

When we first introduced our approach to the component-based development of web applications [10,9] we were driven by the event-based paradigm by means of which typically user interfaces are developed in order to achieve what we called "presentation integration" (in analogy with data and application integration); we did actually not focus on adaptive application features. But a close look at the result of this effort allowed us to identify analogies with our previous research on adaptive and context-aware web applications [6,18,19]: mashing up context-aware web applications based on the described framework effectively means setting up *adaptation rules*, the listeners. The developed framework is thus intrinsically adaptive, a property that we heavily leverage in this paper.

In our future work we will focus on data transformations between events and operations, on complex navigation structures spanning multiple composite pages, on the integration of generic web services in a UISDL-compliant fashion, as well as on classical personalization features by means of dedicated user model components.

References

1. Henricksen, K., Indulska, J.: Modelling and using imperfect context information. In: PerCom Workshops, pp. 33–37. IEEE Computer Society, Los Alamitos (2004)
2. Schewe, K.D., Thalheim, B.: Reasoning about web information systems using story algebras. In: Benczúr, A.A., Demetrovics, J., Gottlob, G. (eds.) ADBIS 2004. LNCS, vol. 3255, pp. 54–66. Springer, Heidelberg (2004)
3. Fiala, Z., Hinz, M., Houben, G.J., Frasincar, F.: Design and implementation of component-based adaptive web presentations. In: Haddad, H., Omicini, A., Wainwright, R.L., Liebrock, L.M. (eds.) SAC, pp. 1698–1704. ACM, New York (2004)
4. Frasincar, F., Barna, P., Houben, G.J., Fiala, Z.: Adaptation and reuse in designing web information systems. In: ITCC (1), pp. 287–291. IEEE Computer Society, Los Alamitos (2004)

5. Baumeister, H., Knapp, A., Koch, N., Zhang, G.: Modelling Adaptivity with Aspects. In: Lowe, D.G., Gaedke, M. (eds.) ICWE 2005. LNCS, vol. 3579, pp. 406–416. Springer, Heidelberg (2005)
6. Ceri, S., Daniel, F., Matera, M., Facca, F.M.: Model-driven development of context-aware Web applications. ACM Transactions on Internet Technology 7 (2007)
7. Daniel, F., Yu, J., Benatallah, B., Casati, F., Matera, M., Saint-Paul, R.: Understanding UI Integration: A survey of problems, technologies. Internet Computing 11, 59–66 (2007)
8. Ceri, S., Daniel, F., Facca, F.M., Matera, M.: Model-Driven Engineering of Active Context-Awareness. World Wide Web Journal 10, 387–413 (2007)
9. Yu, J., Benatallah, B., Casati, F., Daniel, F., Matera, M., Saint-Paul, R.: Mixup: a Development and Runtime Environment for Integration at the Presentation Layer. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, pp. 479–484. Springer, Heidelberg (2007)
10. Yu, J., Benatallah, B., Saint-Paul, R., Casati, F., Daniel, F., Matera, M.: A Framework for Rapid Integration of Presentation Components. In: Proceedings of WWW 2007, pp. 923–932. ACM Press, New York (2007)
11. Want, R., Hopper, A., Falcao, V., Gibbons, J.: The Active Badge Location System. ACM Trans. Inf. Syst. 10, 91–102 (1992)
12. Long, S., Kooper, R., Abowd, G.D., Atkeson, C.G.: Rapid Prototyping of Mobile Context-Aware Applications: The Cyberguide Case Study. In: MOBICOM, pp. 97–107 (1996)
13. Salber, D., Dey, A.K., Abowd, G.D.: The Context Toolkit: Aiding the Development of Context-Enabled Applications. In: CHI, pp. 434–441 (1999)
14. Belotti, R., Decurtins, C., Grossniklaus, M., Norrie, M.C., Palinginis, A.: Interplay of Content and Context. J. Web Eng. 4, 57–78 (2005)
15. Grossniklaus, M., Norrie, M.C.: Information Concepts for Content Management. In: Huang, B., Ling, T.W., Mohania, M.K., Ng, W.K., Wen, J.R., Gupta, S.K. (eds.) WISE Workshops, pp. 150–159. IEEE Computer Society, Los Alamitos (2002)
16. Vdovjak, R., Frasincar, F., Houben, G.J., Barna, P.: Engineering Semantic Web Information Systems in Hera. J. Web Eng. 2, 3–26 (2003)
17. Garrigós, I., Gómez, J., Barna, P., Houben, G.J.: A Reusable Personalization Model in Web Application Design. In: WISM 2005 (2005)
18. Daniel, F., Matera, M., Morandi, A., Mortari, M., Pozzi, G.: Active rules for runtime adaptivity management. In: Workshop Proceedings of ICWE 2007, pp. 28–42 (2007)
19. Daniel, F., Matera, M., Pozzi, G.: Managing Runtime Adaptivity through Active Rules: the Bellerofonte Framework. Journal of Web Engineering 7, 179–199 (2008)

Correlating Time-Related Data Sources with Co-clustering

Vassiliki Koutsonikola[1], Sophia Petridou[1], Athena Vakali[1,⋆], Hakim Hacid[2,*], and Boualem Benatallah[2,*]

[1] Aristotle University of Thessaloniki
[2] University of New South Wales

Abstract. A huge amount of data is circulated and collected every day on a regular time basis. Given a pair of such datasets, it might be possible to reveal hidden dependencies between them since the presence of the one dataset elements may influence the elements of the other dataset and vice versa. Furthermore, the impact of these relations may last during a period instead of the time point of their co-occurrence. Mining such relations under those assumptions is a challenging problem. In this paper, we study two time-related datasets whose elements are bilaterally affected over time. We employ a co-clustering approach to identify groups of similar elements on the basis of two distinct criteria: the direction and duration of their impact. The proposed approach is evaluated using time-related news and stock's market real datasets.

1 Introduction

A huge amount of data is circulated and collected every day on a regular time basis in order to understand and capture various physical and commercial phenomena. Such data involve the recording of electricity demand, temperature and percentages of humidity, fluctuations in a company's sales and a stock's price etc over a specific time period. It is also true that there is a high level of dependencies between such different datasets since, for example, weather changes affect electricity consumption, news reporting influences stock prices, commercial advertisements have impact on consumers acts etc.

A common choice is to model such datasets in the form of event time series which capture measurements describing the raw data at successive (often uniform) time intervals. Analyzing different datasets measurements considering time as their common feature may reveal dependencies and relevance between datasets which otherwise might not be obvious. Considering for example a time series that records the temperatures measured on a daily basis over the summer period and another one recording the values of energy demands for the same period, dependencies between temperatures and quantities of energy demands

⋆ This work was supported by the DEST-ISL Competitive Grants Programme on "Efficient Management of Information Resources Over Ad-Hoc Data Grids", 2007-2008, UNSW, Australia.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 264–279, 2008.

can be found. Such interactions may be further used in forecasting in order to predict future outcomes. News and market datasets are a quite representative such scenario, since, for example, news announcements would be an indication of either the increase or decrease in specific stocks' values. These datasets exist parallel in time and finding relationships and/or dependencies between them will largely impact investors, traders, journalists and news agencies. Relevance between these datasets seems to be of high interest since currently almost all major news agencies offer market information (such as stocks rates, flow) on their portals' index page.

Table 1. Related Work

Datasets	Representation		Methodology
	vectors	time series	
Financial news articles, stock market data [13]	✓		k-NN learning algorithm, regression analysis, neural networks
Newsgroups articles, historical stock market data [12]	✓		Natural Language Processing, Time Delay Neural Network
Archives of news articles and stock data [6]	✓	✓	Segmentation, Clustering, Support Vector Machines
News headlines, quoted exchange rate data [11]	✓		Classification rules
Historical data of electricity consumption and weather [10]		✓	Regression models

Existing approaches that attempt to find dependencies between time-related datasets combine ideas from various research areas such as data mining, engineering and mathematical programming. A summarization of these approaches is given in Table 1. According to the authors knowledge, all earlier work considers that there exists a "one-way" impact between the involved datasets. For example, works in [13], [12], [6] and [11] study the impact of news on financial data while in [10] electricity demands are forecasted in terms of meteorological parameters. However, these dependencies are often bidirectional as the authors realized from their experience in the ADAGE project [1]. In financial world, for example, a banks' corporation announcement could trigger significant stocks' fluctuation while a stock-market's crash would certainly raise political statements. Besides the direction, the duration of these interactions is also of high interest since the impact of one event to another may last for a period instead of the time point of their co-occurrence.

In this paper, we use co-clustering to simultaneously yield clusters of elements from two different but related over time datasets. Co-clustering is proposed as a more suitable approach, which has been used in grouping together elements from different datasets [2], [3], [7]. We apply the proposed model on coinstantaneous news and financial datasets to reveal dependencies between them. It was quite

[1] ADAGE project: http://cgi.cse.unsw.edu.au/ soc/adage/

challenging to deal with such distinct datasets due to their different formats and scales and their preprocessing was carried out by proposing data structures tailored to each dataset and resulting in formats that could be commonly processed. The experimentation results show that the proposed framework manages to reveal relations between news and stocks with respect to their short and long term interactions. Our main contribution is summarized in proposing a framework for clustering mutually affected datasets, by considering the following criteria:

- *duration of impact* is taken into account to construct the data sources representation structures (vectors). A time window parameter w is used to extend original data sources vectors in a common time-aware format which will involve duration in the co-clustering process.
- *direction of influence* is "embedded" in the calculation of similarities between data sources. A weight factor $\alpha \in [0 \dots 1]$, which characterizes the role of a data source ranging from a triggering to a triggered status, is employed.

The remainder of the paper is organized as follows. Section 2 defines our problem whereas Section 3 analyzes the proposed co-clustering approach. Section 4 provides the experimentation on both synthetic and time-related news and stocks market data. The conclusions are presented in Section 5.

2 Problem Formulation

Consider two time-related datasets where elements of the first dataset trigger elements belonging to the other and vice versa. We define $A = \{a_1, \dots, a_n\}$ to be the first dataset containing n elements, $B = \{b_1, \dots, b_m\}$ to be the second dataset consisting of m elements and $T = \{1, \dots, t\}$ to be the set of t time points. Then, for each a_i element, $i = 1, \dots, n$, we define the vector $VA(i,:)$ to track its t measurements:

$$VA(i,:) = (VA(i,1), \dots, VA(i,t)) \tag{1}$$

where $VA(i,l)$, $l = 1, \dots, t$, indicates the value of the element a_i during the time point l. All the $VA(i,:)$ vectors are organized in the n x t table VA. For

Table 2. Basic symbols notation

Symbol	Description
n, m	Number of elements in the two datasets
t	Number of time points
$A = \{a_1, \dots, a_n\}$	Set of the first dataset elements
VA	nxt table of the A dataset measurements
$B = \{b_1, \dots, b_m\}$	Set of the second dataset elements
VB	mxt table of the B dataset measurements
$T = \{1, \dots, t\}$	Set of time points
w	Time window

the second dataset B, we similarly define the VB two dimensional m x t table which consists of m $VB(j,:)$ multidimensional vectors that provide the values of element b_j, $j = 1, \ldots, m$, over time:

$$VB(j,:) = (VB(j,1), \ldots, VB(j,t)) \tag{2}$$

The $VB(j,l)$ element indicates the value of b_j during the time point l.

Based on the above, we consider that a measurement of element a_i (b_j) on time point l i.e. $VA(i,l)$ ($VB(j,l)$) affects measurements on elements b_j (a_i) for a period starting from l and lasting up to $l + w - 1$ (w is the so-called time window), $w = 1, \ldots, t$. In practice, for $1 \leq l \leq t - w + 1$ the time duration of impact is w, whereas for $t - w + 2 \leq l \leq t$ the corresponding impact time period may last from $w - 1$ down to 1.

In the proposed approach, it is important to identify the problem to be solved, since as mentioned earlier we are dealing with two distinct criteria: the bilateral relation between our datasets and the time period of their interaction (expressed by w). More specifically, given the set A of n elements, the set B of m elements, the set T of t time points, the time window w and the desired number of clusters k we are looking for k subsets such that each subset contains both a_i and b_j elements. Members of each subset should be strongly related in terms of both the direction and duration of their impact. Thus, based on the above, we can define the Time-Related Data Co-Clustering problem as follows:

*Problem 1 (*Time-Related Data Co-Clustering). Given two time-related datasets A and B of n and m elements respectively, a set T of t time points, the integers w and k and a *Similarity* function, find a set C of k subsets $C = \{C_1, \ldots, C_k\}$ such that $\sum_{x=1}^{k} \sum_{a_i, b_j \in C_x} Similarity(a_i, b_j)$, $i = 1, \ldots, n$ and $j = 1, \ldots, m$, is maximized.

The *Similarity* function should capture our two main criteria, namely the bilateral influence between data elements and its duration.

3 Capturing Impact between Datasets

3.1 Measuring Similarities

A common measure used to capture similarity between two (same dimension) vectors is the *Cosine Coefficient* [14] which calculates the cosine of the angle between them. The cosine coefficient $CC(i,j)$ between two vectors $VA(i,:)$, where $i = 1, \ldots, n$, and $VB(j,:)$, where $j = 1, \ldots, m$, of the same length t is defined as follows:

$$CC(i,j) = \frac{VA(i,:) \cdot VB(j,:)}{|VA(i,:)| \cdot |VB(j,:)|} = \frac{\sum_{l=1}^{t} VA(i,l) \cdot VB(j,l)}{\sqrt{\sum_{l=1}^{t} VA(i,l)^2 \cdot \sum_{l=1}^{t} VB(j,l)^2}} \tag{3}$$

Using cosine coefficient to measure similarities between the vectors $VA(i,:)$ and $VB(j,:)$, we manage to capture their relation under the assumption that

$VA(i,l)$ and $VB(j,l)$ values can be related on the basis of time l. In our framework, a_i elements may trigger b_j elements and at the same time, b_j elements may trigger a_i elements for up to w successive time points after the triggering element has occurred. However, the correlation coefficient applied on $VA(i,:)$ and $VB(j,:)$ captures neither the bilateral relations nor the impact over the time window w. Thus, we firstly extend the $VA(i,:)$ and $VB(j,:)$ vectors in order to include time window w. Then, we use the correlation coefficient in conjunction with the extended vectors and we create two n x m tables, namely the AB and BA to represent the impact of a_i elements on b_j and vice versa.

Definition 1 (The extended triggering vector)**.** *Given a t-dimensional triggering vector $VA(i,:)$ with measurements of a period of t time points, and a time window w ($w = 1,\ldots,t$), we define its extended triggering t_1-dimensional vector $VAE(i,:)$, $t_1 \geq t$, which is constructed by putting up to w repetitive $VA(i,l)$ values, between each pair of the $VA(i,l)$ and $VA(i,l+1)$ measurements.*

Lemma 1. *Let $VAE(i,:)$ be the extended triggering vector of $VA(i,:)$. Given that the $VA(i,:)$ consists of t measurements which will be repeated up to w successive times and t_1 represents the number of values of the $VAE(i,:)$ vector, it holds that:*

$$t_1 = t * w - \frac{w(w-1)}{2} \tag{4}$$

Proof. We split the t values of the $VA(i,:)$ vector into two groups. The first contains the values l, where $1 \leq l \leq t-w+1$, which will be repeated w successive times. The second group contains the values l, where $t-w+2 \leq l \leq t$, which will be repeated from $w-1$ down to 1 times (i.e. the $t-w+2$ value will be repeated $w-1$ times, the $t-w+3$ value will be repeated $w-2$ times and that will continue until the last t value that will be presented 1 time). This second group contains a number of elements equal to an arithmetic sequence from 1 to $w-1$. As a result:

$$t_1 = (t-w+1) * w + \frac{w-1}{2} * w = t * w - \frac{w*(w-1)}{2}$$

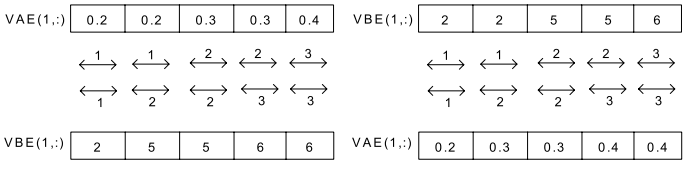
Definition 2 (The extended triggered vector)**.** *Given a t-dimensional triggered vector $VB(j,:)$ of t measurements of a period of t time points, and a time window w ($w = 1,\ldots,t$), we define its extended triggered t_2-dimensional vector $VBE(j,:)$, $t_2 \geq t$, which is constructed by putting a group of up to w values $VB(j,l)\ldots VB(j,l+w-1)$ between each pair of the $VB(i,l)$ and $VB(i,l+1)$ measurements.*

Lemma 2. *Let $VBE(j,:)$ be the extended triggered vector of $VB(j,:)$. Given that the $VB(j,:)$ consists of t measurements which will be repeated as groups of maximum w successive values and t_2 represents the number of values of the $VBE(i,:)$ vector, it holds that:*

$$t_2 = t * w - \frac{w(w-1)}{2} \tag{5}$$

Proof. The proof is similar to that of Lemma 1.

Example 1. Let *VA(1,:)=[0.2 0.3 0.4]* denote the vector with the measurements describing data element a_1 for 3 time points and *VB(1,:)=[2 5 6]* the vector of element b_1 for the same time period. We assume that $w = 2$. Considering that a_1 triggers b_1, the value 0.2 of a_1 that was recorded on time point 1 affects the values 2 and 5 of b_1 on time points 1 and 2. Similarly, the value 0.3 of a_1 will affect values 5 and 6 of b_1 while the last value 0.4 of a_1 will affect only the last value 6 of b_1. Figure 1(a) shows the extended vectors and their relation. On the other hand, considering that b_1 triggers a_1 the extended vectors and their relation are depicted in Figure 1(b). □



(a) a_i measurements trigger b_j measurements (b) b_j measurements trigger a_i measurements

Fig. 1. The extended vectors for $w = 2$

From Lemmas 1 and 2, it holds that $t_1 = t_2$ and thus, we can use the $VAE(i,:)$ and $VBE(j,:)$ vectors, instead of $VA(i,:)$ and $VB(j,:)$, in Equation 3 to calculate their similarity. Since the impact of time window w is "embedded" into the extended vectors the cosine coefficient will capture the duration of impact between elements. Equations 4 and 5 indicate that the cost in terms of space complexity shifts from $O(t)$ to $O(tw)$ i.e. the space burden posed by the extended vectors depends on w value chosen.

Based on the above, we create the n x m AB and BA tables in order to proceed to the impacts' direction criterion. The $AB(i,j)$ element indicates the similarity between $VAE(i,:)$ and $VBE(j,:)$ (evaluated by Equation 3) in case that $VAE(i,:)$ acts as the triggering and $VBE(j,:)$ as the triggered vector, while the $BA(i,j)$ element indicates the similarity between the former vectors in case that $VBE(j,:)$ triggers the $VAE(i,:)$ vector. We notice that the values of elements of the AB and BA tables fall in the interval $[-1..1]$ as $CC(i,j)$ expresses the cosine of the angle that these vectors define.

Next, we combine the information of tables AB and BA in the formula of the $Similarity(a_i, b_j)$ function via a weight factor α:

$$Similarity(a_i, b_j) = \alpha * AB(i,j) + (1 - \alpha) * BA(i,j) \tag{6}$$

The values of α fall in the interval [0..1] differentiating the significance of the bilateral relation between a_i and b_j data elements. More specifically, when $\alpha = 1$, $Similarity(a_i, b_j) = AB(i,j)$, we consider solely the impact of a_i to b_j, while for $\alpha = 0$, $Similarity(a_i, b_j) = BA(i,j)$, we consider that only b_j affects a_i. For any

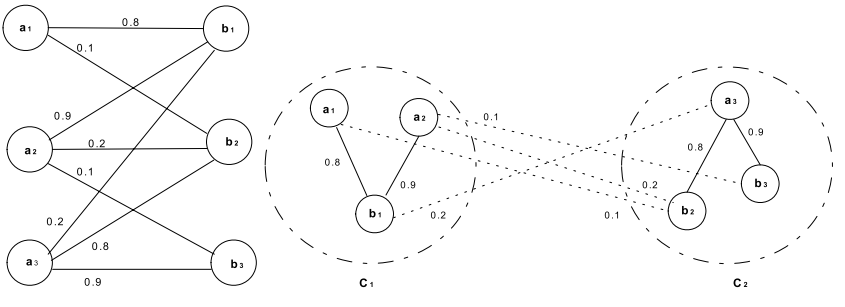
other value of α, we consider a mutual impact between a_i and b_j balanced by the choice of α. Thus, the *Similarity* function (Equation 6) captures our two distinct criteria, since α balances the direction of the bilateral relations between a_i and b_j and duration period w is "embedded" in the construction process of the AB and BA tables (since they are created by the extended triggering and triggered vectors).

3.2 Dataset Representation

Applying a typical clustering algorithm on the table structured defined in previous subsection would yield clusters of elements from only one dataset. Since our problem deals with a simultaneous clustering we need a convenient data structure which, on one hand, will keep the information stored in vectors and tables' and, on the other hand, will enable the datasets co-clustering. A graph is such a convenient structure, since it can represent relations between two sets of elements [1] and it has been already used in co-clustering approaches [2], [7]. In our case, we use a bipartite graph with its vertices and edges indicating the datasets' elements and their similarities respectively. Edges relate data elements on the basis of the direction and duration of their relations, the two criteria of the *Similarity* function.

We assume an undirected bipartite graph $G = (A, B; E)$ where $A = \{a_1, a_2, a_3\}$ and $B = \{b_1, b_2, b_3\}$ are two sets of vertices corresponding to the time-related datasets and E is the set of edges $\{\{a_i, b_j\} : a_i \in A, b_j \in B\}$ connecting nodes in A and B as depicted in Figure 2(a). In this bipartite model, the $\{a_i, b_j\}$ edges are undirected since the relation between a_i and b_j is bidirectional and is expressed by the *Similarity* function (Equation 6), while there are no edges between elements in A or between elements in B.

In practice, the *Similarity*(a_i, b_j) serves as an edge weighting function which captures both the direction and duration of impact between a_i and b_j. Given the



(a) The weighted undirected bipartite graph G

(b) Cut of the bipartite graph G

Fig. 2. Data representation and partitioning

similarity values between each pair of related a_i and b_j vertices, we can define the n x m symmetric table WE to express the weights of graph's edges:

$$WE(a_i, b_j) = Similarity(a_i, b_j) \tag{7}$$

Example 2. In Figure 2(a), $WE(1,1) = 0.8$ while $WE(1,2) = 0.1$ indicating that the element a_1 is more related with the element b_1 than with the element b_2. □

According to Problem 1 each of the k subsets $C_1, C_2, \ldots, C_k$ includes elements from both datasets such that a_i and b_j will belong to the same C_x, where $x = 1, \ldots, k$, once their $WE(a_i, b_j)$ contributes to the maximization of $\sum_{x=1}^{k} \sum_{a_i, b_j \in C_x} Similarity(a_i, b_j)$. Let us consider the graph G depicted in Figure 2(a) and assume that we want to create $k = 2$ clusters. It is obvious that the 2-partitioning of Figure 2(b) should maximize the sum of similarities between elements of the same clusters (intra-clusters sum) and minimize the sum of similarities between elements of different clusters (inter-clusters sum). The last, inter-clusters sum is expressed by:

$$\sum_{a_i \in C_x} \sum_{b_j \in C_y} WE(a_i, b_j) \tag{8}$$

where $x, y = 1, \ldots, k$ and $x \neq y$, and corresponds to the *cut* of the graph G. Its minimization provides a solution to the graph partitioning problem [4]. Therefore, Problem 1 is transformed into a graph k-partitioning problem.

3.3 The Co-clustering Approach

The k-partitioning of the graph G implies the creation of k groups with elements originating from both A and B datasets which is our co-clustering approach's goal. Moreover, given that the edges of the graph G carry information about the direction and duration of impact between elements (Equation 7), our co-clustering approach considers these two criteria.

Given that our problem has been transformed into a cut minimization problem on a weighted undirected graph, we define structures that will provide meaningful information about the underlying data model and will contribute in the co-clustering of a_i and b_j elements. Since the importance of a graph's vertex is characterized by its degree we define two degree tables, namely the n x n diagonal table D_A and the m x m diagonal table D_B. In the weighted graph G the degree of a vertex is the sum of the edges' weights adjacent to it. Thus, the D_A contains the degree of the a_i elements while the D_B consists of the degrees of the b_j elements. Based on WE (Equation 7) we define D_A and D_B as follows:

$$D_A(i,i) = \sum_{j=1}^{m} WE(a_i, b_j), i = 1, \ldots, n$$

$$D_B(j,j) = \sum_{i=1}^{n} WE(a_i, b_j), j = 1, \ldots, m$$

Inspired from spectral clustering approaches in [2] and [7] and given the D_A, D_B and WE tables, we create the n x m two dimensional table WAB:

$$WAB = D_A^{-1/2} WED_B^{-1/2}$$

The $D_A^{-1/2}, D_B^{-1/2}$ tables originate from the diagonal tables D_A, D_B respectively and are used in order to result in the normalized table WAB.

As it has been proved [2], the k left and right singular vectors of WAB will give us a k-partitioning of a_i and b_j elements respectively. We denote as L_A the n x k table of the left singular vectors and R_B the m x k table of the right singular vectors. In order to perform a simultaneous clustering of a_i and b_j elements, we create the $(n + m)$ x k two dimensional table SV defined as:

$$SV = \begin{bmatrix} D_A^{-1/2} L_A \\ D_B^{-1/2} R_B \end{bmatrix}$$

Then, we obtain the desired k clusters by running a typical clustering algorithm such as k-means [9] or fuzzy c-means [5] on SV.

Algorithm 1. The Co-clustering algorithm.

Input: The A and B datasets containing n and m elements respectively over the t time points, the integers w and k and the factor a, where $\alpha \in [0 \dots 1]$.

Ouput: A set $C = \{C_1, \dots, C_k\}$ of k subsets consisting of elements from both A and B such that the sum of inter-clusters similarities defined by (8) is minimized.

1: */*Preprocessing*/*
2: $(VA, VB) = Preprocess(A, B)$
3: */*Capturing duration of impact via w*/*
4: $(VAE, VBE) = ExtendedVectors(VA, VB, w)$
5: */*Capturing direction of impact*/*
6: $AB = CosineCoefficient(TriggeringVAE, TriggeredVBE)$
7: $BA = CosineCoefficient(TriggeringVBE, TriggeredVAE)$
8: $WE = a * AB + (1 - a) * BA$
9: */*Co-clustering*/*
10: $(D_A, D_B) = DegreeTables(WE)$
11: $WAB = D_A^{-1/2} WED_B^{-1/2}$
12: $(L_A, R_B) = SingularVectors(WAB)$
13: $SV = IntegratedTable(D_A, D_B, L_A, R_B)$
14: $C = FuzzyCmeans(SV, k)$

The Algorithm 1 solves the Time-Related Data Co-Clustering problem taking into account the two distinct criteria, namely the direction and duration of datasets' relations. More specifically, during the preprocessing step we build the VA and VB tables according to the Equations 1 and 2. Considering that each row of these tables constitutes a vector, we incorporate time window w in their corresponding extended vectors VAE and VBE. Then, based on these extended vectors we form the AB and BA tables indicating the direction of relations as

described in Subsection 3.1. These relations are quantified by the similarities recorded in the table *WE* (Equation 6). The factor α weights the significance of the bilateral relations. Once the *WE* is created, we proceed to the co-clustering step. We create the D_A and D_B degree tables and then the *WAB* on which we apply a singular value decomposition in order to obtain the k left and right singular vectors organized in tables L_A and R_B respectively. We integrate D_A, D_B and L_A, R_B into SV on which we run the fuzzy c-means clustering algorithm. The algorithm finalizes the k clusters which contain elements from both A and B datasets. In fuzzy c-means the data elements can be assigned to all clusters with different probability. This is preferable compared to a hard clustering approach (e.g. k-means), since we are aiming at a flexible clustering scheme.

4 Experimentation

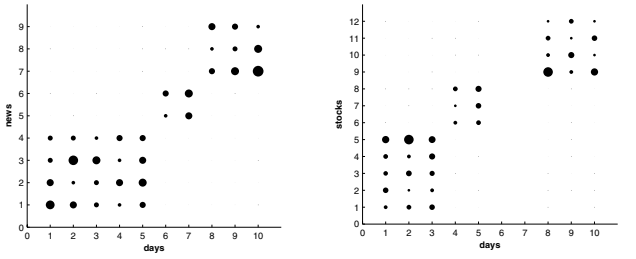
To evaluate the proposed approach we carried out experiments on both synthetic and real datasets. Both types of datasets were needed in order to capture the perspective of the proposed algorithm and its actual behavior. In case of the synthetic data we choose not to experiment with a large-scale dataset in order to be able to drill down on the results [7]. However, our algorithm is scalable and applicable in larger-scale datasets as indicated by the real data experimentation. The algorithm's results are discussed in order to give an insight of its applicability and importance.

4.1 Synthetic Data

We consider the datasets A and B of sizes 9 and 12 respectively over 10 time points. To facilitate the results' discussion we assume that A represents news articles, B corresponds to stock market data and the time period refers to 10 days starting from Monday. Then, the news frequency and stocks' fluctuation, recorded on tables *VA* and *VB* respectively, are visualized on Figures 3(a) and 3(b). The size of circles ranges to denote the values of *VA* and *VB* tables. Thus, all elements of *VA* exhibit some value on different days while in *VB* there are no values during the days 6 and 7 because these days refer to a weekend during which there is no "traffic" in stocks market.

As depicted in Figure 3 we consider $k = 3$ groups of elements in each dataset which are clearly separated over time. In particular, in Figure 3(a) we have the groups $\{a_1, a_2, a_3, a_4\}$, $\{a_5, a_6\}$ and $\{a_7, a_8, a_9\}$ while in Figure 3(b) the elements are arranged as $\{b_1, b_2, b_3, b_4, b_5\}$, $\{b_6, b_7, b_8\}$ and $\{b_9, b_{10}, b_{11}, b_{12}\}$. We experimented tuning the factor α to the values $0.1, 0.5, 0.9$ and fixing the time window to the indicative value $w = 3$. Certainly, in practice, there are events that their impact lasts only one to two days or, to the other extent, whole months but these are the exception.

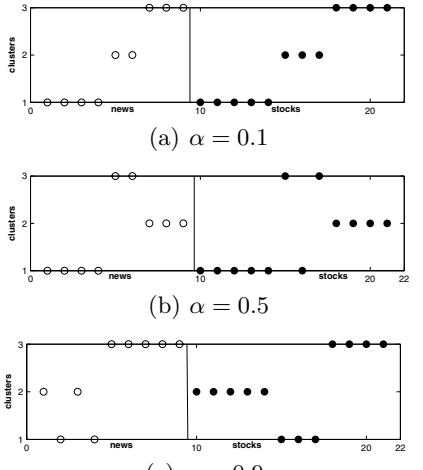
Due to the lack of space we present a small portion of the results in Figure 4. According to Figures 4(a), 4(b) and 4(c), it is apparent that our co-clustering succeeds in grouping together elements from both datasets. Moreover, we highlight results of different α which show the behavior of our approach in terms of



(a) The 9 x 10 news articles table *VA*

(b) The 12 x 10 stock's market data table *VB*

Fig. 3. Synthetic data: the input tables



(a) $\alpha = 0.1$

(b) $\alpha = 0.5$

(c) $\alpha = 0.9$

Fig. 4. Co-clustering over synthetic data for $k = 3$ and $w = 3$

the direction and duration of data interactions. More specifically, from Equation 6 it holds that when $\alpha = 0.1$ our approach considers mainly the impact of stocks fluctuation on news articles. Based on this we can explain the result of Figure 4(a) where, for example, the news $\{a_5, a_6\}$ and stocks $\{b_6, b_7, b_8\}$ belong to the same cluster (C_2) indicating that stocks fluctuation at the end of a week affects news during weekend. For $\alpha = 0.9$ (Figure 4(c)) it seems that a different co-clustering was produced where, for example, the news elements $\{a_5, a_6, a_7, a_8, a_9\}$ were grouped together with stocks $\{b_9, b_{10}, b_{11}, b_{12}\}$ (C_3) denoting that news announcements during weekend have impact on the week stock market's opening. In case that $\alpha = 0.5$ (Figure 4(b)), where there is a mutual interaction between news and stocks, the co-clustering is similar to that of the Figure 4(a) with the exception that the group $\{b_6, b_7, b_8\}$ were split in two clusters. The $\{b_6, b_8\}$ and $\{a_5, a_6\}$ belong to the same cluster (C_3) since the value

$w = 3$ captures the duration of their impact. Specifically, $\{b_6, b_8\}$ occur on Thursday and Friday and influence $\{a_5, a_6\}$ which occur on Saturday and Sunday and vice versa. On the other hand, cluster C_1 consists of $\{b_1, b_2, b_3, b_4, b_5, b_7\}$ and $\{a_1, a_2, a_3, a_4\}$ which all present values during the first 5 days.

4.2 The News and Market Paradigm

Data Workload. The proposed approach has been applied on real news and stock market datasets in order to find groups of related news topics and stocks. Our news dataset was retrieved by the Reuters [2] news archives for a time period of 6 months (June 2004 - November 2004). Codes and especially topic codes are normally used to identify or categorize a news story. Thus, we experimented with the news data based on their topic codes. More specifically, we handled news topics by calculating their daily frequency, since several topic codes may be assigned to a news story, and the idea was to recover the most interesting topics and discard not frequent topics or outliers. The retrieved news data was categorized in 290 topics. However, calculating the daily frequency of each topic for the 6 months i.e. $t = 183$ days we conclude in keeping the 49 most popular topics i.e. $n = 49$ topics (the rest refer to topics with either low i.e. 10 or high i.e. 40000 values of frequency). Thus, the first dataset input to our algorithm is the 49 x 183 news topics' frequency table *VA*.

Our stock-market data was retrieved by the Standard & Poor 500 index (S&P) historical data [3]. The S&P 500 carries information of about 500 companies and their stocks which fall into 119 categories according to the company's operation field (e.g. the *AAPL* stock of the Apple Computer's company belongs to the Computer Hardware category). The retrieved stock market data refers to the same time period i.e. June 2004 - November 2004. The source file (raw data) contains one record per stock and per day. The total number of stocks is $m = 410$ while the 6 month period consists of $t = 183$ days. Given the stock's daily open and close values (no data are given for weekends and holidays since stock markets are closed) we can define its fluctuation which describes the stock's behavior adequately. We calculate absolute values of stocks daily fluctuations since our aim was to find groups of related stocks that are strongly (either positive or negative) affected by news topics and vice versa. Thus, we create the 410 x 183 stocks' fluctuation *VB* table which is our algorithm's second dataset input.

Real Data Experiments. We run the Co-clustering algorithm on the news and stocks datasets tuning the factor α to the values 0.1, 0.5, and 0.9 in order to consider three distinct scenarios where news and stocks are mutually affected, while we initially fixed the time window to $w = 3$. This value of w enables us to observe short term influences which would probably be of the interest of small scale or occasional investors. Furthermore, we experimented with a higher value of time window i.e. $w = 10$ in order to reveal long term interactions which

[2] Reuters: http://www.reuters.com/
[3] S&P500 historical data : http://kumo.swcp.com/stocks/

concern long term traders, financiers or bankers. In our implementation, any value of w in the interval $w = 1, \ldots, t$ can be given, since w is a parameter. However, we select the above values of w, since $t = 183$ and, in practice, the dependencies between news and stock markets data rarely approximate the value of t. Finally, due to the lack of space, results are presented for $k = 4$ and $k = 5$ clusters.

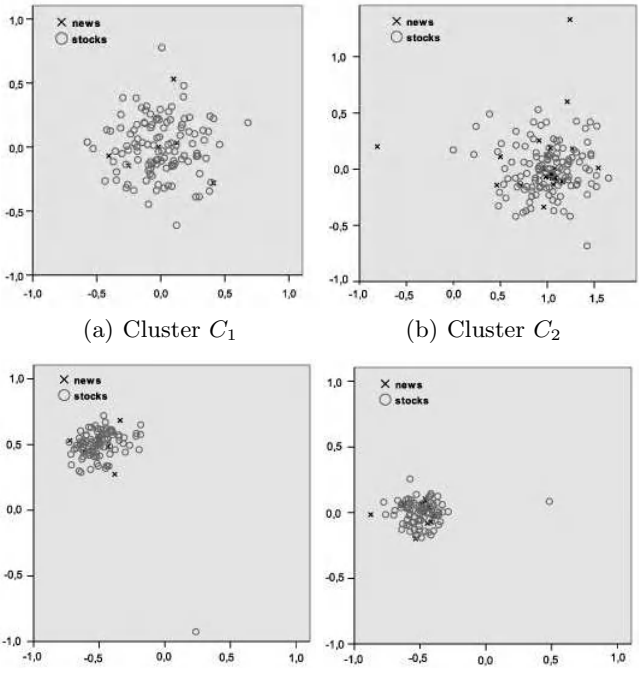
Table 3 presents co-clustering results in terms of the number of members in each cluster for $k = 5$ and for different values of w and α. We observe that the obtained clusters relate elements from both datasets, since there is no cluster consisting solely of news or stocks. Moreover, both news topics and stocks are distributed in a balanced way to the $k = 5$ clusters for $w = 3$ and $\alpha = 0.1, 0.5, 0.9$. Increasing the time window to $w = 10$ the clusters' membership changes, while the clusters remain balanced for the same values of α. Having a greater time window contributes to understanding the duration of impact of news to stocks (and vice versa). This is explained by the fact that an increased (decreased) number of stocks/news topics related to particular news topics/stocks in a cluster shows a long (short) term impact.

Table 3. Cluster members for different values of w and a

	$w = 3$						$w = 10$					
	$a = 0.1$		$a = 0.5$		$a = 0.9$		$a = 0.1$		$a = 0.5$		$a = 0.9$	
	news	stocks	news	stocks	news	stocks	news	stocks	news	stocks	news	stocks
C_1	7	78	22	90	16	95	5	69	9	82	6	90
C_2	11	98	8	70	5	86	4	74	10	93	17	79
C_3	6	97	6	103	6	60	20	91	6	75	8	82
C_4	5	73	6	77	4	89	9	91	9	87	7	70
C_5	20	64	7	70	18	80	11	85	15	73	11	89

Apart from the denoted clusters' balance in terms of their membership it is important to study their quality too. Thus we have proceeded to a correspondence analysis [8] for visualizing the associations between news and stocks in each of the obtained clusters. We indicatively present the algorithm's results for $k = 4$, $w = 3$ and $a = 0.5$ in Figure 5. In each subfigure, which depicts one of the obtained clusters, the "x" marker corresponds to news while the circle marker to stocks. It is apparent that the coclustering algorithm creates groups of closely related news and stocks as indicated by the clusters compactness. Moreover, the obtained clusters are well separated, while there is no overlap between them.

In addition, it would be interesting to attempt a more "conceptual" analysis of the algorithm's results. The algorithm succeeds in grouping news topics and stocks that according to the their "nature" seem to be related. For example, topics that describe news about central banks and interest rates (e.g. CENT and INT) were assigned to the same cluster with stocks referring to regional banks and insurance (e.g. BK, CMA and MI). Similarly, the topic DRU related to pharmaceutical and health care and the stocks that refer to health care distributors, pharmaceutical and health care equipment (e.g. ABC, AET and CAH)



(a) Cluster C_1 (b) Cluster C_2

(c) Cluster C_3 (d) Cluster C_4

Fig. 5. Correspondence analysis results for $k = 4$

were grouped together. In the last example, we observed that fixing time window to $w = 3$ and tuning α to the values $0.1, 0.5, 0.9$ there are specific stocks such as ABC, AET and CAH that remain related to the topic DRU. Independently whether the news "prevail" over stocks ($\alpha = 0.9$) or stocks over news ($\alpha = 0.1$), it seems that there exists a "core" of elements that remain related. Indicatively, for $\alpha = 0.1, 0.5, 0.9$ and $w = 3$ there are $18, 14, 19$ stocks related to the topic DRU respectively. For the corresponding values of α and $w = 10$ the topic DRU was related with $17, 15, 16$ stocks. Some of these stocks are the same with those in case that $w = 3$. Thus, the short term interactions of topic DRU are different compared to the long term ones but there are also interactions which remain stable in spite of the time period.

Studying the above results is useful to parties or individuals of different interests and perspectives, since many people are involved in recommending, trading, publicizing and circulating of both news and market data. For example, the proposed co-clustering would be beneficial for traders who could proceed to recommending certain stock trading acts to their clients, based on same cluster topic codes and stocks for a preferable time interval. Also, news agencies representatives may proceed to certain news topics announcements, based on intense stocks traffic which under co-clustering is shown to influence news.

5 Conclusions

This paper introduces a co-clustering approach which yields clusters consisting of elements belonging to two different but related over time datasets under two main criteria: the direction and duration of data interactions. The Co-Clustering algorithm differentiates data similarity via α and w parameters. The factor α balances the direction of the bilateral relations, while the time window w shifts the duration of data impact. The proposed algorithm has been evaluated using real workloads which correspond to news articles and stock market data. The results showed that our approach succeeds in creating balanced clusters whose membership varies according to α, indicating the news that affect specific stocks and vice versa, as well as according to w revealing different relations with reference to short and long term periods of interactions. Understanding the resulted clusters would facilitate acts of investors, traders, bankers, journalists, news agencies etc.

References

1. Afrati, F., Das, G., Gionis, A., Mannila, H., Mielikainen, T., Tsaparas, P.: Mining Chains of Relations. In: Proc. of the 5th IEEE Int. Conf. on Data Mining, ICDM, pp. 553–556 (2005)
2. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proc. of the 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining, KDD, pp. 269–274 (2001)
3. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-Theoretic Co-clustering. In: Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, KDD, pp. 89–98 (2003)
4. Ding, C., He, X., Zha, H., Gu, M., Simon, H.: A Min-max Cut Algorithm for Graph Partitioning and Data Clustering, pp. 107–114 (2001)
5. Dunn, J.C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics 3, 32–57 (1973)
6. Fung, G., Xu Yu, J., Lam, W.: News Sensitive Stock Trend Prediction. In: Proc. of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, PAKDD, pp. 481–493 (2002)
7. Gao, B., Liu, T., Zheng, X., Cheng, Q., Ma, W.: Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In: Proc. of the 11th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data mining, KDD, pp. 41–50 (2005)
8. Greenacre, M.J.: Correspondence Analysis in Practice. Academic Press, London (1993)
9. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Heidelberg (2001)
10. Mirasgedisa, S., Sarafidis, Y., Georgopoulou, E., Lalas, D.P., Moschovits, M., Karagiannis, F., Papakonstantinou, D.: Models for mid-term electricity demand forecasting incorporating weather influences. Energy 31, 208–227 (2006)
11. Peramunetilleke, D., Wong, R.: Currency exchange rate forecasting from news headlines. In: Proc. of the 13th Australasian database conference, ADC, vol. 24, pp. 131–139 (2002)

12. Sagar, V.K., Kiat, L.C.: A neural stock price predictor using qualitative and quantitative data. In: Proc. of 6th Int. Conf. on Neural Information Processing, ICONIP, vol. 2, pp. 831–835 (1999)
13. Wuthrich, B., Cho, V., Leung, S., Permunetilleke, D., Sankaran, K., Zhang, J., Lam, W.: Daily Stock Market Forecast from Textual Web Data. In: Proc. of IEEE Int. Conf. On System, Man and Cybernetics, vol. 3, pp. 2720–2725 (1998)
14. Zhang, J., Korfhage, R.: A Distance and Angle Similarity Measure Method. Journal of the American Society for Information Science 50, 772–778 (1999)

Context-Aware Mashups for Mobile Devices

Andreas Brodt[1], Daniela Nicklas[1], Sailesh Sathish[2], and Bernhard Mitschang[1]

[1] Universität Stuttgart, Institute of Parallel and Distributed Systems,
Universitätsstraße 38, 70569 Stuttgart, Germany
{brodt,nicklas,mitsch}@ipvs.uni-stuttgart.de
[2] Nokia Research Center, Visiokatu 1, 33720 Tampere, Finland
sailesh.sathish@nokia.com

Abstract. With the Web 2.0 trend and its participation of end-users more and more data and information services are online accessible, such as web sites, Wikis, or web services. So-called mashups—web applications that integrate data from more than one source into an integrated service—can be easily realized using scripting languages. Also, mobile devices are increasingly powerful, have ubiquitous access to the Web and feature local sensors, such as GPS. Thus, mobile applications can adapt to the mobile user's current situation.

We examine how context-aware mashups can be created. One challenge is the provisioning of context data to the mobile application. For this, we discuss different ways to integrate context data, such as the user's position, into web applications. Moreover, we assess different data formats and the overall performance. Finally, we present the TELAR mashup platform, a client-server solution for location-based mashups for mobile devices such as the Nokia N810 Internet Tablet.

1 Introduction

The proliferation of public web services and other online data sources enables new services and applications that combine existing information in a new manner. So-called mashups integrate data and services from multiple sources to provide innovative services, like the visualization of crime statistics from the Chicago Police Department on a street map[1] (one of the first mashups). Mashups are mostly realized by web pages that leverage script languages such as JavaScript, which enables better user interactivity by locally executed functions and the dynamic loading of data from web services. These techniques are often associated with the Web 2.0 trend. Typically, mashups are created dynamically from existing data sources that have no knowledge about their participation. Thus, they can change their interfaces and data formats at any time, which adds a new flavor to the general data integration problem.

Another trend are mobile systems that have become more and more powerful in the last years. Soon, users expect their handheld devices to run the same applications as their desktop systems do. In addition to that, mobile devices feature

[1] http://chicagocrime.org

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 280–291, 2008.

an increasing number of built-in or easily connectable sensors, such as cameras, GPS receivers, or acceleration sensors. One example is the Nokia N810 Internet Tablet that possesses 128 MB of RAM, 2 GB of flash memory, an OMAP2420 microprocessor at 400 MHz as well as a small camera and a GPS receiver. This is sufficient to run a Linux-based operating system and thus numerous Linux applications, including context-aware software, such as car navigation.

However, mobile systems still have limitations in many resources, such as display size, communication bandwidth (due to their wireless connections), energy, or the user's focus. The latter imposes one of the major design issues: mobile applications should be able to adapt to the user's situation. Every piece of information that helps the application to detect that situation is referred to as context. Such adaptive applications are called context-aware applications [1].

The combination of these two trends—mashups and context-aware applications on mobile devices—offers great additional value to the user. By integrating multiple data sources into one experience, new services can be created that are tailored to the user's personal needs. And by using local sensor data on a mobile device, this experience can be adapted to the user's current situation. In this paper, we examine the creation of context-aware mashups according to the following requirements:

- Adaptation should be based on sensors which are built into the mobile device or locally connected. Although our scenario focuses on location data gained from a GPS receiver, the solution should allow arbitrary local sensors.
- The mashups should be user-centric, i. e., the user of a mobile device should benefit from the mashups, rather than a remote person or service provider.
- The mashups should be viewable with the web browser of the mobile device. This prevents a native solution on the mobile device and has potential influence on performance.
- There should be a non-adaptive version of mashups in case no context information is available. This allows viewing the mashups on sensor-equipped mobile devices as well as on desktop computers and thus increases the usefulness of the mashups.
- Multiple data sources using arbitrary data formats and interfaces should be integrated into the mashups. The user should be able add and remove data sources at runtime, according to her current interest.

The example scenario on which we focus in this paper is a context-aware mashup which integrates the user's position obtained from a GPS device, a map from an online map service, and nearby points of interest (POIs) from different online data providers, as depicted in Figure 2. The POI metaphor is used by a large portion of geo-mashups, so scenario covers the main use case of these mashups. We put this scenario into practice using a three-tier system architecture with a Nokia N810 Internet Tablet as the client device. We demonstrated this sample scenario at the EDBT'08 conference [2].

The main contribution of this paper is a system architecture featuring a context provisioning framework for utilizing local sensors in context-aware mashups.

We give an overview on related work in Section 2. In Section 3 we identify the requirements of a context provisioning framework for context-aware mashups, address data integration, and describe our proposed system architecture. We shortly give a few implementation details of the Telar mashup platform in Section 4, discuss the performance of AJAX on mobile devices and illustrate our optimizations. Finally we conclude the paper in Section 5.

2 Related Work

2.1 Context Provisioning

Context is any information that can be used to characterize the situation of an entity [1], which is used to adapt the behavior of a context aware application. In our example scenario, the entity would be the user, and context are his or her location and interest. While the interest—i.e., which data providers should be included in the mashup—is configured by the user, the location should be provided automatically to the mashup system, i.e., by a sensor.

In related research, several context provisioning systems were proposed, e. g., the Context Toolkit [3], the context manager of Henriksen et al. [4], or the Nexus platform [5], to name a few. Unfortunately, none of these approaches are mature enough to build them into a running product, not to talk about standardization. Also, they are meant to manage a complex infrastructure of installed sensors and context services, which is not necessary for context-aware mashups.

On the web, context-aware web pages currently integrate context information obtained from third-party web services. Today, web services locating the user's IP address, such as hostip.info, are able to determine the user's country and occasionally the town. This can be sufficient to embed advertisements into a web page. Services like plazes.com or jaiku.com support uploading context information, such as the current location and activity. Some people embed this information into their web page or blog in order to share it with friends. In principle, it is possible to upload context data to a third-party service. However, this approach cannot satisfy the time and accuracy requirements of systems such as an electronic tour guide.

Heading towards device independent web applications several standards have been defined for supplying context information of a kind. HTTP headers may provide information about the client's web browser and could be extended to convey other context data [6]. However, such extended HTTP headers lack asynchronous notifications, search, control and standardization. Approaches based on profiles, such as CC/PP [7] and UAProf [8], provide only static context attributes, thus lacking asynchronous notifications and mutability. The Device Profile Evolution (DPE) [9] solves this problem, but is limited to mobile phones and follows a server-based approach. However, to reduce latency and increase accuracy, client-based provisioning is desirable.

The context provisioning framework that matches our requirements best is the W3C Delivery Context Client Interfaces [10]. DCCI is a client-based framework

to which both local sensors and remote services can be bound. DCCI represents context as a number of *properties* which are organized hierarchically, using the W3C Document Object Model (DOM). Web pages can access the resulting property tree via JavaScript. DCCI properties may be static (e. g., screen size) or dynamic (e. g., the user's position). The main advantage of DCCI is that it is undergoing standardization within W3C and is (at the time of writing) expected to reach proposed standard status soon.

2.2 Mashups

Mashups are web application hybrids that integrate data from different sources to provide a value added service. They leverage the availability of open web services, RSS feeds, or extract information out of regular web pages using screen scraping. A popular combination is to display information from different sources on a map (geo-mashups), or to enrich search results from one source (e. g., a hotel finder) with information from others (e. g., recommendations and pictures).

The mashup portal programmableWeb.com lists 3046 mashups on May 20, 2008, with an average of 3.14 new mashups per day. Over one third are geo-mashups, 17% are multimedia mashups (video and photo). Floyd et al. [11] show how mashup techniques can be used for rapid prototyping in user-centered software development processes. A study at the Human-Computer Interaction Institute of the Carnegie Mellon University showed that mashups can be even used for end-user programming [12]. IBM emphasizes the great benefits of so-called *Enterprise Mashups* [13], information heavy applications that integrate distributed information within an enterprise in a quick and dynamic way. Erik Wilde [14] applied the mashup idea to the management of large knowledge bases.

Most of the existing mashups are programmed manually. However, a number of mashup platforms exist that facilitate the development: *Mash-o-matic* [15] can be used to generate geo-mashups based on so-called superimposed information. The *Openkapow* platform[2] realizes mashups as a combination of so-called robots, which extract information from RSS streams, web services, or via screen scraping. With online tools like *Yahoo! pipes*[3] or Microsoft's *Popfly*[4], mashups can be built out of predefined components and combined using interactive drag-and-drop interfaces. IBM's *QEDWiki*[5] is an AJAX interface to combine user interface components that are connected to external data providers. IBM is also working on a data mashup service for web and enterprise information called *DAMIA*[6]. Intel's *MashMaker* [16] allows the creation of complex mashups by browsing, rather than writing code, applying the principles of functional programming.

While these platforms are good at integrating various data sources into a new presentation, none of them is capable of utilizing local sensors. Also, our scenario focuses on independent points of interest (POIs), thus complex interaction

[2] `http://openkapow.com`
[3] `http://pipes.yahoo.com`
[4] `http://www.popfly.ms`
[5] `http://services.alphaworks.ibm.com/qedwiki`
[6] `http://services.alphaworks.ibm.com/damia`

between data providers, e. g., as supported by Yahoo! pipes or MashMaker, is not necessary. On the other hand, our scenario may require a lot of flexibility with respect to accessing the data providers.

3 Context-Aware Mashups

3.1 Context Provisioning

Context-aware mashups (and context-aware web-applications in general) based on local sensors require a means of context provisioning with the following characteristics:

Asynchronous Notifications. The mashup needs to be notified of changes in the user's context in order to react accordingly.

Mutability. As sensors may become inactive or new sensors may be connected to the mobile device, the context model needs to be capable of reflecting these changes.

Search. The mashup needs a means to find out, which kinds of context data are available. This in turns requires **metadata** describing the context data.

Control. In order to utilize a local sensor, the mashup needs some degree of control over the sensor, e. g., to activate it or to trigger a measurement.

Standardization. It is of utter importance to have standardized interfaces when exposing an API to a huge multi-platform and multi-vendor system such as the web.

Privacy. The user must be in control of what information is disclosed about her current situation. As mashups are third-party applications, the context provisioning system must ensure privacy before the context data reaches the mashups. The privacy aspect of context provisioning, however, is not in the focus of this paper.

As discussed in Section 2.1, not every context provisioning framework fulfills these requirements. We chose the W3C Delivery Context Client Interfaces (DCCI) [10] as a standardized means for context provisioning, since it possesses most of these characteristics. DCCI uses the DOM event model to provide asynchronous notifications. A web page can register for events, such as the change of a property value or the removal of a property, and is notified via the provided JavaScript event listener function. In addition to that, it is possible to add and remove properties dynamically, thus achieving mutability. Moreover, DCCI properties can have a metadata interface and the DCCI tree is searchable. DCCI does not directly support control over local sensors. However, for simple notions of control, such as activating or triggering a sensor, the event model can be used. Assuming that the value of sensor is only needed when at least one event listener is registered to the respective DCCI property, the property can (de)activate or trigger the sensor accordingly.

The drawback with DCCI is that it is meant to be a consumer interface and therefore lacks a standardized provider interface. Moreover, DCCI is meant to be used within a wider framework with security controls and object management that is not addressed within the current specification.

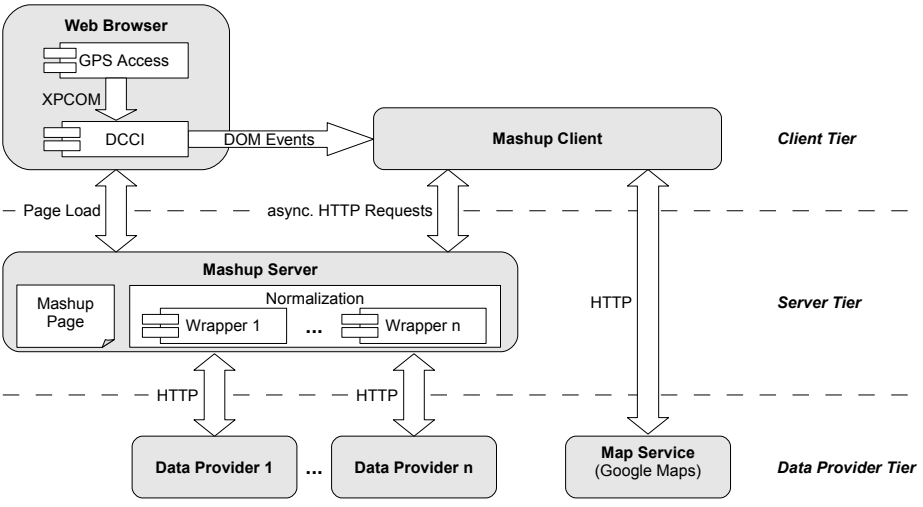


Fig. 1. Proposed system architecture for context-aware mashups

3.2 Data Integration

Our solution to integrate the data from various different data providers is a simple wrapper approach. Small and independent wrapper scripts impose an abstraction layer on the data providers creating a consistent RESTful interface to retrieve the data. The wrappers query the data providers and convert the data into a single well-known format. As our scenario focuses on POIs, a common data model is easy to find. The fact that the wrappers need to be programmed makes user-programming difficult, on the other hand virtually any data source can be accessed. However, given a sufficient amount of wrappers, one can choose which data providers to include in a mashup. In addition, wrappers can be parametrized giving the user or the mashup creator some control.

3.3 Architectural Overview

A graphical overview on our proposed system architecture is given in Figure 1. As with a typical AJAX-based mashup, there are three tiers: A mashup is viewed in the client tier. The web browser loads the mashup page and starts the JavaScript code of the mashup client AJAX application. The mashup page is loaded from the mashup server, which resides on the Internet and constitutes the server tier. Data offered by third-party data providers is used, which are distributed throughout the Internet. The map is loaded from a map service, which, together with the data providers, makes up the data provider tier. Note that the data provider tier is outside of the organizational boundaries of the mashup.

A mashup consists of an HTML page importing the JavaScript files of the mashup client. The mashup client is only deployed to the mashup server and used as-is. It needs to be configured (most notably, the data providers to use

and the initial position of the map), but not programmed. The mashup client asynchronously reads the configuration when the mashup page is loaded. The mashup client then constructs the user interface. It displays a map and visualizes POI data from the data providers. In order to cope with the heterogeneity of data formats and interfaces used by the different data providers, a normalization layer is required. As mentioned in Section 3.2, we used programmed wrappers to achieve a consistent interface for data retrieval, as this approach offers the highest degree of flexibility. For other scenarios with less diversity, other, possibly automatic normalization approaches are applicable (e. g., data providers solely providing RSS feeds, as used by Yahoo! pipes).

Context information, such as the user's location, is integrated into the mashup by extending the web browser. There are two additional components: the DCCI module and the GPS access module. The DCCI module implements the DCCI specification [10] and constitutes the interface for providing context data to web pages. The mashup client registers itself as an event listener to the DCCI module and is notified whenever the user's location changes. The GPS access module connects to the GPS device and ships the location information to the DCCI module. Dividing the context provisioning framework into a client interface and a provisioning module allows further context provisioning modules to be added later on.

The data flow works as follows: Whenever the GPS access module obtains a new location from the GPS device, the location information is updated in the DCCI module. The mashup client, which is registered as an event listener to the DCCI module, is notified about the change via DOM events. Subsequently, the mashup client updates the user's location on the map and centers the map to the new location. If the area shown on the map has significantly changed, the mashup client sends asynchronous HTTP requests to the wrappers, in order to obtain POI data for the new map area. The wrappers translate these requests into calls to the particular APIs of the data providers and convert the resulting data into a unique data format understood by the mashup client. Finally, the mashup client reads the reply sent by the wrappers and visualizes the POI data on the map.

3.4 Mashup Application Development

In order to create a mashup using our proposed architecture, the following steps need to be taken:

1. Select the data providers. Create the respective wrappers, if not yet available.
2. Write an HTML page, into which the map presentation should be embedded.
3. Deploy the mashup client, the wrappers, and the HTML page to a web server.
4. Configure the mashup client defining the initial map area (center point and zoom level) and the data provider wrappers to use.

No AJAX programming is required, as the mashup client handles all map interaction, displays the POIs and integrates the user's location.

Fig. 2. Screenshot of a TELAR mashup on a Nokia N810

4 The Telar Prototype

4.1 Implementation

We put our proposed architecture into practice on a Nokia N810 Internet Tablet, as depicted in the screenshot in Figure 2. The Mozilla-based web browser of the Nokia N810 provides a powerful extension mechanism, using which the DCCI module and the GPS access module were implemented in C++. The two modules communicate directly via XPCOM, the component framework of the Mozilla browser. As DCCI only defines a client interface, the DCCI module had to define its own provider interface as the back end. Although we did not aim at implementing a full-fledged provider framework, as proposed in [17], this provider interface is generic and can be used for different kinds of context data providers. Based on the provider interface, further browser extensions can be written, which expose context data via one or more DCCI properties.

To implement the mashup client we used the Google Web Toolkit[7]. This gave us the possibility to write the non-trivial mashup client in Java utilizing professional development tools. Moreover, the menus and dialogs making up the user interface of the mashup client could be easily created with the widget library of the Google Web Toolkit.

The wrappers were implemented in PHP 5 using XSL stylesheets to convert from XML-based data formats. The wrappers are significantly shorter than 100 lines of code and their implementation was very straight forward. Thus, additional data providers can be added to the TELAR mashup platform without problems and changes in the interface of data providers can be easily resolved.

[7] `http://code.google.com/webtoolkit`

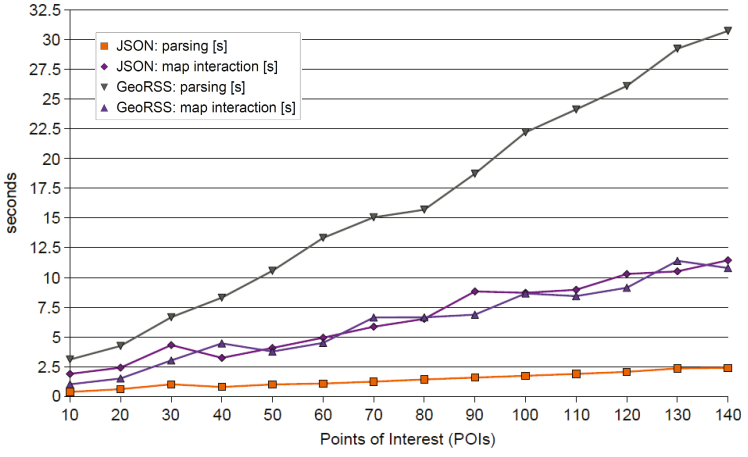


Fig. 3. Time for processing POIs in GeoRSS and JSON format

4.2 Performance Optimizations

When the mashup client and some wrappers were implemented, first tests showed that performance was insufficient. Working fine on a state-of-the-art desktop PC, it could take minutes until a mashup was completely constructed on a Nokia Internet Tablet. A similar experience was made on a Pentium II PC. A first analysis revealed that most of the time was spent for parsing the POI data which the mashup client retrieved from the wrappers. It took additional time to draw the POIs on the map. Both steps are done by JavaScript code interpreted in the browser. In contrast to a desktop PC, the processor of the Nokia Internet Tablet was simply not powerful enough to do this job quickly.

Our first version used an extended version of GeoRSS [18] as serialization format for the POIs. GeoRSS is a standardized, simple, and popular format based on Atom or RSS for describing geographically annotated objects. By using GeoRSS, existing tools and web pages supporting GeoRSS could be used for testing the wrappers and the wrappers could even be reused for other applications.

In order to improve performance, the standardized GeoRSS format was replaced by a proprietary JSON format [19]. As JSON is a subset of JavaScript, it can be parsed very efficiently using the `eval()` JavaScript function, which the web browser implements in native code. Figure 3 shows the performance measurements done with GeoRSS and JSON. For both formats two values were measured on a Nokia Internet Tablet with various amounts of data: the time for unmarshalling the data into objects and the time for drawing these objects on the map. The readings show that for GeoRSS parsing takes much longer than drawing the POIs. Parsing 100 POIs of GeoRSS data takes more than 20 seconds, whereas parsing 100 POIs from JSON takes less than three seconds. As the POIs are in any case unmarshalled to objects first, the time to draw the POIs on the map is independent of the serialization format. Drawing 100 POIs

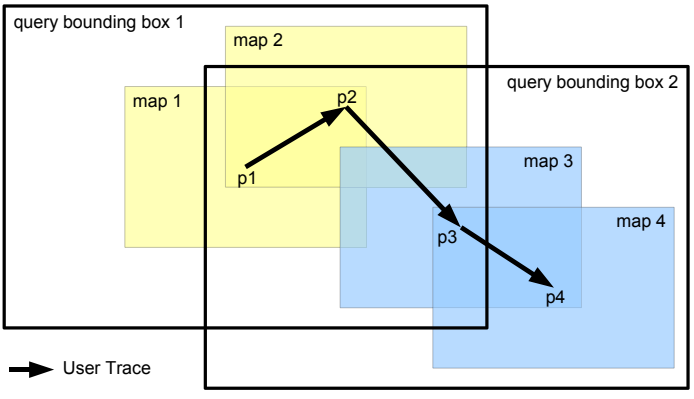


Fig. 4. Query bounding boxes for reducing the number of queries

takes between eight and ten seconds. Thus, despite of using JSON to serialize the POIs, adding the POIs to the map still takes too long.

As adding POIs to the map is expensive for devices with limited resources, unnecessary calls to the map API should be avoided. In the scenario of a walking user, the map changes every time the GPS access module updates the location data in the DCCI tree, e. g., in intervals of a few seconds, but only to a small degree. It would be sufficient to query POIs for the area which was added to the map with the last location update. However, this is not easily feasible, as the new area is generally not rectangular. As most data providers only support rectangular query areas, each query would have to be split in two separate queries, doubling latency.

Instead, we introduced the approach of an *extended query bounding box*. Whenever the mashup client needs new POI data, 25 % of padding is added to the map bounding box. Then, POI data is queried for the resulting query bounding box. Whenever the map bounding box changes, the mashup client first tests, whether the new map bounding box is still within the last query bounding box. Only if this is not the case, a new query bounding box is calculated and new data is queried.

Figure 4 illustrates the query bounding box. To display POIs for map 1, query bounding box 1 is calculated and POIs are retrieved accordingly. As the user moves from position p1 to p2, map 2 is displayed. But as the map bounding box of map 2 is within query bounding box 1, no new data is required. Only when the user walks to position p3, the map bounding box of map 3 is not entirely within query bounding box 1 and new data must be queried. For map 4, the POI data fetched for query bounding box 2 is available again.

As the area for which the data providers are queried is a lot larger in the approach of the query bounding box, a larger number of POIs is retrieved and the initial loading time of the mashup even increases. But once the mashup is loaded, significantly fewer queries are needed. In addition to that, the approach of the query bounding box can cope with changes of the user's walking direction

very well. The query bounding box is extended in all directions, so if the user changes his direction, the POIs will be still available for a while.

5 Conclusions

In this paper, we examined user-centric context-aware mashups for mobile devices on a sample scenario of location-based mashup integrating points of interest (POIs) from arbitrary data providers. We formulated a list of requirements for a context provisioning framework for context-aware mashups and found the W3C Delivery Context Client Interfaces (DCCI) to fulfill these requirements best. We addressed the challenge of integrating POI data from arbitrary heterogeneous data providers. For our scenario, we chose an approach based on manually programmed wrappers to abstract from the various data formats and interfaces, trading flexibility with the possibility of user programming. Proceeding from these fundamental design decisions, we presented a system architecture for context-based mashups making wide use of AJAX. Based on this architecture, we implemented the Telar mashup platform, which puts our scenario into practice on a Nokia N810 Internet Tablet.

We shared our experience with rich web applications making intensive use of AJAX on a mobile device with limited resources: the performance can be sufficient for small applications but is not yet fully satisfying. By switching from GeoRSS to JSON as the serialization format for the POIs we could improve performance significantly. The same holds for a caching strategy based on extended query bounding boxes. Despite these measures, the mashup is still far from comparable to a native application or from viewing the mashup in a PC browser. This is mainly due to the fact that AJAX applications, such as the mashup client, use the web browser in a way for which it was not originally designed, which leads to worse performance.

A main future challenge is implementing privacy in context provisioning frameworks like DCCI. For the purposes of the Telar mashup platform, a simple mechanism similar to popup blockers would be sufficient. However in order to widely equip web browsers with context frameworks, a general solution is required. Further potential for future work lies in the sustainability of mashups, so that even in the case of data providers failing or changing their interfaces mashups can continue to provide user value.

References

1. Dey, A.K.: Understanding and using context. Personal and Ubiquitous Computing 5, 4–7 (2001)
2. Brodt, A., Nicklas, D.: The TELAR mobile mashup platform for Nokia internet tablets. In: Advances in Database Technology - EDBT 2008, 11th International Conference on Extending Database Technology, Munich, Germany, Proceedings (to appear, 2008)
3. Salber, D., Dey, A.K., Abowd, G.D.: The context toolkit: Aiding the development of context-enabled applications. In: CHI, pp. 434–441 (1999)

4. Henricksen, K., Indulska, J.: A software engineering framework for context-aware pervasive computing. In: PerCom, pp. 77–86. IEEE Computer Society, Los Alamitos (2004)
5. Mitschang, B., Nicklas, D., Großmann, M., Schwarz, T., Hönle, N.: Federating location-based data services. In: Data Management in a Connected World (2005)
6. Daviel, A., Kaegi, F.A., Kofahl, M.: Geographic extensions for http transactions. Internet draft, The Internet Society (2007)
7. Klyne, G., Reynolds, F., Woodrow, C., Ohto, H., Hjelm, J., Butler, M.H., Tran, L.: Composite capability/preference profiles (cc/pp): Structure and vocabularies 1.0. Recommendation. In: W3C (January 2004)
8. OMA Device Capability working group: Uaprof. Specification, Open Mobile Alliance (2001)
9. OMA Device Capability working group: Device profile evolution architecture. In: Draft, Open Mobile Alliance (2007)
10. Waters, K., Hosn, R.A., Raggett, D., Sathish, S., Womer, M., Froumentin, M., Lewis, R.: Delivery context: Client interfaces (dcci) 1.0. Candidate recommendation. In: W3C (December 2007)
11. Floyd, I.R., Jones, M.C., Rathi, D., Twidale, M.B.: Web mash-ups and patchwork prototyping: User-driven technological innovation with Web 2.0 and Open Source software. In: HICSS. IEEE Computer Society, Los Alamitos (2007)
12. Wong, J., Hong, J.I.: Making mashups with marmite: towards end-user programming for the web. In: Rosson, M.B., Gilmore, D.J. (eds.) CHI. ACM, New York (2007)
13. Jhingran, A.: Enterprise information mashups: Integrating information, simply. In: Dayal, U., Whang, K.Y., Lomet, D.B., Alonso, G., Lohman, G.M., Kersten, M.L., Cha, S.K., Kim, Y.K. (eds.) VLDB. ACM, New York (2006)
14. Wilde, E.: Knowledge organization mashups. TIK Report 245, ETH Zürich (Swiss Federal Institute of Technology) (2006), `http://dret.net/netdret/publications#wil06f`
15. Murthy, S., Maier, D., Delcambre, L.M.L.: Mash-o-matic. In: Bulterman, D.C.A., Brailsford, D.F. (eds.) ACM Symposium on Document Engineering. ACM, New York (2006)
16. Ennals, R., Gay, D.: User-friendly functional programming for web mashups. In: ICFP 2007: Proceedings of the 2007 ACM SIGPLAN international conference on Functional programming, pp. 223–234. ACM, New York (2007)
17. Sathish, S., Pettay, O.: Delivery context access for mobile browsing. In: ICCGI 2006: Proceedings of the International Multi-Conference on Computing in the Global Information Technology, Washington, DC, USA, p. 18. IEEE Computer Society, Los Alamitos (2006)
18. Reed, C., Singh, R., Lake, R., Lieberman, J., Maron, M.: An introduction to georss: A standards based approach for geo-enabling rss feeds. White Paper OGC 06-050r3, Open Geospatial Consortium Inc. (2006)
19. Crockford, D.: The application/json media type for javascript object notation (json). Request for Comments 4627, The Internet Society (2006)

A Semantic Overlay for Service Discovery across Web Information Systems*

Devis Bianchini, Valeria De Antonellis, Michele Melchiori, and Denise Salvi

Università di Brescia
Dip. Elettronica per l'Automazione
Via Branze, 38
25123 Brescia - Italy
{bianchin,deantone,melchior}salvi@ing.unibs.it

Abstract. Many collaborative organizations require advanced semantic interoperability tools to enable cooperation and communication across distributed Web Information Systems (WIS). Adopting the service-oriented technology, they have improved interoperability at the application level by exporting WIS functionalities as Web services. Furthermore, in order to support effective peer-to-peer collaboration, they require semantic interoperability techniques for service discovery and sharing. In this paper, we focus on semantic interoperability issues for distributed collaboration and provide techniques for building a service semantic overlay, across heterogeneous WIS. In particular, semantic links among peers that offer comparable services in a given domain are defined and maintained over the time. The service semantic overlay is ontology-based for formally organizing the shared services to enhance the capability of service interchange and interoperation among the collaborative systems. The approach and a preliminary experimentation have been proposed to demonstrate practical benefits in the framework of the ESTEEM (Emergent Semantics and cooperaTion in multi-knowledgE EnvironMents) project.

1 Introduction

Nowadays many collaborative organizations require advanced semantic interoperability tools to enable cooperation and communication across distributed Web Information Systems (WIS). Methods and tools devoted to the interconnection of the systems according to a peer-to-peer approach have enabled collaboration at the technological level. Adoption of service-oriented technology has improved interoperability at the application level by exporting WIS functionalities as Web services. In order to support effective peer-to-peer collaboration, semantic interoperability techniques for P2P service discovery are required [1,11].

* This work has been partially supported by the ESTEEM PRIN Project (http://www.dis.uniroma1.it/∼esteem/index.html) and TEKNE FIRB Project (http://www.tekne-project.it/), funded by the Italian Ministry of Education, University and Research.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 292–306, 2008.

For service analysis and automated matchmaking, we have already proposed a methodology [9] and the **S**emantic **D**riven **S**ervice **D**iscovery (SDSD) approach characterized by a novel hybrid matchmaking and discovery environment [7,8]. In this paper, we further extend the approach by proposing a service semantic overlay built over the P2P logical network, where semantic links among services belonging to different peers are maintained and exploited to optimize service discovery according to different policies. Semantic links between two service specifications are computed with respect to the service interfaces, described in terms of the entities exchanged with the service during its execution (input and output parameters) and the operations performed on these entities. Two services are similar with respect to their functionality if they perform the same or similar operations on the same or similar entities. Similarity is evaluated by applying ontology-based techniques. We consider semantic links between services belonging either to a single peer (*intra-peer semantic links*) or to different peers (*inter-peer semantic links*); in this way it is possible to identify possible synergic service centres over the network. Each peer maintains semantic links of interest (preferential semantic links, selected according to threshold based criteria) together with local service descriptions. When a peer receives a service request, semantic links are exploited to optimize both local search within the peer and distributed search of possible candidate services over the P2P network. Furthermore, we propose a preliminary experimental evaluation that shows how semantic link exploitation allows for effective and efficient service discovery without heavily affecting the P2P network workload. The approach described in this paper is part of the ESTEEM (Emergent Semantics and cooperaTion in multi-knowledgE EnvironMents) approach [16], where a comprehensive framework and platform for data and service discovery in P2P systems are proposed, with advanced solutions for trust and quality-based data management, P2P infrastructure definition, query processing and dynamic service discovery in a context-aware scenario.

The paper is organized as follows: Section 2 introduces the considered application scenario; Section 3 describes the ontology-based framework for service semantic overlay management; Section 4 illustrates the service discovery process based on the semantic overlay; Section 5 presents preliminary experimental results; Section 6 discusses related work and points out some differences with our approach; finally, Section 7 presents concluding remarks and future work.

2 Application Scenario

Consider a P2P collaboration among organizations using their WIS in a given application domain, aiming at providing and retrieving services (WIS-NET). In our vision, each service has an abstract description (abstract service) in terms of service functionalities (operations) and input/output messages (parameters), based on the WSDL standard. For each abstract service, a set of concrete implementations is provided on the network. Two distinct peers could register the same abstract service, for which different implementations could be provided, or

they could advertise different abstract services that partially overlap. When a peer is looking for a service, it formulates the service request by specifying the expected functional interface and sends the request to one of the peers of the network. According to this vision and to the service-oriented paradigm, each peer can play three different roles: *(i)* it can store abstract services and references to corresponding concrete implementations (*broker*); *(ii)* it can publish on a broker the implementation of a service, described by its functional interface (*provider*); *(iii)* it can look for a service (*requester*). When a peer joins the network, it can act both as a broker/provider or a requester.

When a peer P_X joins the WIS-NET site, it obtains the list of brokers' IP currently available in the network and starts to contact them. When a broker P_Y replies, P_X can be connected to it. If P_X is a provider, it publishes its services on broker P_Y. If P_X is a requester, it sends a service request to the broker P_Y and waits for the answer. Finally, if P_X is a broker, it receives from P_Y a list of brokers known by P_Y. The proposed application scenario is designed according to the Service-Oriented Architecture (SOA). Brokers receive service requests and identify suitable providers, that are sent to the requesters. When receiving the URLs of providers whose services match their requirements, requesters directly contact service providers to start service invocation.

Brokers are in charge of establishing and maintaining semantic links towards other brokers on the basis of similarity between abstract services they store. Semantic links constitute a *semantic overlay* over the network of collaborative peers. In the rest of the paper, we will explain how the semantic overlay is built, maintained and exploited for service discovery purposes.

3 Ontology-Based Model for the Service Semantic Overlay

Broker service registry. Each broker in the network has its local: *(i)* UDDI Registry, where services are registered with their URL and associated, through `tModels`, to abstract services that represent their functional interface, described using WSDL language; *(ii) peer ontology*, that provides a conceptualization of abstract service operations (e.g., flight reservation) and I/O parameters (e.g., electronic ticket) through concepts and semantic relationships between them; *(iii)* standard service categorization, denoted with *Service Category Taxonomy* (SCT), extracted from available standard taxonomies (e.g., UNSPSC, NAiCS) to classify abstract services. The peer ontology is represented using OWL-DL formalism.

The peer ontology is augmented by a thesaurus containing terms that are related by terminological relationships (as synonymy or hypernymy) to the names of ontological concepts. By means of the thesaurus, matching capabilities based on the peer ontology are extended. Details about the combined use of peer ontologies and thesaurus can be found in [7].

Broker service matchmaker. Each broker is endowed with a matchmaker. In [7] we defined a hybrid matchmaking strategy, where a deductive matchmaking model is combined with a similarity-based model to automatically compare

services on the basis of their functional interface. The deductive matchmaking model is applied to *qualify* the kind of match `MatchType` ($\mathcal{S}_1$,$\mathcal{S}_2$) between two abstract services $\mathcal{S}_1$ and $\mathcal{S}_2$. According to this matchmaking model, it is possible to state if $\mathcal{S}_1$ and $\mathcal{S}_2$ provide the same functionalities ($\mathcal{S}_1$ EXACT $\mathcal{S}_2$), if $\mathcal{S}_1$ provides additional functionalities with respect to $\mathcal{S}_2$ ($\mathcal{S}_1$ EXTENDS $\mathcal{S}_2$) or viceversa, if there is a non empty intersection between functionalities provided by $\mathcal{S}_1$ and $\mathcal{S}_2$ ($\mathcal{S}_1$ INTERSECTS $\mathcal{S}_2$) or if $\mathcal{S}_1$ and $\mathcal{S}_2$ have nothing in common ($\mathcal{S}_1$ MISMATCH $\mathcal{S}_2$). The deductive matchmaker first looks for in the peer ontology to find semantic relationships between operations in $\mathcal{S}_1$ and operations in $\mathcal{S}_2$. Then, the same procedure is applied for output and input parameters. If all operations (respectively, inputs and outputs) in $\mathcal{S}_1$ have equivalent or more specific operations (respectively, inputs and outputs) in $\mathcal{S}_2$ and there are no additional operations (respectively, inputs and outputs) in $\mathcal{S}_2$, then $\mathcal{S}_1$ EXACT $\mathcal{S}_2$; if at least the operations (respectively, inputs and outputs) in $\mathcal{S}_1$ have equivalent or more general operations (respectively, inputs and outputs) in $\mathcal{S}_2$ or $\mathcal{S}_2$ provides additional operations/outputs to $\mathcal{S}_1$, then $\mathcal{S}_2$ EXTENDS $\mathcal{S}_1$; if there are no operations (respectively, inputs and outputs) in $\mathcal{S}_1$ and $\mathcal{S}_2$ that are semantically related in the peer ontology, then $\mathcal{S}_1$ MISMATCH $\mathcal{S}_2$; otherwise, $\mathcal{S}_1$ INTERSECTS $\mathcal{S}_2$.

In case of partial overlapping (EXTENDS or INTERSECTS), the similarity-based matchmaking model is used to *quantify* service similarity $Sim(\mathcal{S}_1, \mathcal{S}_2) \in [0, 1]$ through coefficients properly defined to compare service interfaces. These coefficients evaluate similarity distance between service interfaces by exploiting terminological relationships in the thesaurus. On the other hand, if $\mathcal{S}_1$ EXACT $\mathcal{S}_2$ or $\mathcal{S}_1$ MISMATCH $\mathcal{S}_2$, $Sim(\mathcal{S}_1, \mathcal{S}_2) = 1.0$ or $Sim(\mathcal{S}_1, \mathcal{S}_2) = 0.0$, respectively. Two service interfaces $\mathcal{S}_1$ and $\mathcal{S}_2$ are *similar* (denoted with $\mathcal{S}_1 \approx \mathcal{S}_2$) if `MatchType`($\mathcal{S}_1$,$\mathcal{S}_2$) is not MISMATCH and $Sim(\mathcal{S}_1, \mathcal{S}_2) \geq \delta$, where δ is a similarity threshold. A detailed presentation of the hybrid matchmaking model with experimental results about the matchmaker is given in [7], while in [9] more details about similarity coefficients are provided. In this paper, we show how to exploit the service matchmaking models to improve a semantic-driven search of services among collaborative peers.

Example 1. Consider a broker where e-healthcare services (e.g., remote diagnosis, reservation of laboratory tests and hospitalizations) are published. A portion of the reference peer ontology is shown in Figure 1, together with a portion of the thesaurus. In the example, a request $\mathcal{S}_R$ is compared with an offer $\mathcal{S}$. An EXTENDS match between $\mathcal{S}_R$ and $\mathcal{S}$ is found ($\mathcal{S}$ EXTENDS $\mathcal{S}_R$), since: (i) the input parameters are equivalent; (ii) the required output `Disease` has a corresponding output `PathologicDysfunction` in $\mathcal{S}$ that is more general according to the peer ontology; (iii) $\mathcal{S}$ provides an additional output `Treatment`. To check the correspondence between `Diagnosis` and `DiagnosticTest` is more complex, since `DiagnosticTest` is not present as concept in the peer ontology. This could be a very common situation in a P2P environment, where different reference ontologies could be adopted by distinct nodes on the network. To go beyond this limitation, the thesaurus is exploited: in this example, `Diagnosis` and `DiagnosticTest` are synonyms in the thesaurus and are then considered as

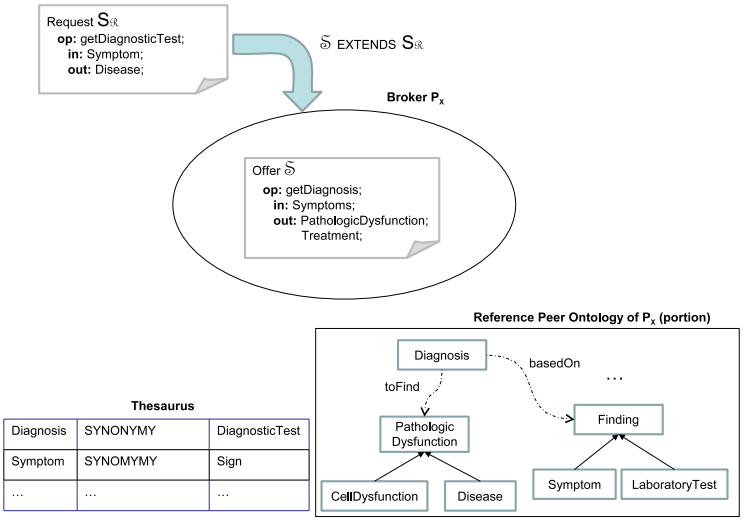


Fig. 1. Example of ontology-based service matchmaking on a broker

matching concepts. Similarity $Sim(\mathcal{S}_R, \mathcal{S})$ in this case is high, since $\mathcal{S}_R$ and $\mathcal{S}$ present many equivalent elements in their interfaces. Then, $\mathcal{S}_R$ and $\mathcal{S}$ are considered as similar ($\mathcal{S}_R \approx \mathcal{S}$). A detailed explanation of how similarity coefficients are calculated is out of the scope of this paper.

Service semantic overlay. Matchmaking information (`MatchType` and Sim) are used to define semantic links among abstract services when advertised on brokers. Semantic links constitute the service semantic overlay. Figure 2 shows an example of service semantic overlay built on the logical network overlay as viewed on the broker P_X. A semantic link between two services $\mathcal{S}_1$ and $\mathcal{S}_2$ is established if they are similar. If $\mathcal{S}_1$ and $\mathcal{S}_2$ are published on the same broker P and $\mathcal{S}_1 \approx \mathcal{S}_2$, then an *intra-peer semantic link* is established between them, denoted with $sl_P(\mathcal{S}_1, \mathcal{S}_2)$. Semantic links are labeled with the similarity degree and the match type.

Example 2. For example, in Figure 2 on broker P_X, abstract service $\mathcal{S}1_X$ (to obtain remote diagnosis together with treatment details and suggested laboratory tests) is connected to $\mathcal{S}3_X$. In particular, $\mathcal{S}1_X$ adds treatment and laboratory test details with respect to $\mathcal{S}3_X$ ($\mathcal{S}1_X$ EXTENDS $\mathcal{S}3_X$). No match is found between $\mathcal{S}3_X$ (or $\mathcal{S}1_X$) and $\mathcal{S}2_X$.

If $\mathcal{S}_1$ and $\mathcal{S}_2$ are published on two different brokers P_1 and P_2 and $\mathcal{S}_1 \approx \mathcal{S}_2$, an *inter-peer semantic link* is established between them, denoted with $isl_{P_1 \rightarrow P_2}(\mathcal{S}_1, \mathcal{S}_2)$. To establish inter-peer semantic links, the broker P_X sends a probe service request for each service $\mathcal{S}i_X$ to be shared with other brokers connected at the logical network overlay. The probe service request contains the interface of $\mathcal{S}i_X$ and the IP address of P_X. The broker P_Y that receives the probe service request, matches it against

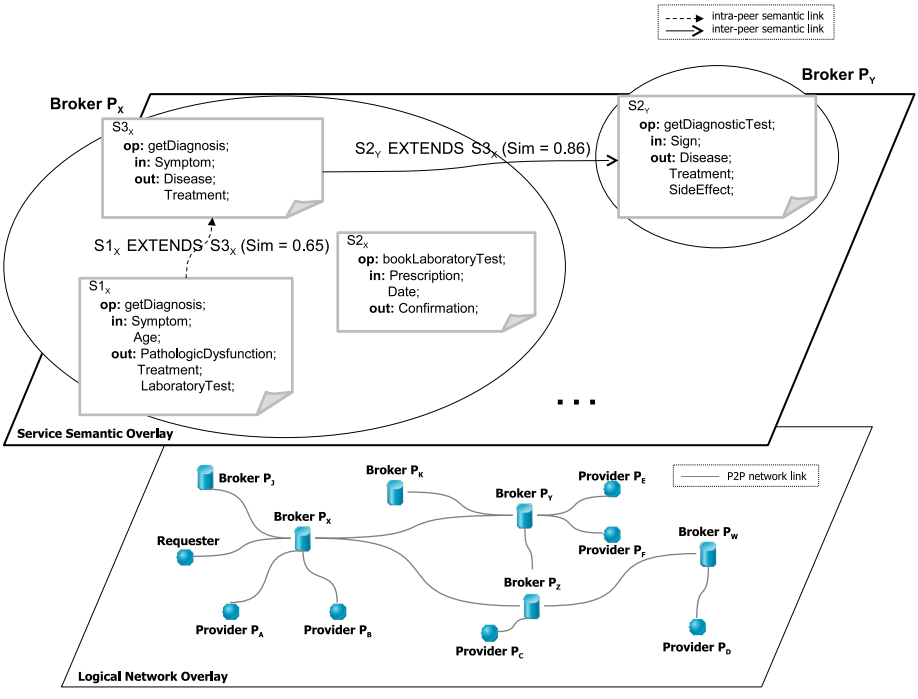


Fig. 2. A portion of service semantic overlay

its own abstract services $\mathcal{S}j_Y$ by applying the matchmaking techniques based on its own peer ontology and obtains for each comparison the `MatchType` and the similarity value. P_Y then replies to P_X with the list of $\mathcal{S}j_Y$ such that $\mathcal{S}i_X \approx \mathcal{S}j_Y$, together with the `MatchType` and the similarity value. P_X establishes an inter-peer semantic link $isl_{P_X \rightarrow P_Y}(\mathcal{S}i_X, \mathcal{S}j_Y)$ for each $\mathcal{S}j_Y$ and P_Y is recognized as a *semantic neighbor* of P_X.

Example 3. For example, in Figure 2, service $\mathcal{S}2_Y$ on broker P_Y adds functionalities with respect to service $\mathcal{S}3_X$ on broker P_X. An inter-peer semantic link is set from $\mathcal{S}3_X$ on P_X to $\mathcal{S}2_Y$ on P_Y.

Logical network overlay evolution. In the considered collaborative scenario there is the need to guarantee the communication among peers and manage the dynamic aspects featuring the P2P network organization. The logical network overlay collects all the peers participating to the collaborative system: each node represents a peer and each link is a logical connection between two peers. In order to guarantee connection between peers an Overlay Management Protocol (OMP) is used, which defines specific procedures to join, leave and modify the logical network overlay. In our approach a shuffling-based OMP is chosen in order to allow more effective information diffusion among peers [18]. This kind of OMP arranges the logical network overlay as a graph in which each peer is directly

connected to a small portion of the entire peer population, that is, their logical neighbors. The shuffling protocol is quite simple: each peer continuously changes the set of its logical neighbors by occasionally contacting a random neighbor, then they exchange some of their neighbors. The obtained result is an inexpensive overlay membership management, in the sense that any joining or leaving of peers is quickly and efficiently managed without overloading the network.

Service semantic overlay evolution. The service semantic overlay has to be managed with respect to changes that occur to the participating brokers: acquisition of the IP address of a new broker in the logical network overlay, disconnection of a semantic neighbor from WIS-NET, publication/cancellation of a service in the broker registry.

When the IP of a new broker P_Y is acquired by broker P_X, due to the shuffling-based protocol at the logical network overlay, P_X sends a probe service request to P_Y and possibly inter-peer semantic links between P_X and P_Y are established based on the received answers. Probe service requests are sent according to a Time-To-Live (TTL) mechanism with a low TTL value to avoid network overload. Brokers that cannot be reached from P_X, due to the low TTL value, will be reached thanks to the shuffling-based protocol. Experimentation is being performed to establish the best value of TTL.

A broker P_X recognizes that a semantic neighbor P_Y becomes unavailable if a given number of messages sent to P_Y are not answered. In this case the inter-peer semantic links toward abstract services published on P_Y are removed from P_X.

Finally, if a new service $\mathcal{S}i_X$ is published on a broker P_X, a probe service request is sent to semantic neighbors of P_X and inter-peer semantic links are established on the basis of obtained answers. If a service is removed, the inter-peer semantic links based on it are removed on the broker on which it has been published.

4 Service Discovery Based on the Service Semantic Overlay

A service request $\mathcal{S}_R$ is formulated by specifying one or more service categories and the desired functional interface. It is sent to a broker P_X, to which the requester is connected when joining WIS-NET, and it is matched against abstract services published on P_X (*local search*). Afterwards, $\mathcal{S}_R$ is sent to the other brokers that are semantic neighbors of P_X, according to different forwarding policies (*distributed search*). Each broker, which receives the request, applies the same matching process. Each broker collects its local concrete implementations for the abstract services matching $\mathcal{S}_R$ with those received from semantic neighbors and sends them back to the broker from which the request came, up to the requester.

Local search. When the service request $\mathcal{S}_R$ reaches the broker P_X, the broker searches for abstract services in its own registry classified in at least one of the categories specified for $\mathcal{S}_R$ and applies the hybrid matchmaking model to build a

list $MS(\mathcal{S}_R) = \{\langle \mathcal{S}1_X, Sim_1, mt_1\rangle, \ldots \langle \mathcal{S}n_X, Sim_n, mt_n\rangle\}$ of matching services such that $\mathcal{S}_R \approx \mathcal{S}i_X$, where $Sim_i = Sim(\mathcal{S}_R, \mathcal{S}i_X)$ and $mt_i = \texttt{MatchType}(\mathcal{S}_R, \mathcal{S}i_X)$.

Intra-peer semantic links on broker P_X can be exploited to efficiently find all available services which match locally. In fact, if $\mathcal{S}i_X$ is found such that $\mathcal{S}_R \approx \mathcal{S}i_X$, then $\mathcal{S}i_X$ is included into $MS(\mathcal{S}_R)$ and only abstract services $\mathcal{S}j_X$ related with intra-peer semantic links to $\mathcal{S}i_X$ are considered as candidate services for the request $\mathcal{S}_R$, according to the rules presented in Table 1.

After performing local search, broker P_X forwards the request to its semantic neighbors with respect to $\mathcal{S}i_X \in MS(\mathcal{S}_R)$. If no matching service has been found locally, P_X selects randomly a set of brokers connected at the logical network overlay and forwards $\mathcal{S}_R$ to them.

Table 1. Exploitation of intra-peer semantic links during local search

$\texttt{matchType}(\mathcal{S}_R, \mathcal{S}i_X)$	$\texttt{matchType}(\mathcal{S}i_X, \mathcal{S}j_X) \neq$ MISMATCH	$\texttt{matchType}(\mathcal{S}_R, \mathcal{S}j_X)$
R1) $\mathcal{S}i_X$ EXACT $\mathcal{S}_R$	$\mathcal{S}i_X$ <*matchType*> $\mathcal{S}j_X$	$\Rightarrow \mathcal{S}j_X$ <*matchType*> $\mathcal{S}_R$
R2a) $\mathcal{S}i_X$ EXTENDS $\mathcal{S}_R$	$\mathcal{S}j_X$ EXACT|EXTENDS $\mathcal{S}i_X$	$\Rightarrow \mathcal{S}j_X$ EXTENDS $\mathcal{S}_R$
R2b)	$\mathcal{S}j_X$ INTERSECTS $\mathcal{S}i_X$	evaluate $\texttt{MatchType}(\mathcal{S}j_X, \mathcal{S}_R)$
R2c)	$\mathcal{S}i_X$ EXTENDS $\mathcal{S}j_X$	evaluate $\texttt{MatchType}(\mathcal{S}j_X, \mathcal{S}_R)$
R3a) $\mathcal{S}_R$ EXTENDS $\mathcal{S}i_X$	$\mathcal{S}j_X$ EXACT $\mathcal{S}i_X$	$\Rightarrow \mathcal{S}_R$ EXTENDS $\mathcal{S}j_X$
R3b)	$\mathcal{S}i_X$ EXTENDS $\mathcal{S}j_X$	$\Rightarrow \mathcal{S}_R$ EXTENDS $\mathcal{S}j_X$
R3c)	otherwise	evaluate $\texttt{MatchType}(\mathcal{S}j_X, \mathcal{S}_R)$
R4a) $\mathcal{S}i_X$ INTERSECTS $\mathcal{S}_R$	$\mathcal{S}j_X$ EXACT $\mathcal{S}i_X$	$\mathcal{S}j_X$ INTERSECTS $\mathcal{S}_R$
R4b)	otherwise	evaluate $\texttt{MatchType}(\mathcal{S}j_X, \mathcal{S}_R)$

Example 4. In Figure 3, since $\mathcal{S}3_X$ EXTENDS $\mathcal{S}_R$, then also $\mathcal{S}1_X$ EXTENDS $\mathcal{S}_R$ according to rule R2a on Table 1. The kind of match between $\mathcal{S}_R$ and $\mathcal{S}1_X$ is not evaluated, while $\mathcal{S}2_X$ is not considered for $\mathcal{S}_R$, since no intra-peer semantic links exist between $\mathcal{S}2_X$ and the other abstract services ($\mathcal{S}3_X$ and $\mathcal{S}1_X$), that are relevant for $\mathcal{S}_R$.

Distributed search. Once matching abstract services have been found on peer P_X, two forwarding policies based on inter-peer semantic links can be applied. Each broker which receives $\mathcal{S}_R$ from a semantic neighbor applies one of the two forwarding policies (minimal/exhaustive); the search stops according to a Time To Live mechanism.

Minimal policy. Search over the semantic overlay stops when matching services which fully satisfy the request have been found; this strategy is performed according to the following rules:

1) if $\exists \mathcal{S}i_X \in MS(\mathcal{S}_R)$ such that $\mathcal{S}i_X$ EXACT | EXTENDS $\mathcal{S}_R$, it is not necessary to forward the request to semantic neighbors, since concrete implementations of $\mathcal{S}i_X$ already satisfy completely the request; otherwise

2) for each $\mathcal{S}i_X \in MS(\mathcal{S}_R)$ such that $\mathcal{S}_R$ INTERSECTS | EXTENDS $\mathcal{S}i_X$, the request is not completely satisfied by $\mathcal{S}i_X$ and is forwarded to semantic neighbors, that could add further functionalities to those already provided by $\mathcal{S}i_X$; however, broker P_X does not consider semantic neighbors that provide services

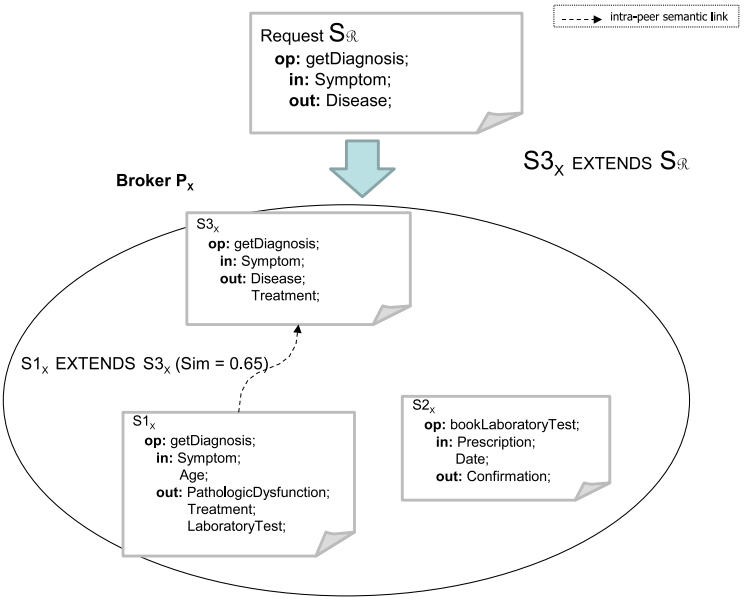


Fig. 3. An example of exploitation of intra-peer semantic links during local search

$\mathcal{S}j_Y$ such that $\mathcal{S}i_X$ EXTENDS | EXACT $\mathcal{S}j_Y$, because this means that $\mathcal{S}j_Y$ does not provide additional functionalities with respect to those already provided by $\mathcal{S}i_X$;

3) if no semantic neighbors exist for any abstract service $\mathcal{S}i_X \in MS(\mathcal{S}_R)$, the request is forwarded to one of the brokers that are connected to P_X in the logical network overlay (randomly chosen).

Example 5. According to the situation depicted in Figure 4, if a request $\mathcal{S}_R$ reaches broker P_X and $\mathcal{S}3_X$ INTERSECTS $\mathcal{S}_R$, the request is forwarded both to broker P_Z (1) and P_Y (2). On broker P_Z, the matchmaker is applied and $\mathcal{S}1_Z$ EXACT $\mathcal{S}_R$, then the search stops (3) and P_Z replies to P_X with the URLs of concrete implementations associated to $\mathcal{S}1_Z$. On broker P_Y, the matchmaker is applied and $\mathcal{S}2_Y$ INTERSECTS $\mathcal{S}_R$, then P_Y replies to P_X with the URLs of concrete implementations associated to $\mathcal{S}2_Y$, but also in this case the request is not further forwarded to P_H (4), since $\mathcal{S}5_H$ does not add additional capabilities with respect to $\mathcal{S}2_Y$.

Exhaustive policy. This policy follows the same rules of the previous one, but it does not stop when matching services that fully satisfy the request are found; according to this policy, also for each $\mathcal{S}i_X \in MS(\mathcal{S}_R)$ such that $\mathcal{S}i_X$ EXACT | EXTENDS $\mathcal{S}_R$, the request $\mathcal{S}_R$ is forwarded to semantic neighbors to find other services that could present, for example, better non functional features (not discussed in this paper).

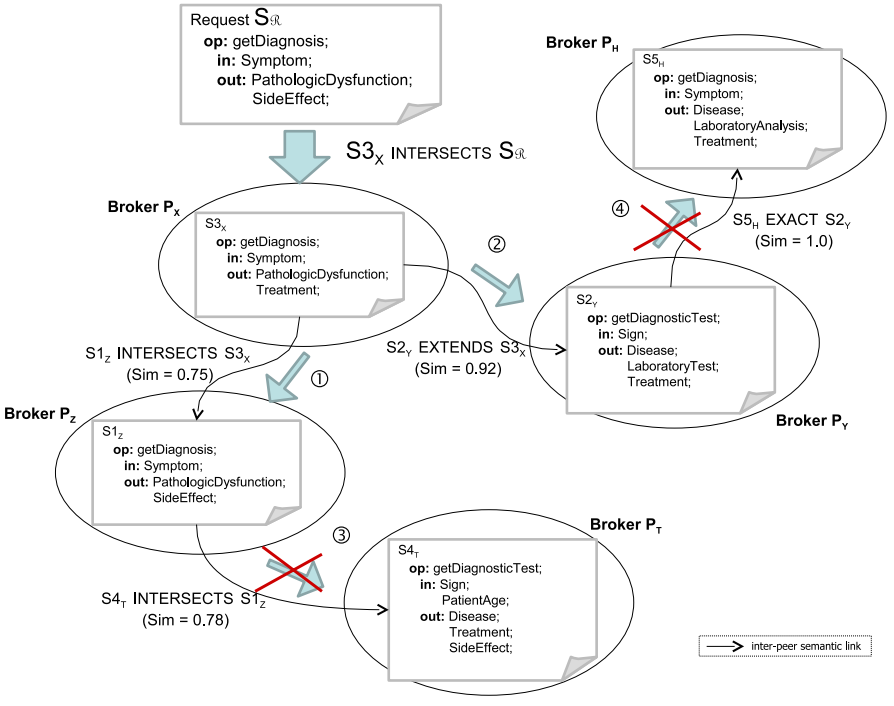


Fig. 4. An example of exploitation of inter-peer semantic links during distributed search

Example 6. In Figure 4, the request is forwarded also from broker P_Z to the broker P_T. Broker P_Z collects results from P_T together with results found locally and then replies to P_X, from which the request came.

5 Experimental Evaluation

In this section, we present a preliminary evaluation of performances of our approach. In particular, we are interested in evaluating experimentally the semantic forwarding strategy. To this purpose, we performed a set of simulations based on NeuroGrid [13], an extensible network overlay simulator in which we have implemented the P2P-SDSD service semantic overlay and the minimal forwarding policy explained in Section 4.

The simulations we have run compares P2P-SDSD forwarding policy with the Gnutella one [10] both in terms of efficiency and scalability. Actually, Gnutella is oriented to file discovery, but for purpose of comparison with P2P-SDSD we have implemented in the simulations a web service discovery process that exploits the Gnutella forwarding policy. The choice of a comparison with Gnutella is due to the fact that both P2P-SDSD and Gnutella define an overlay network built on top of an unstructured P2P network. Apart from the architectural similarities, we

have considered Gnutella since its message forwarding strategy is well-known and it is frequently considered as a reference example. Some other P2P forwarding strategies have been also considered for a comparison with P2P-SDSD and we plan to perform additional experiments in future work. In particular, P2P-SDSD and Gnutella service discovery have been compared on these parameters:

- *hit ratio*: with respect to a submitted request it is the ratio among the number of services retrieved by the distributed search policy and the number of services retrieved if all offered services would be available on the node receiving the request and the local search applied; the higher the hit ratio, the more effective the distributed search by providing an answer similar to a local search; this parameter depends on the TTL value;
- *generated messages*: it is the number of overall messages generated to answer a service request; this parameter depends on the TTL (Time To Live) value and on the peer's average number of connections to its neighbors; the lower the generated messages, the better the scalability, since a lower number of messages per request reduces the possibility of network congestion.

Simulation Parameter	Description
N	Total number of brokers on the network
NS	Total number of available services
Initial TTL	Initial TTL associated to a request message
Simulation Type	P2P-SDSD or Gnutella
γ	Probability that two brokers has established semantic links ($\in$ (0,1))
ASP	Average number of services per broker

Fig. 5. Simulation Parameters

The experiments have been performed: (i) to demonstrate the better hit ratio results of our distributed search with respect to Gnutella search; (ii) to confirm that the use of the P2P-SDSD request forwarding policy results in an improved scalability. The different experimental settings for the simulation are obtained by setting some operative parameters shown in Figure 5. These parameters determine how the simulated logical overlay network is built and run. The simulator generates a random P2P network [13] of N peers and NS artificial services. Services are randomly assigned to each peer according to the ASP parameter. Given a pair of artificial services $\mathcal{S}_1$ and $\mathcal{S}_2$ the kind of match between them is established according to the following distribution of probability: P($\mathcal{S}_1$ MISMATCH$\mathcal{S}_2$) = 50%, P($\mathcal{S}_1$ INTERSECT $\mathcal{S}_2$) = 25%, P($\mathcal{S}_1$ EXTENDS $\mathcal{S}_2$) = P($\mathcal{S}_2$ EXTENDS $\mathcal{S}_1$) = 10%, P($\mathcal{S}_1$ EXACT $\mathcal{S}_2$) = 5%.

The γ value is the probability that a pair of peers has already exchanged probe queries and therefore have possibly established inter-peer semantic links. The Initial TTL determines the maximum number of times a message can be forwarded on the network and therefore the higher this value, the higher the number of peers involved in answering the request. In particular, for the results discussed in the following, the Initial TTL has been varied between 1 and 12. Moreover, we have supposed a medium sized network having N=100 brokers in

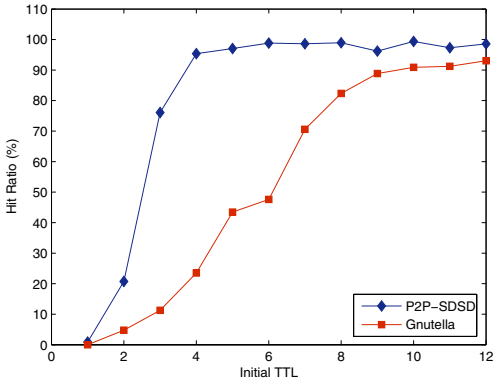


Fig. 6. Hit Ratio vs. Initial TTL

which NS=50 services are globally available and ASP=20 services are on average available on each broker. Finally, the probability that two broker have already exchanged probe queries and consequently established semantic links has been set to 60% ($\gamma = 0.60$).

Figure 6 compares P2P-SDSD and Gnutella approaches with respect to the hit ratio by setting different values for the Initial TTL parameter. The figure shows how P2P-SDSD overperforms the Gnutella even with low TTL values. This can be explained by the fact that inter-peer semantic links allow to selectively reach the most of the peers that provide relevant services with a low number of request forwardings. As TTL gets higher also the Gnutella performs better since the request reaches the most of the peers in the network, but the network overload increases.

The analysis of generated messages has been performed on a series of 13 simulations with Initial TTL value set to 5 and a network of 100 peers. In each simulation we ran, a request has been submitted and we have collected the results obtained with both P2P-SDSD and Gnutella on the same simulated network. In these simulations the average number of generated messages for P2P-SDSD is nearly 19.31, that is about 50% lower than the value of 38.32 obtained by the Gnutella approach.

6 Related Work

In literature, P2P semantic-driven resource discovery has attracted much attention from Web services and Semantic Web area and relies on several efforts in related research fields, such as data integration and emergent semantics in P2P environments [1,6,11], to go beyond limitations of centralized service-oriented architectures. However, in P2P environments also semantic heterogeneity and scalability issues must be addressed. The former requires adoption of ontology-based matchmaking techniques; several current approaches [14,17] rely on a common ontology to express service semantics. In our approach we admit different

reference ontologies, whose semantic gap is bridged thanks to terminological relationships in the thesaurus. For what concerns scalability, traditional flooding-based approaches are affected by low performances while the network size increases, since no organization of services based on semantic similarity is adopted. Recent works proposed such a semantic organization both for structured [2,15] and unstructured P2P network architectures. Service discovery approaches for structured P2P architectures try to go beyond scalability limitations by organizing services through DHT (Distributed Hash Tables) structures, which require much efforts for maintenance and are less flexible. In [12] a semantic service discovery scheme called UbiSearch is proposed for a large-scale ubiquitous computing environments, where services that are semantically close to each other are mapped to nearby positions, efficiently confining the search space for a service request while maintaining high accuracy with respect to a centralized scheme.

Our approach focuses on the service discovery problem applied to unstructured P2P networks. The same problem has been studied in [3,4,5,17]. In [5] a P2P-based system to support efficient access to e-catalogs is provided, where scalability issues are solved by organizing information space in communities that are inter-related using peer relationships, defined as mappings between ontologies. Selection of relevant peers to which queries must be forwarded is based on a query rewriting algorithm. METEOR-S [17] uses a centralized *registry ontology* to classify peer registries. During the discovery process, *registry ontology* is browsed to find the proper registry to which the request must be submitted. ARTEMIS [3] defines a P2P network, where each peer has an ontology, based on medical information standards, to annotate services. Peers store the services they provide in a mediator super-peer. A peer sends a request to its reference mediator expressed in terms of its own ontologies; mediator uses ontology mappings to find matching services in its local registry and forwards the request to other mediators. WSPDS [4] describes a P2P network where peers have local DAML-S ontologies to provide service semantics and links with other peers based on an average similarity between services they provide. When a request is submitted to a peer, it searches for local matching results and forwards the request to all the linked peers, independently from the current request or the local results of the query. In our approach a service semantic overlay is built by relating similar services through inter-peer semantic links in a network of collaborative peers. Like WSPDS, no common ontology is required and no super-peer is defined. However, in our approach, combined use of intra-peer and inter-peer semantic links allows for application of fine-grained request forwarding strategies.

7 Conclusions

In this paper, we proposed a P2P service discovery approach to enable cooperation and communication across distributed Web Information Systems, where Web Services are used to export WIS functionalities. Semantic interoperability is based on a service semantic overlay, over the logical network overlay, built by establishing semantic links among peers that offer comparable services. The semantic overlay is exploited to speed up service discovery and improve its

efficacy without affecting network overload. Preliminary experiments have been performed to confirm the advantages derived from the exploitation of semantic overlay if compared with traditional Gnutella approach. In particular, experimentation are being performed on collaborating entities that are organized in a P2P network, where additional network overload due to service semantic overlay management is relevant mainly during the semantic overlay contitution. Although dynamism of P2P networks has been considered also in this paper, by proposing evolution strategies both for the service semantic overlay and for the logical network overlay, further experimentation will evaluate the impact of the proposed approach on open P2P networks, where dynamism is even more accentuated, and concrete applications in those kinds of networks will be investigated.

References

1. Aberer, K., Cudre-Mauroux, P., Hauswirth, M.: The Chatty Web: Emergent Semantics Through Gossiping. In: Proc. of the 12th International World Wide Web Conference, Budapest, Hungary, pp. 197–206 (2003)
2. Arabshian, K., Schulzrinne, H.: An Ontology-based Hierarchical Peer-to-Peer Global Service Discovery System. Journal of Ubiquitous Computing and Intelligence (JUCI) 1(2), 133–144 (2007)
3. The ARTEMIS Project: A Semantic Web Service-based P2P Infrastructure for the Interoperability of Medical Information Systems, http://www.srdc.metu.edu.tr/webpage/projects/artemis/
4. Banaei-Kashani, F., Chen, C., Shahabi, C.: WSPDS: Web Services Peer-to-Peer Discovery Service. In: Proc. of the Int. Conference on Internet Computing (IC 2004), Las Vegas, Nevada, USA, pp. 733–743 (2004)
5. Benatallah, B., Hacid, M.S., Paik, H.Y., Rey, C., Toumani, F.: Towards semantic-driven, flexible and scalable framework for peering and querying e-catalog communities. Information Systems, Special Issue on Semantic Web and Web Services 31(4-5), 266–294 (2006)
6. Bernstein, P., Giunchigiloa, F., Kementsietsidis, A., Mylopoulos, J., Serafini, L., Zaihrayeu, I.: Data Management for Peer-to-Peer Computing: A Vision. In: Proc. of the 5th International Workshop on the Web and Databases (WebDB 2002), Madison, Wisconsin, pp. 89–94 (2002)
7. Bianchini, D., De Antonellis, V., Melchiori, M.: Flexible Semantic-based Service Matchmaking and Discovery. World Wide Web Journal (2008), http://www.springerlink.com/content/4514473142836174/
8. Bianchini, D., De Antonellis, V., Melchiori, M., Salvi, D.: Semantic Driven Service Discovery for Interoperability in Web Information Systems. In: CAiSE Int. Workshop on Web Information Systems Modelling (WISM 2007), Trondheim, Norway, pp. 743–754 (2007)
9. Bianchini, D., De Antonellis, V., Pernici, B., Plebani, P.: Ontology-based Methodology for *e*-Service discovery. Journal of Information Systems, Special Issue on Semantic Web and Web Services 31(4-5), 361–380 (2006)
10. The Gnutella Protocol Specification v.0.4, http://www.stanford.edu/class/cs244b/gnutella_protocol_0.4.pdf
11. Halevy, A., Ives, Z., Suciu, D., Tatarinov, I.: Schema Mediation in Peer Data Management Systems. In: Proc. of the 19th International Conference on Data Engineering (ICDE 2003), Bangalore, India, pp. 505–516 (2003)

12. Kang, S., Kim, D., Lee, Y., Hyun, S.J., Lee, D., Lee, B.: A Semantic Service Discovery Network for Large-Scale Ubiquitous Computing Environments. ETRI Journal 29(5), 545–558 (2007)
13. NeuroGrid Home Page, http://www.neurogrid.net
14. Paolucci, M., Sycara, K.P., Nishimura, T., Srinivasan, N.: Using DAML-S for P2P Discovery. In: Proceedings of the Int. Conference on Web Services (ICWS 2003), Las Vegas, Nevada, USA, pp. 203–207 (2003)
15. Schmidt, C., Parashar, M.: A Peer-to-Peer Approach to Web Service Discovery. World Wide Web Journal 7(2), 211–229 (2004)
16. The ESTEEM Team. Emergent Semantics and Cooperation in Multi-Knowledge Environments: the ESTEEM Architecture. In: Proceedings of the VLDB Int. Workshop on Semantic Data and Service Integration (SDSI 2007), Vienna, Austria, pp. 1–12 (2007)
17. Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S., Miller, J.: METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services. Journal of Information Technology and Management, Special Issue on Universal Global Integration 6(1), 17–39 (2005)
18. Voulgaris, S., Gavidia, D., van Steen, M.: CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. Journal of Network and Systems Management 13(2), 197–217 (2005)

Filtering Techniques for Rewriting XPath Queries Using Views

Rui Zhou[1], Chengfei Liu[1], Jianxin Li[1], and Junhu Wang[2]

[1] Swinburne University of Technology, Melbourne, Australia
{rzhou,cliu,jili}@ict.swin.edu.au
[2] Griffith University, Gold Coast, Australia
J.Wang@griffith.edu.au

Abstract. In this paper, we propose several filtering techniques for rewriting XPath queries using views. The work is motivated by scenarios dealing with large quantities of queries and views, such as semantic query caching and data integration. Considerable rewriting computation could be saved if we manage to efficiently discover that, given a query Q and a view V, there does not exist a rewriting for Q using V. In contrast to $O(|Q||V|)$ ($O(|Q||V|^2)$) time complexity to compute an equivalent rewriting (contained rewriting) for $XP^{\{/,//,[]\}}$, we devise linear algorithms running in $O(|Q|)$ to filter queries for both equivalent rewriting and contained rewriting. Our filtering algorithms can be extended to support queries and views in $XP^{\{/,//,[],*\}}$, where the equivalent (or contained) rewriting existence problem is still coNP-hard.

1 Introduction

Answering queries using views (also known as rewriting queries using views) is to utilize previously defined (possibly materialized) views to evaluate queries in order to save the cost of accessing large real database or provide a privacy-preserving publishing. It is a classic problem, and appears in many applications, such as query optimization, data integration, data warehouse and query caching [1]. With the prevalence of XML technologies on the web, rewriting XML queries using XML views has caught the attention of both researchers and system designers, and is believed to be a promising technique in web application development.

Two types of rewritings for XPath [2] queries have been studied in the literature, because XPath serves as the core sub-language of the major XML query languages such as XQuery [3] and XSLT [4]. One is *equivalent rewriting* [5]: Given a materialized view V of a database D, an equivalent rewriting Q' of a query Q, runs over the view V producing the same set of answers as evaluating Q over D, i.e. $Q'(V) = Q(D)$[1]. However, an equivalent rewriting may not always exist, and moreover part of answers covered by the view are still valuable. Therefore

[1] Here, we use $Q(V)$ and $Q(D)$ to denote the returned query results by evaluating Q on V and D respectively.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 307–320, 2008.

contained rewriting [6] is introduced and can be described as follows: Given a view V on a database D, a contained rewriting Q' of a query Q, runs over V producing a subset of answers as evaluating Q over D, i.e. $Q'(V) \subseteq Q(D)$. In this paper, we study filtering techniques for both types of the rewritings without schema information. We will take schema into account in future work.

We now introduce the motivation of this work. Users may issue a large number of queries against a view V. Although to test whether a query Q can be answered by V (i.e. whether there exists a rewriting for Q using V) is P-TIME [2] efficient, with complexity $O(|Q||V|)$ for equivalent rewriting [5] and $O(|Q||V|^2)$ for contained rewriting [6], it is still of great importance if we can cheaply filter part of (as many as possible) unanswerable queries for V. Consequently, a lot of computation cost will be saved. Fig. 1 shows the framework of evaluating queries using materialized view V. The shaded filtering step is the focus of this paper. Obviously, the filtering step should possess two properties: (i) It should not introduce false negatives, which means if we can find a rewriting for a query using V, the query should not be rejected; (ii) It should be more efficient than computing a rewriting for the query, otherwise we would rather directly find a rewriting. In real applications, there may also be a large number of views, for instance, in data integration, many sources publish their views in Local-As-View architecture. Therefore filtering techniques play a significant role when we have to deal with plenty of queries and views.

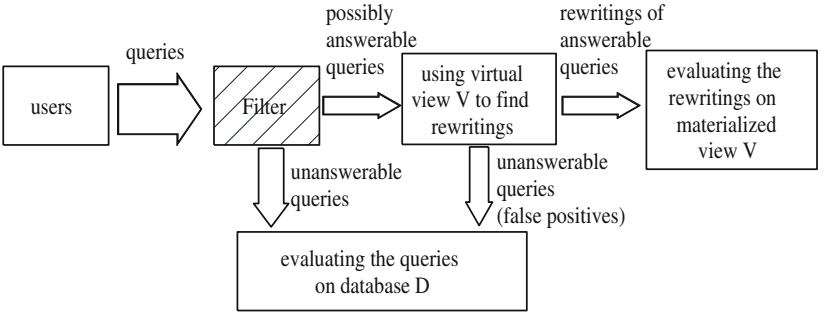


Fig. 1. The framework of answering queries using view V

In this paper, we devise a set of $O(|Q|)$ algorithms to filter queries. And we study the filtering for both equivalent rewriting and contained rewriting. The basic idea is to verify if the structural relationships in a query could be satisfied in a view, given that label preserving and structure preserving are the key conditions in discovering a homomorphism (for finding an equivalent rewriting) or a useful embedding (for finding a contained rewriting). We use index to capture the structural relationships in the view, and develop two algorithms for equivalent rewriting, i.e. Lazy Algorithm and Eager Algorithm. Moreover, Eager Algorithm

[2] Actually this is the result for XPath subset $XP^{\{/,//,[]\}}$, and complexity is even worse as coNP-hard for queries and views in $XP^{\{/,//,[],*\}}$.

can be modified to support contained rewriting. We first study all of the above for queries and views in subset $XP^{\{/,//,[]\}}$, featuring child axes, descendant axes and branches in XPath, and then discuss the problem for $XP^{\{/,//,[],*\}}$ including wildcards. Our contributions are highlighted as follows:

- We propose a novel index to capture the relationships between nodes in a view, and the index can be utilized to filter unanswerable queries.
- We devise a set of linear algorithms to filter queries with respect to both equivalent rewriting and contained rewriting.
- We study the problem with both queries and views in subclass $XP^{\{/,//,[],*\}}$, which contains most popular features in XPath.

The rest of this paper is organized as follows. In Section 2, we will give some notations and rewriting background knowledge. Then we introduce the filtering index serving as the base for filtering techniques in Section 3. The filtering algorithms will follow in Section 4 including two algorithms for equivalent rewriting and one for contained rewriting. In Section 5, we extend XPath queries from $XP^{\{/,//,[]\}}$ to a larger subset $XP^{\{/,//,[],*\}}$. Related work is given in Section 6. Finally, we draw a conclusion and propose some future work in Section 7.

2 Preliminaries and Problem Statement

In this section, we introduce some background knowledge and notations used in the following sections.

2.1 XPath Query

XPath is the core subclass of XML query languages. We consider a subset of XPath featuring child axes (/), descendant axes (//), branches ([ ]), and wildcards (*). It can be represented by the following grammar:

$$P \rightarrow l|*|P/P|P//P|P[/P]|P[//P]$$

where l is a label from alphabet Σ. We denote this subset as $XP^{\{/,//,[],*\}}$. We will first discuss a restricted subset $XP^{\{/,//,[]\}}$ in Section 3 and Section 4, and then add $*$ in Section 5.

Definition 1. *An XPath query Q can be expressed as a tree pattern $(N_q, E_q, r_q, d_q, \Sigma_q)$, where*

- *N_q is the node set, and $\forall n \in N_q$, n has a label in a finite alphabet Σ_q, denoted as $label(n)$;*
- *E_q is the edge set, and $\forall e \in E_q, type(e) \in \{/, //\}$. We use the word "pc-edge"("ad-edge") to represent the type of an edge, "/"("//").*
- *r_q is the root node of the query;*
- *d_q is the distinguished (return) node of the query, identified with a circle;*

Similarly, an XPath view V can be expressed as $(N_v, E_v, r_v, d_v, \Sigma_v)$. For ease of discussion, we introduce the following definition.

Definition 2. *Given a view $V = (N_v, E_v, r_v, d_v, \Sigma_v)$ and two nodes $n_1, n_2 \in N_v$: (I) $pc(n_1, n_2)$ holds in V if n_1 is the parent node of n_2; (II) $ad(n_1, n_2)$ holds in V if n_1 is an ancestor node of n_2.*

Obviously, we have the following corollaries:

- e=(n_1,n_2) is a pc-edge. $\Leftrightarrow$ $pc(n_1, n_2)$ is true.
- e=(n_1,n_2) is a ad-edge. $\Rightarrow$ $ad(n_1, n_2)$ is true.
- $pc(n_1, n_2)$ is true. $\Rightarrow$ $ad(n_1, n_2)$ is true.

2.2 XPath Query Rewriting Using View

Equivalent rewriting and contained rewriting have been formulated in Section 1. We now recall two techniques to find them, i.e. homomorphism [5] and useful embedding [6].

Definition 3. *Given two tree patterns $t_1 = (N_{t_1}, E_{t_1}, r_{t_1}, d_{t_1}, \Sigma_{t_1})$ and $t_2 = (N_{t_2}, E_{t_2}, r_{t_2}, d_{t_2}, \Sigma_{t_2})$, a **homomorphism** is a matching $h : N_{t_1} \rightarrow N_{t_2}$, satisfying:*

- *Root preserving: $h(r_{t_1}) = r_{t_2}$;*
- *Label preserving: $\forall n \in N_{t_1}$, $label(n) = *$ or $label(n) = label(h(n))$;*
- *Structure preserving: $\forall e = (n_1, n_2) \in E_{t_1}$, if e is pc-edge, $pc(h(n_1), h(n_2))$ holds in t_2; otherwise $ad(h(n_1), h(n_2))$ holds in t_2.*

With the definition of homomorphism, we have: an equivalent rewriting for Q using V exists, only if there exists a node n in Q such that two homomorphisms exist between $Q_{up}(n)$ and $V_{up}(d_v)$, and d_q is not an ancestor of node n. Here, $Q_{up}(n)$ is the subpattern obtained by removing all descendants of n in Q, $V_{up}(d_v)$ is the subpattern obtained by removing all descendants of d_v in V. Fig. 2 shows an example. Q' called compensation query, is the equivalent rewriting for Q using V. In this paper, we won't discuss the construction of the compensation query, and will only focus on determining the existence of a rewriting. And according to the above, we ignore the descendants of d_v and use V to represent $V_{up}(d_v)$ in the rest of the paper.

Definition 4. *Given two tree patterns $t_1 = (N_{t_1}, E_{t_1}, r_{t_1}, d_{t_1}, \Sigma_{t_1})$ and $t_2 = (N_{t_2}, E_{t_2}, r_{t_2}, d_{t_2}, \Sigma_{t_2})$, an embedding is a **partial** matching $f : N_{t_1} \rightarrow N_{t_2}$, satisfying root preserving, label preserving, structure preserving as in Definition 3, and additionally is upward closed: if node n in N_{t_1} is defined by f, all ancestors of n in N_{t_1} are defined by f. For each path of t_1, we call the last defined node an anchor node. An embedding is a **useful embedding** if both of the following hold: (i) every anchor node n_a in t_1 satisfies:*

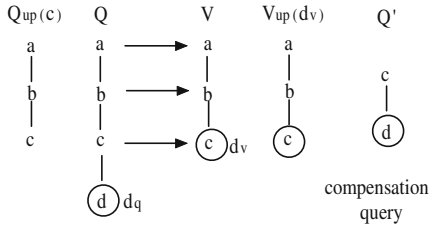


Fig. 2. Homomorphism and equivalent rewriting

- n_a *is a leaf node;*
- *or:* $f(n_a) = d_{t_2}$*;*
- *or:* $ad(f(n_a), d_{t_2})$ *holds in* t_2 *and* $\forall n_b((n_a, n_b) \in E_{t_1})$, (n_a, n_b) *is an ad-edge.*

and (ii) for the anchor node n_d *on the distinguished path of* t_1*, either* $n_d = d_{t_1}$ *or* $ad(n_d, d_{t_1})$ *holds in* t_1*.*

A contained rewriting for Q using V exists, if there exists an useful embedding from Q to V. The above is a brief introduction including some key ideas. We point readers to [5,6] to see detailed discussions of finding equivalent rewritings and contained rewritings based on homomorphism and useful embedding.

3 Building Index for Filtering

In order to illustrate how to filter an unanswerable query Q w.r.t. a view V, we first introduce an important index I built on V. Index I serves as the basic structure for filtering algorithms in the next section.

Index I consists of two parts, I_{pc} and I_{ad}. Each part is a matrix containing $|\Sigma_v|^2$ entries, where $|\Sigma_v|$ is the number of labels in view V. Each entry is associated with a label pair (lab_1, lab_2) (where lab_1, $lab_2 \in \Sigma_v$), and has a bit vector with length $|paths(V)|$. Here, $|paths(V)|$ is the number of paths in V. We denote the bit vector as $I_{pc}(lab_1, lab_2)/I_{ad}(lab_1, lab_2)$. We use $I_{pc}(lab_1, lab_2)[i]$ to represent the ith bit of $I_{pc}(lab_1, lab_2)$, and obviously $I_{pc}(lab_1, lab_2)[i] \in \{0, 1\}$. If every bit of $I_{pc}(lab_1, lab_2)$ is "0", we say $I_{pc}(lab_1, lab_2) = 0$.

We utilize index I to capture the pc(ad)-relationship between two nodes in V. Precisely speaking, take pc-relationship as an example, if $pc(n_1, n_2)$ holds on the ith path in V, and $label(n_1) = lab_1$, $label(n_2) = lab_2$, then $I_{pc}(lab_1, lab_2)[i] = 1$. Similar principle can be applied on ad-relationship, i.e. if $ad(n_1, n_2)$ holds, $I_{ad}(label(n_1), label(n_2))[i] = 1$. Note here n_1, n_2 do not have to be adjacent nodes. Fig. 3 gives a view V and its corresponding index I_{pc} and I_{ad}.

Find an algorithm to build I using V is not difficult. Algorithm 1 shows an example. Line 1 takes $O(|V|)$ time, searching for the leaf nodes and numbering them. The number of leaf nodes N_{leaf} equals to $|paths(V)|$. Line 2 runs in $O(|E|)$ and finds out the corresponding path set $P(n)$ for each node n. Each path in $P(n)$ runs through n. Line 3-7 assigns values to the entries in I_{pc} and I_{ad}. Since

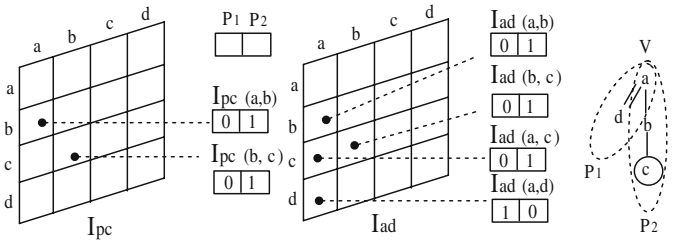


Fig. 3. Index I_{pc} and I_{ad} built on view V

Algorithm 1. Constructing Index for View V

Input: a view $V = (N_v, E_v, r_v, d_v, \Sigma_v)$
Output: an Index I

1: Find the leaf node set $N_{leaf} \subset N_v$ and number the leaf nodes;
2: Traversing V bottom-up, find the path set $P(n)$ for each node n;
3: **for all** node pair (n_1, n_2) in V **do**
4: **if** $pc(n_1, n_2)$ holds **then**
5: For $I_{pc}(label(n_1), label(n_2))$, set bit positions in $P(n_1) \cap P(n_2)$ to 1;
6: **end if**
7: For $I_{ad}(label(n_1), label(n_2))$, set bit positions in $P(n_1) \cap P(n_2)$ to 1;
8: **end for**
9: **return** I;

we have to set values for $|\Sigma_v|^2$ entries in I, it is not surprising that this step needs $O(|\Sigma_v|^2)$ time, bounded by $O(|V|^2)$. Building index is not of linear complexity, but once we have the index built, we can devise linear algorithms to filter large part of unanswerable queries .

4 Filtering Algorithms for Two Types of Rewritings

In this section, we will first introduce a basic idea to filter an unanswerable query Q using V. Then based on the basic idea, we will give two algorithms, Lazy Algorithm and Eager Algorithm to detect if it is impossible to find an equivalent rewriting for Q using V. Eager Algorithm can be modified to filter queries w.r.t. contained rewritings.

4.1 Basic Idea

Recall in Section 2, label preserving and structure preserving are the key conditions in discovering a homomorphism (for equivalent rewriting) or a useful embedding (for contained rewriting). It is desirable if we can efficiently discover that the structural relationships in Q could not be satisfied in V. We explain it with an example, see Fig. 4. Assume indexes are built for V_1, V_2, V_3 and V_4,

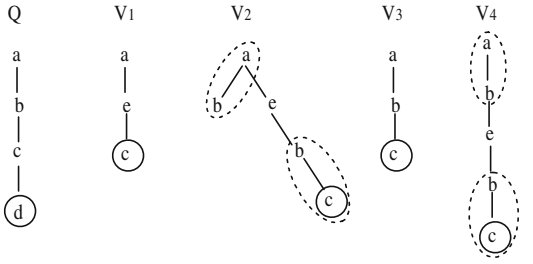


Fig. 4. Basic idea of filtering algorithms

Q is unanswerable w.r.t. V_1, because $pc(a, b)$ and $pc(b, c)$ do not hold in V_1; Q is unanswerable w.r.t. V_2 as well, because $pc(a, b)$ and $pc(b, c)$ do not hold on the same path in V_2, whereas a/b and b/c are on the same path in Q. However, Q cannot be filtered by V_3 and V_4. We can answer Q with V_3. But for V_4, Q is a false positive since there does not exist a rewriting for Q using V_4, though $pc(a, b)$ and $pc(b, c)$ hold on the same path in V_4 as they are in Q.

To summarize, given a query Q and a view V, let $e_1 = (n_1, n_2)$, $e_2 = (n_3, n_4)$ be two edges on the same path in Q (assume e_1, e_2 to be pc-edges without loss of generality), it is impossible to find an equivalent rewriting for Q using V if the following holds:

$$I_{pc}(label(n_1), label(n_2)) \wedge I_{pc}(label(n_3), label(n_4)) = 0$$

which means if label pairs, $(label(n_1), label(n_2))$ and $(label(n_3), label(n_4))$, do not lie on the same path in V, we cannot find a rewriting for Q. Note here, if (n_1, n_2) is an ad-edge, we use $I_{ad}(label(n_1), label(n_2))$ instead of $I_{pc}(label(n_1), label(n_2))$. The above observation is the key idea in our following algorithms.

Here, the complexity of the bit-And operation depends on the size of the bit vectors (is proportional to the length of bit vectors), which is $|paths(V)|$, the number of paths in the view. But in practical settings, this length is probably limited by computer word length, or a small multiple thereof, so this may not impact so much the actual performance. We treated this operation as $O(1)$ in the following paragraphs.

4.2 Equivalent Rewriting

For equivalent rewriting, the first step is to trim Q into Q' according to Lemma 4.7 in [5]. This can be done by firstly finding a node n on the distinguished path of Q, satisfying length(r_v,d_v)=length(r_q,n), where length(n_1,n_2) means the number of path steps between node n_1 and node n_2; and secondly removing all the descendants of n in Q, i.e. $Q' = Q_{up}(n)$. The reason to trim Q into $Q_{up}(n)$ is: nodes under n do not contribute to detecting the existence of an equivalent rewriting, since n is supposed to be mapped onto d_v. This step runs in $O(|V|)$ and serves as the first step in both Lazy Algorithm and Eager Algorithm.

Algorithm 2. Lazy Algorithm for Filtering Equivalent Rewriting

Input: a query Q, a view V and an index I on V
Output: a boolean value

```
1: Find a node n on the distinguished path of Q, such that length(r_v,d_v)=length(r_q,n),
   and Trim Q into Q_up(n);
2: for each path p_i ∈ Q_up(n) do
3:    Initiate a bit vector I_0 with length |paths(Q_up(n))| and every position set to 1;
4:    for each edge (n_1, n_2) on p_i do
5:       if (n_1, n_2) is a pc-edge then
6:          I_0=I_0 ∧ I_pc(label(n_1), label(n_2));
7:       else
8:          I_0=I_0 ∧ I_ad(label(n_1), label(n_2));
9:       end if
10:   end for
11:   if I_0 = 0 then
12:      return false;
13:   end if
14: end for
15: return true;
```

Lazy Algorithm. We now introduce the main part of Lazy Algorithm (see Algorithm 2). We will explain why we use the word "lazy" later. After trimming Q into $Q_{up}(n)$, we identify, for each path p_i in $Q_{up}(n)$, if the edge relationships on p_i can be satisfied on a single path in V. We do this by retrieving the bit vector of each label pair $(label(n_1), label(n_2))$ that associates an edge (n_1, n_2) on p_i from index I, and performing bit-AND($\wedge$) operation to the bit vectors (line 3-10). If there doesn't exist any bit set to "1" in the result bit vector, which means there doesn't exist a path in V that can accommodate all edge relationships on p_i, then Q can be filtered (line 11-13); otherwise we go on to test other paths in $Q_{up}(n)$. In Fig. 5, Q is unanswerable using V_1, because $ad(a, e)$ is not satisfied on any path in V_1. For V_2, Q is answerable. Later we will know, with respect to contained rewriting, Q turns to be answerable using V_1.

Lazy Algorithm runs in $O(|Q||E_q|)$, where $|Q|$ is the number of nodes in Q, $|E_q|$ is the number of edges in Q. $O(|Q||E_q|)$ is bounded by $O(|Q|^2)$, since $|E_q| = |Q|-1$. Note here, the complexity is $O(|Q|^2)$, not $O(Q)$, because one edge may be computed for several times, depending on how many paths are sharing the edge.

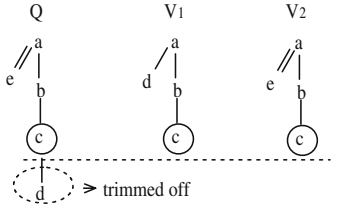


Fig. 5. Filtering queries with respect to equivalent rewriting

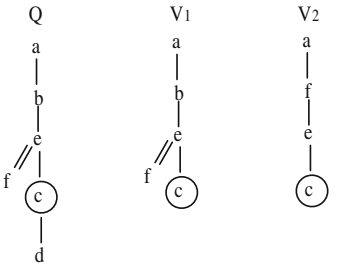


Fig. 6. Drawbacks of Lazy Algorithm

This is also one of the disadvantageous aspects of the algorithm. We will explain right away in this paragraph. In practice, we expect the filtering algorithm could be in $O(|Q|)$, otherwise the filtering step may not be efficient. The reason of Lazy Algorithm being expensive lies in two aspects: (a) **Repeated computation**. See Fig. 6, for view V_1, the algorithm computes $I_0 \wedge I_{pc}(a,b) \wedge I_{pc}(b,e)$ twice, once for path $a/b/e/c$ and once for $a/b/e//f$. Here, $a/b/e$ is shared by the two paths. (b) **Useless computation**. For view V_2, Q can be filtered at an early stage. Because $pc(a,b)$ doesn't hold in V_2, whereas all paths in Q contain edge (a,b). As a result, we can draw the conclusion right away after examining $pc(a,b)$ in V_2. We will remedy the two drawbacks in Eager Algorithm.

Eager Algorithm. We now introduce Eager Algorithm (Algorithm 3) in detail. After trimming query Q as in Lazy Algorithm, we push a pair (r_q, I_0) into a global queue, where r_q is the root of query Q, I_0 is bit vector with lenghth $|paths(V)|$ and every position set to "1". In each iteration of the loop, pop a pair (n_0, I_0) from the global queue and detect if $I_0 = 0$ (line 5-8). If yes, that means relationships on path from r_q to n_0 cannot be fully satisfied in V, and thus Q can be filtered out. Otherwise, if n_0 is a leaf node, that means path from r_q to n_0 in Q can be satisfied in V, then we go on to test other paths in Q, i.e. pop another pair from the global queue and repeat the iteration. If n_0 is not a leaf node, for each child n_i of n_0, we obtain the next partial result from r_q to n_i by bit-ANDing I_0 with $I_{pc}(lable(n_0), label(n_i))$ (or $I_{ad}(lable(n_0), label(n_i))$). This new bit vector and node n_i are afterwards pushed into the global queue as a pair for future test.

Eager Algorithm runs in $O(|Q|)$, and is in linear complexity. Edges in Q are visited only once. This is achieved by utilizing a queue to record the intermediate results. In contrast to Lazy Algorithm, Eager Algorithm will report if a query is unanswerable as early as possible. On the other hand, the space cost of Eager Algorithm $O(|Q|)$ is worse than Lazy Algorithm $O(1)$.

4.3 Contained Rewriting

For contained rewriting, we cannot do the trimming step as for equivalent rewriting, since Lemma 4.7 in [5] doesn't hold for contained rewriting. Fig. 7 shows an

Algorithm 3. Eager Algorithm for Filtering Equivalent Rewriting

Input: a query Q, a view V and an index I on V
Output: a boolean value

```
1: Find a node n on the distinguished path of Q, such that length(r_v,d_v)=length(r_q,n),
   and Trim Q into Q_up(n);
2: Initiate a queue globalQueue and initiate a bit vector I_0 with length
   |paths(Q_up(n))| and every position set to 1;
3: globalQueue.in(r_q, I_0);
4: while ¬globalQueue.empty() do
5:    (n_0, I_0) = globalQueue.out();
6:    if I_0 = 0 then
7:       return false;
8:    end if
9:    if n_0 is not a leaf node then
10:      for each child node n_i of n_0 in Q do
11:         if e = (n_0, n_i) is a pc-edge then
12:            globalQueue.in(n_i, I_0 ∧ I_pc(label(n_0), label(n_i)));
13:         else
14:            globalQueue.in(n_i, I_0 ∧ I_ad(label(n_0), label(n_i)));
15:         end if
16:      end for
17:   end if
18: end while
19: return true;
```

example. The reason is that finding a contained rewriting is based on finding a useful embedding, which is a one-direction partial matching. A useful embedding is different from a homomorphism in that, for a useful embedding f, each anchor node n_a (if not a leaf node) possesses either of the following properties: (i) n_a is mapped to d_v; (ii) n_a connects its children with //. As a result, not every path in Q needs to be fully matched, and the unembedded part of each path can be further tested in the real (materialized) view. This brings challenges for testing the existence of contained rewritings for Q using V. Fortunately, we are able to modify Eager Algorithm for filtering Q w.r.t. contained rewriting, because useful embedding is upward closed and Eager Algorithm accordingly traverses the query tree in a top-down manner. Full algorithm is shown in Algorithm 4.

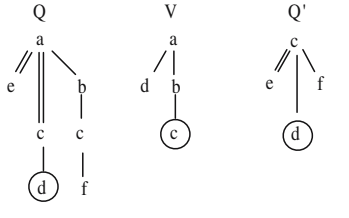


Fig. 7. Q' is a contained rewriting for Q using V

Algorithm 4. Algorithm for Filtering Contained Rewriting

Input: a query Q, a view V and an index I on V
Output: a boolean value

```
1: Initiate a queue globalQueue and initiate a bit vector I_0 with length |paths(Q))|
   and every position set to 1;
2: globalQueue.in(r_q, I_0);
3: while ¬globalQueue.empty() do
4:   (n_0, I_0) = globalQueue.out();
5:   if I_0 = 0 then
6:     return false;
7:   end if
8:   if n_0 is not a leaf node then
9:     if label(n_0) = label(d_v) && I_0[i_d] = 1 then
10:      continue;
11:    else
12:      for each child node n_i of n_0 in Q do
13:        if e = (n_0, n_i) is a pc-edge then
14:          globalQueue.in(n_i, I_0 ∧ I_pc(label(n_0), label(n_i)));
15:        else
16:          if I_0[i_d] ≠ 1 then
17:            globalQueue.in(n_i, I_0 ∧ I_ad(label(n_0), label(n_i)));
18:          end if
19:        end if
20:      end for
21:    end if
22:  end if
23: end while
24: return true;
```

We won't explain it detailedly, since it is similar to Algorithm 3. Here, line 9-11 and line 12-20 attempt to determine if n_0 is an anchor in Q by checking the above two properties (i)(ii) respectively. And i_d in line 9 and 16 means the path number of the distinguished path in Q.

5 XPath Queries with Wildcard Nodes

We have discussed filtering techniques for subclass $XP^{\{/,//,[]\}}$. In this section, we extend our work to a general case $XP^{\{/,//,[],*\}}$, allowing $*$ to appear in both the query and the view.

When we have wildcard nodes in view V, the alphabet turns into $\Sigma_v \cup \{*\}$. However, we choose not to consider $*$ when building index I for V, because the algorithms will introduce false negatives. Fig. 8(a) gives an example. We cannot find a homomorphism or useful embedding from Q to V, because the $*$ node in Q cannot be mapped to the $*$ node in V. But actually, $/a/*//c$ and $/a//*/c$ are equivalent patterns. Rewritings do exist for Q using V. Therefore, to avoid false negatives (corresponding the first property in Section 1), we ignore $*$ nodes when

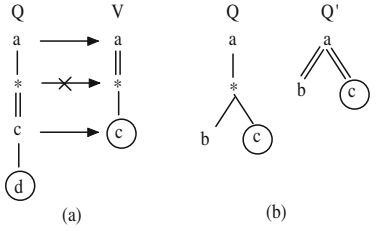


Fig. 8. Considering wildcard nodes in queries and views

testing structural relationships. Algorithm 1 can be reused for $XP^{\{/,//,[],*\}}$, with a last step added to eliminate $*$ entries in I.

For the query Q, a wildcard node n_* can be eliminated as follows: connect each node in Children(n_*) to Parent(n_*) with // and then delete n_*. By repeatedly performing this operation until there is no $*$ node in Q, we can remove all $*$ nodes without losing relationships between all the other nodes. The result query Q' is in $XP^{\{/,//,[]\}}$. A simple example is given in Fig. 8(b). Thereafter we are able to use the established method in Section 3 and Section 4 to detect if Q' can be filtered. Since Q' is a relaxed form of Q (Q is contained in Q'), if Q' can be filtered, Q is doomed to be filtered. The filtering algorithms will not introduce false negatives.

6 Related Work

Answering queries using views has been extensively studied for a long time. Halevy [1] did a survey on this problem over relational databases and pointed out its wide impact on a number of data management applications, such as query optimization, data integration, data warehouse design and semantic query caching. Efficient algorithms were developed as well, eg. MiniCon [7], bucket [8], inverse-rules [9, 10], to tackle the problem in the relational context.

While in the XML context, fruitful research achievements have been made. Containment for a fragment of XPath queries $XP^{\{/,//,[],*\}}$, including child axes, descendant axes, branches and wild cards, is shown to be coNP-compelete in [11], though for three subclasses (combining child with any two of descendant, branch and wildcard), $XP^{\{/,[],*\}}$, $XP^{\{/,//,[]\}}$ and $XP^{\{/,//,*\}}$, the containment problem is in PTIME. [11] also proposed a PTIME-efficient but incomplete algorithm to determine containment in $XP^{\{/,//,[],*\}}$. And this homomorphism-based algorithm is thereafter utilized by the works [12, 5] to evaluate equivalent rewritings of XPath queries using materialized views. Contained rewriting is studied in [6] with and without schema, but only for $XP^{\{/,//,[]\}}$. Our work does not overlap with the above works, and can be considered as a preceding step to accelerate the performance of the state-of-the-art works. In addition, other works [13,14,15] have considered schema information (or constraints) in reformulating queries or determining XPath containment. The work [16] uses multiple nested views in stead of a single view to rewrite an XQuery expression using XQuery views.

7 Conclusions and Future Work

In this paper, we have developed some filtering algorithms for rewriting XPath queries using views. These algorithms are able to discover unanswerable queries efficiently without actually computing the rewritings using views. Filtering algorithms are studied for both types of rewritings, equivalent rewriting and contained rewriting. XPath queries and views are allowed in $XP^{\{/,//,[],*\}}$, which is a representative subset of XPath queries including child axes, descendant axes, branches and wildcards.

There are several promising directions for future work. One is to consider constraints in schema if view schema is available. Another is to discuss filtering techniques for answering queries using multiple views, i.e. combining a set of views to answer a query. The last but not least aspect is to study the effectiveness of the filtering algorithms. We expect the outcome as: the less false positives produced, the better a filtering algorithm is.

Acknowledgments. We would like to thank the reviewers of this paper for their insightful comments and suggestions. This work was supported by the Australian Research Council Discovery Project under the grant number DP0878405.

References

1. Halevy, A.Y.: Answering queries using views: A survey. VLDB J. 10(4), 270–294 (2001)
2. Berglund, A., Boag, S., Chamberlin, D., Fernndez, M.F., Kay, M., Robie, J., Simon, J.: XML path language (XPath) 2.0. In: W3C Recommendation (January 2007), `http://www.w3.org/TR/xpath20`
3. Boag, S., Chamberlin, D., Fernndez, M.F., Florescu, D., Robie, J., Simon, J.: XQuery 1.0: An XML query language. In: W3C Recommendation (January 2007), `http://www.w3.org/TR/xquery`
4. Kay, M.: XSL transformations (XSLT) version 2.0. In: W3C Recommendation (January 2007), `http://www.w3.org/TR/xslt20/`
5. Xu, W., Özsoyoglu, Z.M.: Rewriting XPath queries using materialized views. In: VLDB, pp. 121–132 (2005)
6. Lakshmanan, L.V.S., Wang, H., Zhao, Z.J.: Answering tree pattern queries using views. In: VLDB, pp. 571–582 (2006)
7. Pottinger, R., Halevy, A.Y.: Minicon: A scalable algorithm for answering queries using views. VLDB J. 10(2-3), 182–198 (2001)
8. Srivastava, D., Dar, S., Jagadish, H.V., Levy, A.Y.: Answering queries with aggregation using views. In: VLDB, pp. 318–329 (1996)
9. Qian, X.: Query folding. In: ICDE, pp. 48–55 (1996)
10. Duschka, O.M., Genesereth, M.R.: Answering recursive queries using views. In: PODS, pp. 109–116 (1997)
11. Miklau, G., Suciu, D.: Containment and equivalence for an XPath fragment. In: PODS, pp. 65–76 (2002)
12. Balmin, A., Özcan, F., Beyer, K.S., Cochrane, R., Pirahesh, H.: A framework for using materialized XPath views in XML query processing. In: VLDB, pp. 60–71 (2004)

13. Deutsch, A., Tannen, V.: Reformulation of XML queries and constraints. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) ICDT 2003. LNCS, vol. 2572, pp. 225–241. Springer, Heidelberg (2002)
14. Wood, P.T.: Containment for XPath fragments under DTD constraints. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) ICDT 2003. LNCS, vol. 2572, pp. 297–311. Springer, Heidelberg (2002)
15. Neven, F., Schwentick, T.: XPath containment in the presence of disjunction, DTDs, and variables. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) ICDT 2003. LNCS, vol. 2572, pp. 312–326. Springer, Heidelberg (2002)
16. Onose, N., Deutsch, A., Papakonstantinou, Y., Curtmola, E.: Rewriting nested XML queries using nested views. In: SIGMOD Conference, pp. 443–454 (2006)

Efficient Top-k Data Sources Ranking for Query on Deep Web*

Derong Shen, Meifang Li, Ge Yu, Yue Kou, and Tiezheng Nie

Northeastern University, Shenyang, China, 110004
shenderong@ise.neu.edu.cn, Li.Meifang@gmail.com,
yuge@ise.neu.edu.cn, kouyue@ise.neu.edu.cn,
nietiezheng@ise.neu.edu.cn

Abstract. Efficient Query processing on deep web has been gaining great importance due to large amount of deep web data sources. Nevertheless, how to discover the most relevant data sources on deep web is still a challenging issue. Inspired by observations on deep web, the paper presents a novel top-k ranking strategy to rank relevant data sources according to user's requirement. First, it applies an attribute based dominant pattern growth (ADP-growth) algorithm to mine the most dominant attributes, and then employs a top-k style ranking algorithm on those attributes to exploit the most relevant data sources with candidate pruning and early termination, which considers the probability of result merging. Further, it improves the algorithm by incorporating relevant attributes based searching strategy to find the data sources, which has been proved of higher efficiency. We have conducted extensive experiments on a real world dataset and demonstrated the efficiency and effectiveness of our approach.

1 Introduction

At present, the growing prevalence of data sources on deep web has drawn great attention all over the world [1],[2]. The "query-only" access mode essentially distinguishes data sources on deep web from those on link-based surface web. Survey in April 2004 estimated 450,000 online databases, with 307,000 deep web sources and a total of over 1.2 million query interfaces [1], and much richer with web developing. However, information on deep web is only available as responses to dynamic queries, and thus users are trapped in finding the right sources for query.

Thereby, how to select the most useful data sources is an urgent issue to be resolved.

1.1 Observations and Motivation

Deep web, featuring mostly structured data sources with a dominating ratio of 3.4:1 versus unstructured ones[1]. Survey [1] indicates two important features of deep web:1) proliferating sources: as web scales, many existing data sources tend to

* This work is supported by the National Science Foundation (60673139, 60573090), the National High-Tech Development Program (2008AA01Z146).

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 321–336, 2008.

provide structured information in the same domain, such as car, book, paper domains; 2) converging vocabularies: the aggregate schema vocabulary of data sources in the same domain tends to converge at relatively small size. Studies have observed that data sources under deep web conform to Zipf-like distribution[1], revealing the property of heavy-tailed distribution[1],[3]. With this observation, we found two significant properties. First, the heavy-ranked attributes (e.g., Title, Author, ISBN in book domain in Fig.1,) are extremely frequent, occurring in almost every schema of deep web. Consider frequencies and correlations of those attributes exclusively can greatly reduce time cost in data sources ranking since their scores are dominant in schema matching. Second, for those tail-ranked attributes (e.g., Address(Destination) in book domain in Fig.1), they rarely occur and thus are much less important in matching since their rareness indicates that very few other sources will consider them useful. Moreover, the rare attributes property poses some challenges in schema matching since rare attributes will result in an overestimated frequency, and the schema has to be large enough to justify its occurrence being sufficiently rare. He et al. have addressed this issue by attribute selection and rare attribute smoothing [4],[5].

Inspired by data mining, this paper proposes another method to handle this problem, i.e., an attribute based dominant pattern growth (ADP-growth) algorithm to mine the dominant pattern item sets. Note here we distinguish dominant attribute from frequent one since the dominance of an attribute is not only characterized by its frequency, but also its correlations with other attributes and the users' preference over it based on their submitted queries, thus our ADP-growth algorithm is an improved modification of the FP-growth algorithm[6],[7] for our scenario.

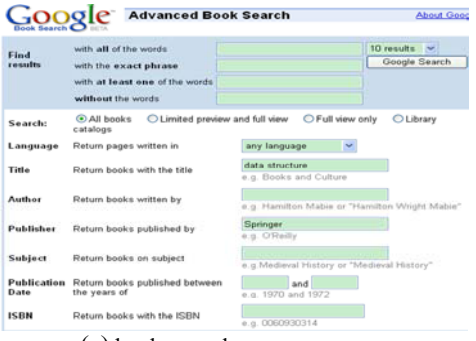
Inspired by approximate search, based on the feature of interface attribute of deep web sources, a top-k data source ranking strategy is proposed to select the best-effort data sources on user's requirement with candidate pruning and early termination. By accessing these selected data sources only, user can obtain satisfied responding results with lower time cost.

For instance, four query-interfaces of deep web sources are shown in Fig. 1, where only parts of attributes (e.g., the dominant attributes such as Title, Author, Publisher and ISBN) are matched to select the best data sources to reduce the matching cost. Moreover, only the best-effort data sources (e.g., data sources (a) and (b) may be the top-2 data sources selected among them if attribute Publisher is one of constraints.) are accessed to meet user's requirement.

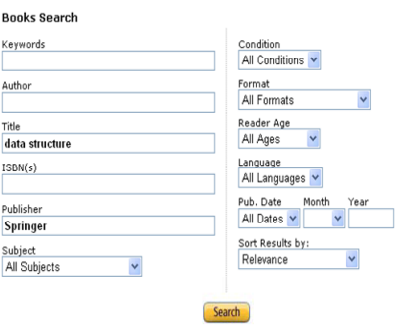
1.2 Problem Statement

Given the deep web data sources $DS = \{ds_1, ds_2, ..., ds_n\}$, and the corresponding data source interfaces $DSI = \{dsi_1, dsi_2, ..., dsi_m\}$, with each dsi specifying the entrance of its data source ds (note here that one data source may have a number of interfaces), and the user's query to the data source interface with a set of attributes $QA = \{A_1, A_2, ..., A_r\}$, where $A_i = \{a_i, op_i, val_i\}$ with a_i specifying the i^{th} attribute, op_i specifying the i^{th} operator (e.g., greater than, between) and val_i being the value of a_i specified by the user.

Our goal is to find data sources that best meet user's requirement, i.e, given the set of data sources DS and user's query QA, we want to find and rank data sources

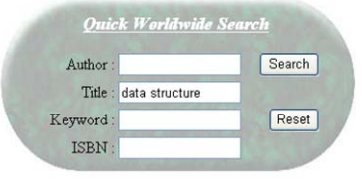


(a) books.google.com



(b) www.amazon.com/

(c) www.bookfinder.com/



(d) www.usedbooksearch.co.uk/

Fig. 1. Interface form samples of deep web sources

according to their relevance to QA in descending order, $DS' = \{ds_1', ds_2', ..., ds_k'\}$, by means of selecting the corresponding interface $DSI' = \{dsi_1', dsi_2', ..., dsi_k'\}$. Note that the number of data sources in DS' and that of interfaces in DSI' are equal since only one query interface of a data source is selected.

1.3 Contributions

The contributions of this paper are threefold: First, as preprocessing, it proposes an attribute based dominant pattern growth (ADP-growth) algorithm to extract the most dominant attributes. Second, an ADP-growth based top-k ranking algorithm (TRA) is employed to rank deep web data sources based on those dominant attributes, which considers the probability of result merging. Further, it incorporates the feature of relevant attributes in relevant data sources and proposes a combined strategy to improve the algorithm. Third, extensive experiments are conducted to verify the effectiveness and efficiency of the proposed strategy and compare it with existing work.

The rest of this paper is organized as follows. We start by a Naïve Top-k data sources ranking strategy in Section 2; Section 3 proposes a ADP-growth based Top-k ranking approach to resolve the deficiencies of it, and Section 4 further improves the strategy with a combined approach. Extensive experiments are conducted in Section 5. Section 6 describes a summary of related work, and finally Section 7 draws the conclusion.

2 A Naïve Top-k Data Sources Ranking

Top-k data sources ranking is useful since retrieving too many data sources may be disturbing and the user is probably only interested in top-k results. Thus, given deep web data sources and user's query with a set of attributes, our goal is to find the data sources that best satisfy the user's query and ranking them.

A naïve approach for top-k data sources ranking is given a user's query, simply matching the attributes of each data source interface to the query, calculating the similarity of each attribute with an IR-style formula and the final scores of the result data sources with a monotonic aggregation function (e.g., summation of the score of each attribute), then ranking the data sources according to the final scores. Suppose a data source ds_k with *m* interfaces, we have formula 1 with notations defined in Section 1.2.

$$Score(ds_k, QA) = \sum_{i=1}^{m} Score(dsi_i) \ .$$
$$Score(dsi_i) = \sum_{j=1}^{r} w_j * Sim(a_j, dsi_i) \tag{1}$$

Where dsi_i is a query interface of ds_k , $Sim(a_j, dsi_i)$ is a similarity function which evaluates the j^{th} attribute of query QA against the matching attribute of the data source interface dsi_i, and returns a value in the range [0,1]. Since each attribute might have a different factor of importance, we use a weight factor w_j (w_j >0) that adjusts the significance of each attribute according to user's preference. $Score(ds_k, QA)$ is the final score of the k^{th} data source with respect to QA.

Note that computing the similarity between each attribute of a query and that of every data source interface directly is inefficient since hardly does any data source interface share the same attributes for a given query and thus computing the similarity or these interfaces is wasteful. To improve the efficiency, an inverted list is created in advance. For each attribute a_j, an inverted list of format $\{dsi_1, ..., dsi_k\}$ is generated and stored, where dsi_k is the identifier of an interface containing a_j. In addition, a hash table is created to locate for each given attribute the storage location of the inverted list of the attribute. These two data structures, namely the inverted list and the hash table, permit efficient calculation of the similarities of all data source interfaces that have positive similarities for any query. For a query with *m* attributes, hash table is used to quickly locate the inverted lists for the attributes. The *m* inverted lists essentially contain all the data needed to calculate the similarity between the query and every data source interface that contains at least one attribute of the query.

This approach is simple but suffers from several deficiencies. First, it has to calculate the similarity of every attribute of a given query, thus the CPU cost is tremendous in large scale integration. Second, it ignores all properties of the attributes on deep web, and thus the ranking result may be imprecise (see Example 1).

Example 1. *Given a query interface for hotels with a set of attributes {Address (Country, City, Street), Beds, Price, Date, Rank}, suppose user Linda wants to search for some information about a hotel and thus specifies and submits her query. Unfortunately, she may get a list of retrieved data sources that some are completely useless*

yet still rank high, e.g., a car rental data source may rank high if some of its attributes has high score though one attribute (e.g., Beds) may score zero since the aggregated final score is still high.

Third, for a query with many attributes, if a source contains many attributes with low relevance, the summation of the final score may be still high though it is actually irrelevant. Therefore, without considering specific properties of attributes and their distribution in a certain domain on deep web, retrieving often too many irrelevant data sources is disturbing. Such problems are practically frequent enough and thus deserve our efforts to be resolved.

3 ADP-Growth Based Top-k Ranking

According to observations in Section 1, we know that some attributes are much more frequent than others, and they are dominant for the selection of data sources. We evaluate the importance of an attribute from three factors, respectively frequency, correlation and preference, and obtain its final score by assigning different weights to each factor, which we will elaborate soon. We call attributes with higher score as dominant attributes. As clarified above, we know that if a source contains many insignificant attributes with low relevance or the scores of some dominant attributes are zero, though its final score may be quite high, it is probably irrelevant. To solve such problem, we propose an attribute based dominant pattern growth (ADP-growth) algorithm to mine dominant attributes for data sources selection, and after acquiring those dominant attributes, we process a top-k style ranking with those attributes for the ranking of relevant data sources.

3.1 ADP-Growth Algorithm

As preliminary, we apply the following definitions similar to [7, 8], with some modifications for the algorithm in our scenario.

Definition 1. Support Degree. *Let* $AS = \{a_1, a_2, ..., a_n\}$ *be the set of total attributes in* DS*, given an attribute set* $X \subseteq AS$*, sup(X) is defined as the number of data sources containing at least one attribute of X in* DS*.*

For simplicity, we normalize *sup(X)* by dividing it to the total number of data sources in DS thus *sup(X)* is in the range of [0, 1].

Definition 2. Correlation. *given two attributes* a_i *and* a_j*, the correlation between* a_i *and* a_j*,* $cor(a_i, a_j)$*, is defined as formula 2:*

$$cor(a_i, a_j) = \sum (f_{a_i} - \overline{a_i})(f_{a_j} - \overline{a_j}) \Big/ (n-1)\sigma_{a_i}\sigma_{a_j} \cdot$$
$$\overline{a_i} = \sum_{i=1}^{n} f_{a_i} \Big/ n \tag{2}$$

$$\sigma_{a_i} = \sqrt{\sum_{i=1}^{n} (f_{a_i} - \overline{a})^2 \Big/ (n-1)}$$

Where f_{a_i} is the frequency of attribute a_i in a data source ds_k, n is the number of data sources in DS, $\overline{a_i}$ is the mean frequency of attribute a_i, while σ_{a_i} is the standard variance of a_i.

For example, in Fig.1, according to the frequency and correlation of attributes on deep web sources, attribute Title is more important than Publisher, and Author and Title have nearly the same importance. While, comparing attribute Author with Title, users usually take more interested in Title, thus preference to an attribute is included for ranking the dominant attributes of deep web in this paper.

User's preference to an attribute a_i is calculated through historical information of user's submitted queries to interfaces since given an interface to the deep web data sources, an user may only specify some of the attributes while leaving others blank. Those specified attributes are required or user preferred attributes.

Definition 3. Preference. *Let* $UAS = \{a_1, a_2, ..., a_s\}$ *be the set of unique attributes in* DS*, and all the attributes in a data source are mapped to ones in UAS, thus the preference,* $pre(a_i)$*, is defined as the percentage of* a_i *to all the specified attributes,*

$$pre(a_i) = \sum_{j=1}^{n} p_{a_i} \Big/ \sum_{k=1}^{s} \sum_{j=1}^{n} p_{a_k} \cdot \tag{3}$$

Where p_{a_i} is the frequency of specified attribute a_i in a data source, n is the number of data sources in DS. For higher efficiency, we update $pre(a_i)$ only after a certain period rather than compute it every time at running time, whose cost is still attributed to preprocessing.

For example, attributes with specified values, such as Title and Publisher in (a) and (b) of Fig.1, are user's preferred attributes.

According to frequency, correlation and preference, the dominant attribute set are generated, details in the following.

Definition 4. Dominance. *For an attribute set* $X \subseteq AS$*, X's dominance* $Dm(X)$ *is obtained by formula 4:*

$$Dm(X) = w_1 * Sup(X) + w_2 * \| \sum_{a_i, a_j \in X} Cor(a_i, a_j) \| + w_3 * \| \sum_{a_i \in X} pre(a_i) \| \cdot \tag{4}$$

where w_1, w_2 and w_3 are different weights that we found 0.68, 0.21, 0.11 respectively work best in our experiments (details omitted for space limitation); $\| \sum_{a_i, a_j \in X} Cor(a_i, a_j) \|$, $\| \sum_{a_i \in X} pre(a_i) \|$ are normalized values in the range [0,1] by dividing to their respective summations.

Definition 5. Maximal Dominant Pattern. *given an attribute set* $X \subseteq AS$, *we say X is a maximal dominant pattern if X's dominance is no less than user predefined minimum threshold min_s, i.e.,* $Dm(X) \geq min\_s$, *and for any other attribute set Y satisfies* $Dm(Y) > min\_s, \forall (Y \subseteq AS \wedge X \subset Y)$.

Therefore, we apply *Dm(X)* value to our ADP-growth algorithm that integrates above three factors contributing to the importance of an attribute, with better precision and recall as verified in our experiments. Given the deep web data sources *DS* and a minimum threshold, the problem of finding the complete close set of dominant attributes is transformed to the maximal dominant pattern mining problem. Similar to FP-growth, we have the following two lemmas.

Lemma 1. *Any proper subset of a dominant pattern can not be a maximal dominant pattern.*

Lemma 2. *The subset of any dominant pattern is still a dominant pattern.*

Thus, we find the most dominant attributes in *DS* in two steps. First, we construct an attribute based dominant pattern tree (ADP-tree) as the compact data structure, which is an extended prefix-tree structure storing the information of dominant patterns. In ADP-tree, each attribute is specified as a node and the dominant items are those dominant attributes. Each node is composed of four parts, with *node_name* and *node_count* specifying the properties of a node, and *node_link* and *node_parent* the two pointers. It creates a dominant item head table *Htable* to travel the tree, including *item_name* and *item_head*, where the *item_head* points to the first node of the same name in ADP-tree. Details of the ADP-tree construction are as follows.

Step1: Scan *DS* once, generate dominant attribute set *FA* and calculate *Dm(FA)*, sort *FA* in descending order of *Dm(FA)* and get dominant attribute list L_{FA};

Step2: Create the root of ADP-tree *T*, labeled as "null";

Step3: For each attribute in *DS*, sort the attribute according to the order of $L_{FA,}$ specified as $\{fa_H; fa_L\}$, where fa_H is the first attribute while fa_L is the list of the remaining attributes;

Step4: While fa_L is not null, call insert_tree ($\{fa_H; fa_L\}$, *T*), i.e., if there is a node *N* with *node_name*= fa_H, then *node_count(N)+1*, else create a new node *N*, with *node_name*=fa_H, *node_count(N)=1*, and linked to its parent node *T* by *node_parent* and linked to the node of the same name by *node_name*.

Initially, only dominant length-1 attributes has nodes in the tree. Second, we propose ADP-growth based on the ADP-tree, which starts from a dominant length-1 pattern as an initial suffix pattern, examines the set of dominant attributes co-occurring with the suffix pattern, constructs its ADP-tree, and performs mining recursively with such a tree. The pattern growth is achieved via concatenation of the suffix pattern with new ones generated from ADP-tree. Since dominant attributes in any data source is always encoded in corresponding path of dominant pattern trees, ADP-growth ensures the completeness of the result, which works as follows.

Algorithm 1. ADPgrowth Algorithm

Input: ADP-tree of DS, min_s

Output: Maximal Dominant Attribute Set of DS, i.e., $MDA(DS, min\_s)$

// initialize dominant attribute set FA, $FA \subset AS$

1: $MDA(DS, min\_s) = \Phi$, $FA = \Phi$;
2: For all $a_i \in AS$ occurring in DS, do
3: $FA = FA \cup \{a_i\}$
4: $DS' = \Phi$, Htable = Φ; /*construct DS' */
5: For all $a_j \in AS$ in DS with $j>i$, do
6: If $Dm(FA \cup \{a_j\}) \geq min\_s$, then
7: $Htable = Htable \cup \{a_j\}$
8: End if
9: End for
10: For all $(fa_H; fa_L) \in DS$, with $fa_H \in FA$ do
11: $DS' = DS' \cup \{FA \cap Htable\}$
12: End for
// depth first recursion
13: Compute $MDA(DS', min\_s)$
14: $MDA(DS, min\_s) = \text{MDA}(DS, min\_s) \cup \text{MDA}(DS', min\_s)$
15: End for

Note here we use a_i instead of A_i, meaning that those attributes do not necessarily have to be assigned a value in this scenario.

3.2 ADP-Growth Based Top-k Ranking Algorithm

ADP-growth algorithm works well in finding dominant attributes. Then we work on those attributes to select relevant data sources. Since given a user's query, our goal is to find the top-k data sources ranked according to the relevance with the query, we can early terminate the query processing if we find the top-k data sources, without exploiting accesses on all the data sources. Therefore, we propose ADP-growth based top-k ranking algorithm (TRA) to speed up the data sources ranking in exploiting large scale deep web data sources.

From above, we obtained sorted list of dominant attributes according to their dominance, $MDA = \{a_1', a_2', ..., a_m'\}$, with scores representing the dominance of each attribute. To rank the data sources, we sort the data sources according to the most dominant attribute a_i' in descending order of its values in each data source, and for each attribute a_i' maintain an inverted list L_i on data sources where we keep a list of ordered pairs $(ds_j, fp(a_i', ds_j))$,

$$fp(a_i', ds_j) = w_1 * freq(a_i', ds_j) + w_2 * pre(a_i', ds_j) \quad . \tag{5}$$

$$freq(a_i', ds_j) = \begin{cases} 1 & score(dsi_i) \geq \theta \\ 0 & otherwise \end{cases}$$

Where $freq(a_i', ds_j)$ is normalized frequency of attribute a_i' in data source ds_j, dsi_i is a query-interface of ds_j, with $score(dsi_i)$ defined as formula 1, θ a specified threshold; $pre(a_i', ds_j)$ is the same as $pre(a_i')$ defined in formula 3. Thus for m unique attributes, n data sources, we generate a dominance matrix $DM=\{fp_{ij}\}_{n\times m}$, with $fp_{ij} = fp(a_i', ds_j)$. Then we process a top-k style access on these data sources, which does not necessarily access all the data sources and hence with higher dominance and less cost. We adopt the idea of probabilistic guarantee for those unvisited data sources in evaluating their probabilistic scores [8],[9], i.e., if the probability of a data source, namely the summation of the evaluated frequency and the upper bound of those unvisited is below a given threshold ε (e.g., set to 0.1), we can discard the data source from the candidate set, where each candidate is a data source ds_i that has been visited in at least one list and may qualify for the final top-k result. Top-k style ranking algorithm adopts the Prob-k approach [9] and mainly concerns about what attribute to probe next so that we can drop some unpromising candidates and minimize the query cost. However, our TRA distinguishes from Prob-k approach in that it considers the possibility of result merging, i.e., it both considers the input interface and output interface of data sources in top-k ranking by join operations between different ones. The join operation is induced when the information of one data source is not sufficient for user's query and thus has to execute join operation with other data sources.

Example 2. *Consider a submitted user interface on automobile {Price, {Make, Manufacturer},{Brand, Model}, Year, Color, Mileage}, and there are some data sources with attributes not sufficient to meet the users' requirement, but they refer to the same information with different type of description,* ds_1*={ID, Price, {Make, Manufacturer},{Brand, Model}, Color },* ds_2*={ID, Price, {Brand, Model}, Mileage},* ds_3*={ID, {Brand, Model}, Year, Color, Mileage}, thus we have to merge the data sources with join operations to satisfy the query.*

Since join operation is expensive and tricky, we should find an optimal plan for data sources scheduling. We do join operation only when the number of ranked data sources is less than user specified *k*. since join operation is expensive, the time cost will increase. Thus, some users may prefer retrieving satisfied data sources for time efficiency, while others may want a better result. Thus, when the number of ranked data sources is less than *k*, we will optionally execute this step based on user's option to proceed or not. When proceeding is required, we adopt a score guided join strategy as in [10], i.e., we first schedule the source with higher score which means it has higher probability to meet the query, and only do join operation if necessary. As verified in Section 5, TRA further improves precision and recall of ranking results. Details of the algorithm are as follows.

Therefore, we find the top-k data sources in two steps. First, we apply ADP-growth algorithm to select the most dominant attributes for matching deep web data source

interfaces. For those selected dominant attributes, we process ADP-growth based top-k ranking algorithm (TRA) to rank the relevant data sources without accessing the whole ones, thus the strategy proposed in this paper is quite efficient.

Table 1. Notations and descriptions for TRA

Notations	Descriptions
MDA	Maximal Dominant Attribute Set;
k	User specified number of results;
$\mathcal{E}$	Threshold for minimal probability bounds;
λ	Threshold for candidates, i.e., the worst score of current k^{th} data source;

Algorithm 2. ADP-growth based Top-k Ranking Algorithm (TRA)

Input: *MDA*, threshold $\mathcal{E}$, *DS* , k

Output: Top-k Data Sources *TDS*

1: $TDS=\Phi$; candidates= Φ ; $\lambda=0$;
2: Repeat
3: For all attribute of *MDA* in index lists L_j do
4: $fp(ds_{ij}) = L_j$.getfp(); // get dominant value
5: $E(ds_i) = E(ds_i) \cup \{ds_i\}$;
6: $high_{ij} = fp(ds_{ij})$;
7: $worstscore(ds_i) = \sum_{ds_i \in E(ds_i)} fp(ds_i)$;
8: $bestscore(ds_i) = worstscore(ds_i) + \sum_{ds_i \notin E(ds_i)} high_{ij}$
9: If ($worstscore(ds_i) > \lambda$) then
10: $TDS=TDS \cup \{ds_i\}$;
11: replace min($worstscore(ds_i') \mid ds_i' \in TDS$);
12: remove ds_i from *candidates*;
13: Else if ($bestscore(ds_i) > \lambda$) then
14: $candidates = candidates \cup \{ds_i\}$;
15: Else
16: drop ds_i from *candidates* if $ds_i \in candidates$;
17: End if
18: End if
19: λ =min{ $worstscore(ds_i') \mid ds_i' \in TDS$};
20: For all $ds_i' \in candidates$ do
21: update $bestscore(ds_i')$ by current $high_{ij}$;
22: If ($bestscore(ds_i') \le \lambda$ or P[$bestscore(ds_i') > \lambda$] $< \mathcal{E}$) then
// with probabilistic pruning*/
23: drop ds_i' from candidates;
24: End if
25: En d for
// less than k data sources satisfying the query

Table 1. (*continued*)

```
26:        If (candidates=Φ or max{ bestscore(ds_i') | ds_i' ∈ candidates} ≤ λ then
                                  Table1.(Continued)
27:          If stop   // user option
28:            return TDS;  //  early termination
29:          If proceed
30:            do JoinOperation( );
31:          End if
32:        End if
33:      End for
34:      Until ||TDS|| = k
```

```
// JoinOperation( )
    For j = i..n, do
      If ds_i.attribute[t] equals to ds_j.attribute[t']
        list[key++]=j;
      End if
    End for
  // join ds_i with other data sources with greedy approach
    rank ds_j in descending order by its attribute[t'] score
    ds_i = Join ( ds_i , ds_j );
```

Here we adopt round-robin schedule for sorted accesses, where $E(ds_i)$ is the set of evaluated list where ds_i has been visited, initially Φ; $high_{ij}$ is the upper bound for the normalized dominant value of unvisited data sources; $P[bestscore(ds_i') > \lambda]$ is the probability that ds_i' has global score above λ.

4 Combined Approach

Kabra et al.[11] observed that the semantics and relationships between deep web sources are self revealing through their query interfaces, that is there are two mutually recursive phenomena: 1) Relevant attributes occur in relevant sources; and 2) Relevant sources contain relevant attributes. Hence, it developed a co-occurrence based attribute graph for capturing the relevance and using them in ranking of the sources in the order of relevance to user's requirement. Inspired by this observation, we propose another co-occurrence based approach to capture the relevant attributes, i.e., we employ mutual information to define the relevance between two attributes (see Formula 6).

$$MI(a_i, a_j) = P(a_i, a_j)\log(P(a_i, a_j) \Big/ (P(a_i)p(a_j))) \ .$$

$$P(a_i, a_j) = C(a_i, a_j) \Big/ \sum_{a_i', a_j'} C(a_i', a_j') \tag{6}$$

$$P(a_i) = C(a_i) \Big/ \sum_{a_i'} C(a_i')$$

Where $C(a_i, a_j)$ is the frequency of co-occurrences of attributes a_i and a_j within a query interface to deep web data sources and $\sum_{a_i', a_j'} C(a_i', a_j')$ is that of all the attributes, $C(a_i)$ is the number of occurrence of a_i in the interface and $\sum_{a_i'} C(a_i')$ is that of all the attributes.

Therefore, we acquire mutual information of two attributes indicate their relevance. Then, we generate a mutual information matrix (*MIM*) indicating the relevance of every two attributes, i.e., for m dominant attributes, $MIM = \{MI_{ij}\}_{m \times m}$, with $MI_{ij} = MI(a_i, a_j)$ specifying the relevance of a_i to a_j. From formula 6, $MI_{ij} = MI_{ji}$, thus we further improve TRA by incorporating both the relevance and dominance feature of attributes in deciding which attribute to probe next to find the relevant data sources. In TRA in Section 3, we have dominance matrix $\mathrm{DM}=\{fp_{ij}\}_{n \times m}$ indicating the importance of each attribute in one data source. Now we develop a combined matrix $CM = \{dr_{ij}\}$, with $dr_{ij} = \sum_{k=1}^{m} (MI_{ik} * fp_{kj})$ considering both the dominance and relevance of an attribute. Thus, we can apply dr_{ij} instead of fp_{ij} to TRA (see Example 3).

Example 3. *For simplicity, we only consider three attributes* $\{a_1, a_2, a_3\}$ *and three data sources* ds_1, ds_2, ds_3. *Suppose we have already obtained dominant matrix DM and mutual information matrix MIM, thus we can generate combined matrix CM, see in the follows:*

$$DM = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{ccc} ds_1 & ds_2 & ds_3 \end{array} \\ \begin{bmatrix} 0.9 & 0.7 & 0.6 \\ 0.7 & 0.8 & 0.4 \\ 0.6 & 0.21 & 0.7 \end{bmatrix} \end{array} \quad MIM = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{ccc} a_1 & a_2 & a_3 \end{array} \\ \begin{bmatrix} 1 & 0.4 & 0.5 \\ 0.4 & 1 & 0.2 \\ 0.5 & 0.2 & 1 \end{bmatrix} \end{array} \quad CM = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{ccc} ds_1 & ds_2 & ds_3 \end{array} \\ \begin{bmatrix} 1.48 & 1.125 & 1.11 \\ 1.18 & 1.122 & 0.78 \\ 1.19 & 0.72 & 1.08 \end{bmatrix} \end{array}$$

If DM is used in TRA algorithm, a_2 *is the second probed attribute then the third one* a_3, *and* ds_1, ds_2 *are the* 1^{th} *and* 2^{th} *ranking results. While, CM, instead of DM, is applied in TRA algorithm, the dominant attribute* a_3 *is the second probed attribute instead of attribute* a_2, *and the ranks are* ds_1, ds_3, *since the mutual information between dominant attribute* a_1 *and* a_3 *are higher than that between* a_1 *and* a_2 *even though the final score of* ds_2 *is higher then that of* ds_3.

The combined approach better improves the efficiency and precision especially in large scale data integration for ranking top-k data sources (See Fig.3 and Fig.4).

5 Experimental Evaluation

For experimental evaluation, we conducted various experiments to evaluate time cost, the precision and recall on the results of top-k data sources with variants of the number of data sources.

5.1 Datasets and Setup

We perform our experiment on the datasets *UIUC Web Integration Repository,* which contains 494 Web query interfaces and totally 370 attributes providing information about diverse domains, viz., airfares, automobiles, books, car rentals, hotels, jobs, movies, and music records and is available on-line [12].

First, we import the data sources we intended for evaluation to an Oracle 10g database. Then, to implement the proposed approaches, we created inverted index lists stored as tables with appropriate indexes to the database. Besides, we generate a hash table for matching the attributes and acquiring the score. All experiments were run on a Pentium® 4 CPU 2.40 GHZ with 512M RAM.

5.2 Comparisons

To quantify the evaluation, we test on the deep web repository with different number of data sources. For better performance comparisons, we evaluate on three metrics, viz., the time cost, precision and recall, where the latter two are the criteria to measure the quality of retrieval. The major findings of our study are: (1) Efficiency of Prob-k and TRA-k approaches. (2) Better quality of ranking results for combined and TRA-k approaches.

Here we distinguish TRA approach from Prob-k by measuring the case when the number of ranked data sources is less than k and user's preference is proceeding for k data sources, thus we denote it as TRA-k in the following figures, and Relevant represents the approach which only considers the co-occurrence of attributes in [11] and does not adopt top-k style approach, while Combine represents the combined approach developed in Section 5.

Fig. 2 shows the time cost of the naïve approach, Prob-k, TRA-k and Relevant. The time cost for TRA-k approach is much less than naïve and relevant approaches for it is executing on the dominant attributes only, and thus we demonstrate that with the ADP-growth algorithm mining the dominant attributes, the time cost is greatly decreased especially when the data sources scale is large. On the other hand, TRA-k is slightly more expensive than Prob-k for the join operation.

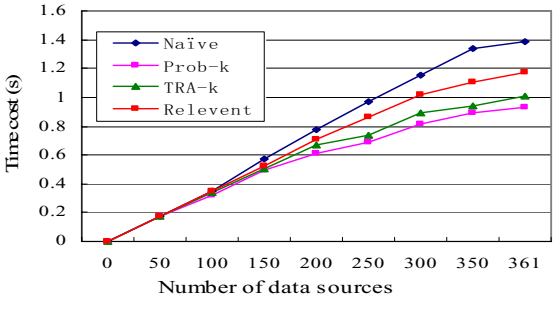


Fig. 2. Time cost of Naïve, Prob-k, TRA-k and Relevant approaches

Fig.3 and Fig. 4 show the precision and recall of the three approaches respectively. We can see that the naïve approach may perform better in precision when the number of data sources is small. However, as the scale of data sources increases, TRA-k approach gradually proves its effectiveness and the combined approach which considers both the dominance and relevance properties of the attributes is the best among five approaches especially the data sources scales.

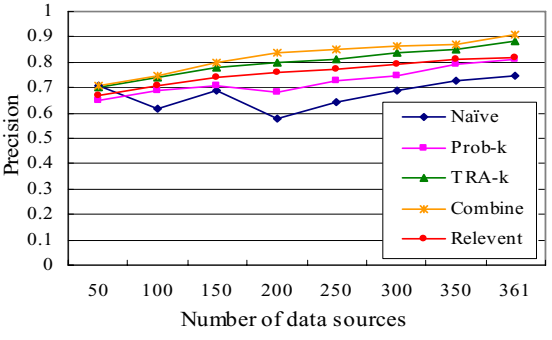


Fig. 3. Precision of Naïve, Prob-k, TRA-k, Combined and Relevant approaches

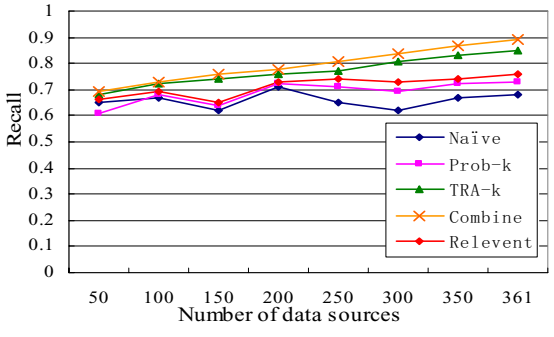


Fig. 4. Recall of Naïve , Prob-k, TRA-k, Combined and Relevant approaches

6 Related Work

A wealth of work has been done in deep web research since the rich amount of data sources has gained great attentions recently. To help users find the relevant data sources and query over them, many ongoing research efforts, e.g., MetaQuerier [13] and WISE [14], have been conducted on the investigation of large scale deep Web data sources. Raghavan et al. [15] proposed HiWe, a task specific crawler for searching deep web while Barbosa et al. [16] presented a new crawling strategy to automatically build a deep web directory for locating the data sources. Ipeirotis et al. [17],[18] classified contents of text databases by sending probing queries to the sources and they developed QProber [19], a system for automatic classification of deep web

databases. Caverlee et al. [20] discovered interesting relationships among deep web databases by probing the target data sources to guide the selection of deep web databases. Shu et al. [21] proposed a method to model the querying capability of a query interface based on the concept of atomic queries. Further, an extensive survey [1] on deep web was conducted to observe characteristics of the sources and study the implications of these characteristics for exploring and integrating those deep web databases.

Besides, top-k retrieval has been gaining importance in information retrieval literature [22]. Ever since Fagin proposed top-k ranking algorithm, the top-k style algorithms have been flourishing. I. F. Ilyas et al. [10] considered top-k join queries in relational database and Chang et al. [23] presented Mpro algorithm for interleaving probing on tuples with substantial cost savings. Marian et al. [24] proposed Upper algorithm by adopting an adaptive per-tuple probe scheduling strategy and further developed a Whirlpool architecture for the adaptive processing of top-k queries in XML. Michel et al proposed a framework for distributed top-k query algorithm-KLEE, which achieves performance gains in terms of network bandwidth, query response times and lighter peer loads at low result-quality penalties [25]. Theobald et al. [8],[9] developed a probabilistic score prediction technique for early candidate pruning when approximate top-k results is acceptable by the user and further proposed a self-tuning incremental query expansion for top-k query processing.

Thus, this paper studies the problems and features for deep web integration and employs a top-k style algorithm to find the relevant data sources.

7 Conclusion

This paper studied the observations on deep web and the resulting problems for data sources ranking. We first employed the attribute based dominant pattern growth algorithm to mining the dominant attributes and proposed several strategies to rank the top-k data sources on deep web with a top-k style ranking algorithm. We further improve our method with a combined approach by considering the relevance property of the attributes, which has proved higher efficiency and precision especially in large scale data integration. We have conducted a comprehensive experiment on the UIUC repository and evaluated and compared the different approaches with three metrics, and thus demonstrated the efficiency and better ranking quality of our approach.

Next, we will study on providing a unified framework for top-k data sources ranking on deep web.

References

1. Chang, K.C.-C., He, B., Li, C., Patel, M., Zhang, Z.: Structured databases on the web: Observations and implications. SIGMOD Record 33(3), 61–70 (2004)
2. BrightPlanet.com. The deep web: Surfacing hidden value, http://brightplanet.com/deepcontent/
3. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On Power-Law Relationships of the Internet Topology. In: Proc. of ACM SIGCOMM, pp. 251–262 (1999)
4. He, B., Chang, K.C.-C.: Statistical schema matching across web query interfaces. In: Proc. of SIGMOD, pp. 217–228. ACM Press, New York (2003)

5. He, B., Chang, K.C.-C., Han, J.: Discovering complex matchings across web query interfaces: A correlation mining approach. In: Proc. of SIGKDD, pp. 147–158. ACM Press, New York (2004)
6. Han, J., Wang, J., Lu, Y., Tzvetkov, P.: Mining Top-k frequent closed patterns without minimum support. In: Proc. of ICDM, pp. 211–218. IEEE Computer Society, New York (2002)
7. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proc. of ACM SIGMOD, pp. 1–12. ACM Press, New York (2000)
8. Theobald, M., Weikum, G., Schenkel, R.: Top-k query evaluation with probabilistic guarantees. In: Proc. of VLDB, pp. 648–659 (2004)
9. Theobald, M., Schenkel, R., Weikum, G.: Efficient and Self-tuning Incremental Query Expansion for Top-k Query Processing. In: Proc. of SIGIR, pp. 242–249. ACM Press, New York (2005)
10. Ilyas, I.F., Aref, W.G., Elmagarmid, A.K.: Supporting Top-k Join Queries in Relational Databases. In: Proc. of VLDB, pp. 754–765 (2003)
11. Kabra, G., Li, C.K., Chang, K.C.-C.: Query Routing: Finding Ways in the Maze of the Deep Web. In: Proc. of WIRI, pp. 64–73. IEEE Computer Society, New York (2005)
12. UIUC Web Integration Repository, `http://eagle.cs.uiuc.edu/metaquerier`
13. Chang, K.C.-C., He, B., Zhang, Z.: Toward large scale integration: Building a metaquerier over databases on the web. In: Proc. of CIDR, pp. 44–55 (2005)
14. He, H., Meng, W., Yu, C., Wu, Z.: Wise-integrator: An automatic integrator of web search interfaces fore-commerce. In: Proceedings of VLDB, pp. 357–368 (2003)
15. Raghavan, S., Garcia-Molina, H.: Crawling the hidden web. In: Proc. of VLDB, pp. 129–138 (2001)
16. Barbosa, L., Freire, J.: Searching for Hidden-Web Databases. In: Proc. of WebDB (2005)
17. Ipeirotis, P.G., Gravano, L.: Distributed search over the hidden-web: Hierarchical sampling and selection. In: Proc. of VLDB, pp. 394–405 (2002)
18. Ipeirotis, P.G., Gravano, L., Sahami, M.: Probe, count, and classify: Categorizing hidden web databases. In: Proc. of ACM SIGMOD, pp. 67–78. ACM Press, New York (2001)
19. Gravano, L., Ipeirotis, P., Sahami, M.: QProber, a system for automatic classification of hidden-web databases. ACM TOIS 21(1), 1–41 (2003)
20. Caverlee, J., Liu, L., Rocco, D.: Discovering Interesting Relationships among Deep Web Databases: A Source-Biased Approach. World Wide Web 9, 585–622 (2006)
21. Shu, L., Meng, W., He, H., Yu, C.: Querying Capability Modeling and Construction of Deep Web Sources. In: Benatallah, B., Casati, F., Georgakopoulos, D., Bartolini, C., Sadiq, W., Godart, C. (eds.) WISE 2007. LNCS, vol. 4831, pp. 13–25. Springer, Heidelberg (2007)
22. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. Journal of Computer and System Sciences 66, 614–656 (2001)
23. Chang, K.C.-C., Hwang, S.-W.: Minimal Probing: Supporting Expensive Predicates for Top-K Queries. In: Proc. of SIGMOD, pp. 346–357. ACM Press, New York (2002)
24. Marian, A., Sihem, A.Y., Nick, K., Divesh, S.: Adaptive Processing of Top-k Queries in XML. In: Proc. of ICDE, pp. 162–173. IEEE Computer Society, New York (2005)
25. Michel, S., Triantafillou, P., Weikum, G.: KLEE: A Framework for Distributed Top-k Query Algorithms. In: Proc. of VLDB, pp. 637–648. ACM Press, New York (2005)

Optimizing Distributed Top-k Queries

Thomas Neumann[1], Matthias Bender[1], Sebastian Michel[2], Ralf Schenkel[1], Peter Triantafillou[3], and Gerhard Weikum[1]

[1] Max-Planck-Institut Informatik, Saarbrücken, Germany
[2] École Polytechnique Fédérale de Lausanne, Switzerland
[3] RACTI and University of Patras, Greece

Abstract. Top-k query processing is a fundamental building block for efficient ranking in a large number of applications. Efficiency is a central issue, especially for distributed settings, when the data is spread across different nodes in a network. This paper introduces novel optimization methods for top-k aggregation queries in such distributed environments that can be applied to all algorithms that fall into the frameworks of the prior TPUT and KLEE methods. The optimizations address 1) hierarchically grouping input lists into top-k operator trees and optimizing the tree structure, and 2) computing data-adaptive scan depths for different input sources. The paper presents comprehensive experiments with two different real-life datasets, using the ns-2 network simulator for a packet-level simulation of a large Internet-style network.

1 Introduction

1.1 Motivation

Top-k query processing is a fundamental cornerstone of multimedia similarity search, ranked retrieval of documents from digital libraries and the Web, preference queries over product catalogs, and many other modern applications. Conceptually, top-k queries can be seen as operator trees that evaluate predicates over one or more tables, perform outer joins to combine multi-table data for the same entities or perform grouping by entities (e.g., by document ids), subsequently aggregate a "goodness" measure such as frequencies or IR-style scores, and finally output the top-k results with regard to this aggregation. Ideally, an efficient query processor would not read the entire input but should rather find ways of early termination when the k best results can be safely determined.

These issues have been intensively researched in recent years (e.g., [7,10,11, 16,19]),and are now fairly well understood for a centralized setting. The current state-of-the-art algorithms for distributed top-k querying [2,4,15,22] address the peculiarities of a distributed setting (in particular communication cost), but fall short of being a perfect solution for really large-scale distributed settings (e.g., highly decentralized and dynamic peer-to-peer systems), where even other performance issues become critical and require different compromises. This paper develops novel techniques to address the peculiarities of such large-scale systems and shows their practical viability.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 337–349, 2008.

Conceptually, the data we consider resides in a (virtual) table that is horizontally partitioned across many nodes in a wide-area network; partitionings are typically along the lines of value ranges, creation dates, or creators. The queries we evaluate on the (virtual) union of all partitions compute the top-k globally most frequent, least frequent, or highest scoring items across the entire network. Further, we assume a monotonic aggregation function such as maximum, minimum, or (weighted) summation. This framework has important real-world applications: (i) Network monitoring over distributed logs [9], e.g., aggregating transferred bytes by IP address or URL; (ii) sensor networks, e.g., aggregating data about air or water pollution by time-of-day or elevation; (iii) mining of social communities and their behavior [8], e.g., aggregating data from social tagging or clickstreams by users.

1.2 Computational Model

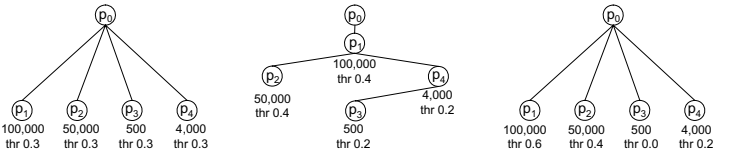
Following [4,15,22], we consider a distributed system with m nodes p_j, $j = 1, ..., m$. We assume that every node can communicate with every other node – possibly with different network costs, but without any limitation of functionality. This can, if necessary, be assured by means of "proxy" nodes. Each node p_j owns a fragment of an abstract relation, containing items I and their corresponding (local) values $v_j(I)$. Such pairs are accessible at each node p_j in sorted order by descending value, i.e., in a (physically or virtually) sorted list L_j. Notice that an item usually appears in the lists of more than one node; often, some popular items appear in the lists of nearly all nodes.

A query $q(k)$, initiated at a node p_{init}, aims at finding the k items with highest aggregated values $V(I) = Aggr_{p_j} v_j(I)$ over all nodes p_j. We use summation for value aggregation throughout the paper, other monotonic functions are supported analogously. Scanning the local list L_j allows each node p_j to retrieve and ship a certain number of its locally highest-value items. The receiving node (e.g., p_{init}) can then employ a threshold algorithm [10,11,16] for value aggregation and determine if previously unseen result candidates potentially qualify for the final top-k result, or if deeper scans or further probings of unknown values are needed to safely eliminate result candidates.

This work sets aside node failures during query execution. In case of temporary node failures or nodes leaving the system, we can adopt the method of [1], which proposes to send partial results directly to the query initiator, or we can apply a re-organization step for the affected portion of the query execution plan.

1.3 Contribution and Outline of the Paper

A standard way of performing distributed top-k aggregation queries is illustrated in Fig. 1 (a), which shows 4 input lists on 4 different nodes and the message flow to a fifth node (p_0) that has posed a top-k query. The 4 lists have different sizes, and we assume that the query processing uses a uniform value threshold of 0.3 for its scan depth. We will later contrast this execution plan with better ones based on our methods.



(a) baseline (b) baseline + hierarchical grouping (c) baseline + adaptive thresholds

Fig. 1. Execution plans illustrating the optimization techniques

To scale up top-k query processing to hundreds of nodes, this paper contributes two novel techniques:

1. The flexible formation of hierarchical groups of node subsets that are considered together. This divide-and-conquer paradigm (cf. Fig. 1 (b)) avoids overly broad top-k aggregation queries that involve too many nodes at the same time and could lead to (incoming) bandwidth bottlenecks at the root of the aggregation. On the other hand, it introduces the combinatorial problem of choosing appropriate groups and forming a tree of cascaded top-k operators (possibly with different k at different stages). We provide exact methods and heuristic approximations for solving this optimization.
2. While previous methods have usually propagated uniform scan depth thresholds to other nodes, we propose an adaptive method for choosing different scan depth thresholds at different nodes, driven by the statistical information about the value distributions in the local lists (cf. Fig. 1 (c)).

The paper presents an extensive evaluation, based on two different real-life datasets and realistic workloads, to demonstrate the scalability of our approaches and their superior performance compared to prior work. The underlying network is simulated by the ns-2 network simulator, a highly detailed and validated model for Internet traffic, widely used in the networking community.

2 Related Work

Top-k query processing has received much attention in a variety of settings such as similarity search on multimedia data [11,16], ranked retrieval on text and semi-structured documents [24,19], network and stream monitoring [4,6], collaborative recommendation and preference queries[5,12], and ranking of SQL-style query results on structured data sources in general [18,21]. Within this rich body of work, the *TA* (threshold algorithm) family for monotonic score aggregation [10,11,16] has proven to be an extremely efficient and highly versatile method.

The first distributed TA-style algorithm for top-k queries over Internet data sources has been proposed by Bruno et al. [14]. It allows both sorted and random access to input lists but tries to avoid random accesses depending on access

costs and limitations of the data sources. Scheduling strategies for random accesses to expensive data sources were also addressed in [5] for a setting with centralized sorted accesses. Zhang and Suel [24] consider distributed variants of TA with sorted accesses only, continuously sending parts of the lists between the network nodes until the top-k answers have been found, where the number of communication steps is limited only by the size of the shortest list.

In contrast, state-of-the-art algorithms for distributed top-k aggregation use a fixed number of communication rounds to bound latency and aim to minimize the total bandwidth consumption. The first algorithm in this family is *TPUT* (Three-Phase Uniform Threshold) [4], where a coordinator, typically the query initiator, executes a 3-phase distributed threshold algorithm as follows:

- Phase 1: Retrieve the top-k list entries from each of m network nodes and compute the k-th largest aggregated value of these items ($mink$), assuming a value of 0 for all unknown values.
- Phase 2: Revisit all m nodes and ask for all list items with $value > mink/m$, recompute $mink$, and eliminate candidates which cannot qualify anymore for the global top-k items.
- Phase 3: Retrieve all missing values for the remaining top-k candidate items by random accesses to the input lists where the items have not yet been seen.

TPAT [22] is a modification of TPUT where the $mink/m$ threshold is adapted to the specifics of the value distributions; however, the authors state that their solution may incur very high computational cost.

KLEE [15] is a framework for distributed top-k processing that utilizes a combination of histograms and Bloom filters to reduce the communication costs of TPUT-style algorithms. When a node is requested to return its locally best items, it piggybacks a histogram of the local value distribution and also Bloom filters as compact synopses of the items for each of the top-c histogram cells or groups of consecutive cells (where c is a tunable parameter). The receiver of these synopses can combine the Bloom filters from different network nodes for an approximate aggregation of values. The additional information obtained from the per-cell synopses often allows the query processor to derive a higher $mink$ threshold than TPUT would have; the subsequent round(s) of retrieving all list entries with value above $mink/m$ is more restrictive and can save communication as well as processing costs.

Specific network topologies are considered in [2,23], leading to optimizations for hypercube or tree topologies. Unlike these approaches, TPUT and KLEE have been designed for general networks without any assumptions on network topology. In a previous paper [17] we studied the hierarchical execution of top-k queries by formulating the query using algebraic operators. This provided a first step towards optimizing top-k queries, but was limited to TPUT queries without threshold tuning. The optimization techniques introduced in this paper are applicable to any algorithm of the TPUT or KLEE families.

3 Cost Prediction

All cost-based optimizations rely on cost predictions to decide which execution strategy is preferable. We now discuss the two prediction primitives required for our top-k optimization. First, we must predict the number of items in a list with a value above a threshold, or similarly the value of the item at a certain position in a list. Second, we must predict the cost of the resulting network transfers, as they affect the observable runtime.

3.1 Estimating Item Counts and Values

Consider a top-k query over m input lists L_i, $i = 1, \ldots, m$, spread across m nodes. If the threshold for each list is known, we can easily estimate the number of items in the local range scan at each node by employing data distribution synopses (our implementation uses histograms) for the local values of each node. Similarly, we can estimate the value of an item at position d_i in list L_i from such synopses. The number of distinct items read from m' lists, $1 < m' \leq m$, given thresholds for each list, can be computed from the convolution of the synopses of these lists. This assumes stochastic independence between different lists, a postulate commonly made for tractability. Although the assumption rarely holds in practice, models based on independence have been very successful for many prediction tasks and applications that require such statistical reasoning.

3.2 Estimating the Network Costs

The basic network primitive used by the algorithms are 1:n transfers: one node requests data from n other nodes, first sending the request and then collecting the data. The cost of such a 1:n transfer is estimated using the latency $Latency[i, j]$ between two nodes, the maximum bandwidth $Bandwidth[i, j]$ between two nodes, and the effective bandwidth $EffBandwidth[n]$ in a 1:n communication. (one node sending messages to its children or all children replying to their parent, in a bursty manner). The last parameter models the TCP protocol overhead, as the theoretical bandwidth is usually not achievable. These parameters can be estimated or determined empirically for a given network. The costs are determined by the highest latency and the total transfer time according to the effectively available bandwidth (see Figure 2). For entire query trees, the total costs are computed by summing over the slowest path in the tree. In our experiments, the predicted costs were usually within 10% of the real costs.

We determined it empirically by running repeated transfer experiments with the ns-2 network simulator; it could be measured the same way in a real system.

4 Hierarchical Grouping and Its Optimization

TPUT and KLEE employ a flat execution strategy similar to Fig. 1 (a): all nodes send their items directly to the query initiator. This execution model is

networkCosts(O,$T = \{T_1, \ldots, T_n\}$)
Input: a peer O; bytes T_i transferred between O and i
Output: the estimated network cost
1 $l = 0$; $b = 0$
2 **for each** $T_i \in T$
3 $l = \max(l, 2 * Latency[O, i])$
4 $b = b + \frac{T_i}{\min(Bandwidth[O,i], EffBandwidth[|T|])}$
5 **return** $l + b$

Fig. 2. Cost Computation for 1:n Transfers

wasteful for a number of reasons. First, it incurs unnecessary communication. For example, consider a query with one very large and several small input lists residing on different nodes. It would be better to perform the top-k query at the node with the large list, have the small nodes ship their items to that node, and only send the final result to the initiator. Second, the nodes compete for network bandwidth, as all of them send their items to the initiator forming the top-k aggregation. If instead several nodes aggregated data from other nodes and only sent their aggregated results to the querying node, the total bandwidth consumption could be reduced.

We apply a hierarchical grouping of nodes in the second phase of these algorithms to reduce transfer costs. Figure 1 (b) illustrates an example execution plan for a query with $m = 4$ input lists L_1 through L_4 on 4 different nodes p_1 through p_4. Instead of querying all 4 nodes for their local items with value above the threshold $mink/4$, the query initiator contacts only node p_1, which itself contacts p_2 and p_4 with a threshold of $mink/3$ (the last third of the threshold remains at p_1). Node p_4 subsequently forwards the request to its children in the execution plan, again dividing the threshold by the respective number of children. For node p_4, the new threshold is $mink/(3 * 2)$, as p_4 has 2 children, including the (local) list L_4. Note that the threshold for the relatively large node p_2 is higher than the threshold in a flat execution, $mink/4$, reducing the number of items sent. When p_4 has received all items from its children, it aggregates them with the items of its own list and sends the result to its parent in the execution plan, p_1. As some items may occur in multiple lists, the number of items sent to p_1 is typically less than in a flat execution.

Using such a hierarchical grouping can improve the query execution, but, depending on the sizes and value distributions of the input lists, may also adversely affect performance by adding latency and transfer cost (as data must pass through more than one node). Therefore, the hierarchical grouping must be constructed by a query optimizer that computes the cost of the candidate trees and chooses the best alternative. The cost of a candidate tree refers to its total execution time, which in our model is dominated by the bandwidth-delimited data transfer times and additional network latencies.

4.1 Dynamic Programming Approach

One way to find the optimal hierarchical structure is to employ dynamic programming (DP). Note that we only optimize the second phase of the algorithms, so the *mink* threshold is already known in advance and we only have to organize the aggregation of items. The cost of each aggregation step is determined by the costs of its slowest input (*max*) and the bandwidth limitations for getting the input data to the aggregating node (basically a weighted *sum*). This allows for the following theorem:

Theorem: The optimal solution of a hierarchical grouping problem can be constructed from optimal solutions for its subproblems (i.e., execution trees for subqueries). □

Proof (sketch): Consider an optimal solution to a problem that consists of a non-optimal solution to a subproblem. If we replaced the non-optimal solution to the subproblem by an optimal one, the resulting solution to the overall problem would have the same cost (in this case we do not care) or it would be cheaper than the original solution which would contradict the assumption. □

Fig. 3 shows the optimization algorithm in pseudo-code. The algorithm applies DP in a top-down formulation with memoization. The DP table maps $(lists, mink) \rightarrow (node \rightarrow plan)$, i.e., for each combination of input lists and *mink* threshold, we keep the optimal plan for each possible target node where the subquery result could reside. In our distributed setting, the placement of data also has to be taken into account. This leads to the following optimization process: the algorithm always considers all possible nodes as location for the result, i.e., it operates on sets of plans – one plan for each possible node where the final result could reside. A (sub-)problem can always be solved by using a flat execution, i.e., aggregating the input nodes at the target (lines 3-4). If the problem consists of more than one input node, the aggregation can instead be performed hierarchically: the problem is split into smaller problems whose results are then combined (lines 6-11). As it might be better to perform the entire aggregation at one node and merely ship the results, the algorithm considers the cost of this case (lines 12-13). For the transfer cost, the number of items transferred from a group of nodes to their parent is estimated using the predictions of Section 3.

The DP algorithm can be implemented with an upper bound of $O(m4^m)$, but $\Omega(2^m)$ is a lower bound which makes using DP infeasible for large m. For large m, we use a faster heuristics instead of the exact DP.

4.2 Fast Heuristics

The hierarchical structure is a divide-and-conquer strategy for the aggregation; therefore, we want to partition the lists such that the resulting partitions exhibit approximately equal costs. In our cost model, lists with similar cardinality will cause similar effort; so we heuristically partition the list L of all data-item lists as follows:

```
buildHierarchy(L,mink)
Input: set L of all data-item lists; value threshold mink
Output: set of optimal execution plans, one for each node
1  if (I, mink) has already been solved then return known solution
2  b = empty plan set
3  for each p ∈ nodes
4    b[p] = flat aggregation of L at p, threshold mink
5  if |L| > 1 then
6    for each P = {L_i ⊂ L}, P partitioning of L
7      L' = {buildHierarchy(L_i, mink/|P|)|L_i ∈ P}
8      for each p ∈ nodes
9        L_p = {i[p]|i ∈ L'}
10       a =aggregation of L_p at p
11       if a.costs< b[p].costs then b[p] = a
12   for each p_1, p_2 ∈ nodes
13     if transfer(b[p_1], p_2).costs< b[p_2].costs then b[p_2] =transfer(b[p_1], p_2)
14 store b as solution for (L, mink) in DP table
15 return b
```

Fig. 3. DP Algorithm for Optimal Grouping

- S_L : L sorted by cardinality of items above $mink/|L|$
- O_L : every "odd" list of S_L $(L_1, L_3, ...)$ (sorted by desc. cardinality)
- E_L : every "even" list of S_L $(L_2, L_4, ...)$ (sorted by asc. cardinality)

We expect that O_L and E_L are similar, so O_L and E_L would already be a good partitioning, but the cardinalities can vary widely. Therefore, we consider moving some of the shorter lists (tail of O_L, head of E_L) from one partition to another. We concatenate O_L and E_L (which are sorted reversely), and cut the resulting list at any position to get partitioning candidates. The resulting search space is no longer exponential, allowing for an implementation in $O(m^2)$ using search space pruning. This heuristics works very well in practice and allows very fast construction of competitive execution trees even for large numbers of input lists.

5 Adaptive Thresholds

After determining an initial *mink* threshold, both TPUT and KLEE in their second phase request all data items that may qualify for the top-k result. They conservatively distribute the necessary value mass uniformly over all input lists and request all items with a local value above $mink/m$ (cf. Fig. 1 (a)). However, as value distributions vary widely across lists, data-adaptive thresholds that are specifically tuned to the individual lists (cf. Fig. 1 (c)) are promising and were already considered in [22], but deemed computationally intractable and not pursued much further. Our approach chooses adaptive thresholds by first choosing scan depths and then deriving appropriate value thresholds.

We formally define this optimization problem as follows. Assume that we scan the m input lists to depths $d_1, d_2, \ldots, d_m$, and the values at these list positions

are $v(d_1)$, $v(d_2)$, ..., $v(d_m)$, respectively. We need to ensure that we scan deep enough so as not to miss any potential top-k candidate; this mandates the constraint $\sum_{i=1}^{m} v(d_i) \leq mink$, with uniform thresholding being a special case. We aim to minimize the total cost of shipping list entries, which is equivalent to minimizing $\sum_{i=1}^{m} d_i$, subject to the introduced constraint. For given scan depths d_i we can estimate the resulting $v(d_i)$ (see Section 3).

This problem is NP-hard, as we can reduce the Knapsack problem to our problem, so an exact solution is out of the question as we address applications with large m. However, we can devise practically good approximations based on the following heuristics. We optimize the maximum scan depth over the m lists (instead of the sum of the scan depths). In a lightly loaded network with all m scans proceeding in parallel on different nodess, this objective function would be appropriate for minimizing the latency of this phase. For our actual objective function, minimizing the total network costs, it is merely a heuristics, but turns out to be a fairly good approximation. If we minimize the deepest scan, i.e., $max_{i=1}^{m} d_i$, we can set all d_i to the same maximum, so that we effectively deal with only one free variable as $d_1 = d_2 = ... = d_m$. We still need to ensure that this choice of d_i satisfies the constraint. We can perform a binary search over the possible choices to find the lowest d_i without violating the constraint. Note that this approach of uniform scan depths usually results in non-uniform local thresholds at which the scans on the individual lists stop.

6 Experiments

6.1 Setup

We have implemented all algorithms and our testbed in C++. To obtain reproducible and comparable results, we simulate the network; running experiments over multiple nodes in a real-world network such as PlanetLab would suffer from unpredictable and unreproducible interference by other applications and network nodes. As a simulation environment we used the ns-2 network simulator [13], which is a state-of-the-art packet level network simulator. We used the Inet-3.0 topology generator [20] to create an Internet-style topology with 3037 nodes and bandwidths ranging from 1 MBit/s in the leaves to 10 GBit/s in the backbone. For the experiments, we assigned the individual data lists to random nodes, and employed the network simulator to compute the execution time for each algorithm. All results are averages over 10 random placements.

Algorithms under Comparison. Our experimental evaluation focuses on the following combinations of our techniques with the existing algorithms: **TPUT** is the three-phase uniform threshold algorithm [4]. We do not consider the variant of TPUT that uses a compression technique based on hash array encoding to decrease the network bandwidth consumption. We consider it an orthogonal issue to apply compression techniques to any of the investigated algorithms. **KLEE** is an extension of TPUT that employs histograms and Bloom filters [15]; we set KLEE's tuning parameter as $c = 5\%$. We use only the three-phase KLEE variant, and disregard

the KLEE-4 variant of [15] as its additional filtering step would be orthogonal to the issues studied here. **AdaptiveTPUT** is an extension of TPUT that uses our adaptive thresholding described in Section 5. **AdaptiveKLEE** is the equivalent extension of KLEE. **TreeTPUT** uses hierarchical query execution plans introduced in Section 4, in addition to the adaptive-threshold technique. In the experiments, we use the fast heuristics. **TreeKLEE** is the equivalent extension of KLEE.

Approximate vs. Exact Mode. KLEE has explicitly been designed as an approximate algorithm [15]. However, it can be turned into an exact algorithm by adding an additional random-lookup phase at the end. The resulting algorithms can be considered as TPUT variants flavored with KLEE's techniques plus our optimization techniques. On the other hand TPUT has been designed as an exact algorithm [4], but can be transformed into an approximate algorithm by skipping the random-lookup phase at the end. In our experiments we study TPUT and KLEE both in exact and in approximate mode.

Datasets. The **WorldCup** HTTP server log collection[1] consists of about 1.3 billion HTTP requests recorded during the 1998 FIFA soccer world cup. The data is provided as 249 individual access logs. We constructed 249 nodes by converting each access log into a node. The task is to identify the top-100 clients that caused the most traffic on a given set of nodes. The **Retail Benchmark** consists of retail market basket [3]. A set of 100 nodes was generated by randomly assigning each of the ∼88k transactions to exactly one node, modeling a situation in which the transactions had occurred at distributed shopping sites. At each node, we generated all possible triplets of basket items present in any of the transactions, yielding a total number of $51,788,094$ ($16,769,821$ distinct) triplets. As for queries, we are interested in finding the globally most frequent triplets, using only a subset of the 100 nodes (i.e., retail stores).

Metrics. We consider the following metrics: (1) **Query response time**, the "wall-clock" time for the benchmarks in the network simulation, and (2) **Relative recall**, the overlap between the top-k results produced by approximate algorithms and the true top-k results produced by an exact method.

6.2 Results

Fig. 4 (left) shows average response times for the Retail benchmark for different query sizes (number of nodes queried) in exact mode. Each point in the chart is computed by averaging over 10 independently chosen random queries. For all queries, TPUT is improved by AdaptiveTPUT and further improved by TreeTPUT. Interestingly, KLEE performs worse than TPUT, and AdaptiveKLEE performs slightly worse than KLEE for query size 20. This is caused by additional random lookups in Phase 3: while KLEE retrieves less data items in Phase 2, it requires more random lookups in Phase 3, which are quite expensive. For query size 20, the scan depth balancing aggravates this by reading even less data items

[1] *http://ita.ee.lbl.gov/html/contrib/WorldCup.html*

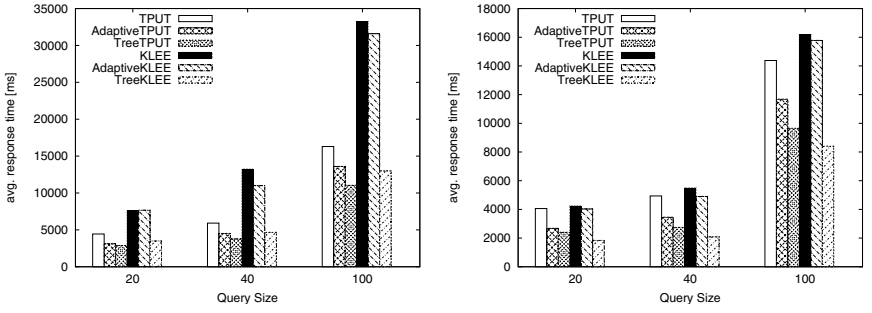


Fig. 4. Retail results in exact (left) and approximate (right) mode

and thus requiring more random lookups. For larger query sizes 40 and 100 the benefit of adaptive scan depths outweighs the additional random lookups. The TreeKLEE variant performs much better than KLEE and AdaptiveKLEE, but is still slower than TreeTPUT.

Fig. 4 (right) illustrates average response times in approximate mode; Table 1 shows relative recall for the largest queries. The differences between TPUT and KLEE are smaller here, but still KLEE is slower due to its relative expensive Phase 1 communication. TreeKLEE can make use of the improved thresholds and performs better than TreeTPUT. AdaptiveKLEE is only a minor improvement over KLEE here, while TreeKLEE improves the runtime of KLEE up to a factor of 2. For TPUT the adaptive scan depths have a much larger impact, with AdaptiveTPUT performing nearly as good as TreeTPUT for small queries.

Fig. 5 illustrates average response times for the WorldCup benchmark. In exact mode (Fig. 5 (left)), AdaptiveTPUT/AdaptiveKLEE improve the average response time only slightly over TPUT and KLEE, whereas TreeTPUT and TreeKLEE improve the run-time up to a factor of 4. Table 1 shows relative recall for different numbers of nodes in approximate mode. Here, the run-time effects (Fig. 5 (right)) are similar to the exact case.

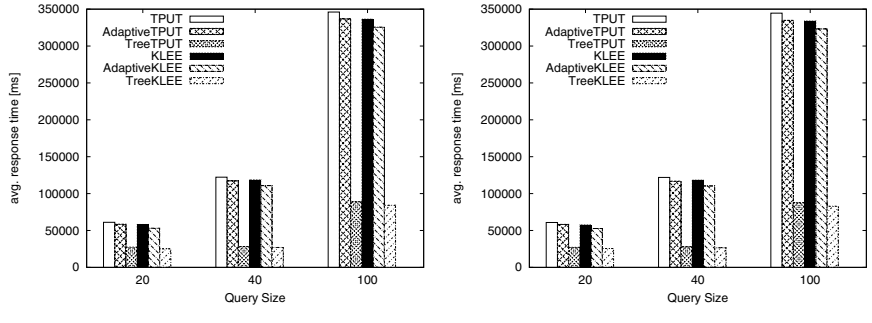


Fig. 5. Worldcup results in exact (left) and approximate (right) mode

Table 1. Recall results in approximate mode

Dataset	#nodes	TPUT	Adaptive TPUT	Tree TPUT	KLEE	Adaptive KLEE	Tree KLEE
Retail	40	0.98	0.97	0.97	0.92	0.90	0.90
Retail	100	0.98	0.96	0.96	0.92	0.90	0.90
Worldcup	40	0.98	0.96	0.95	0.97	0.95	0.94
Worldcup	100	0.99	0.97	0.96	0.98	0.96	0.95

6.3 Discussion

Overall, TreeTPUT/TreeKLEE are the best-performing and most robust algorithms. They are superior to all competitors in all cases, and significantly outperform the base algorithms with run-time gains up to a factor of 4. In exact mode, TreeTPUT is slightly preferable to TreeKLEE (which is not surprising, as KLEE was designed as an approximate algorithm). In approximate mode, TreeKLEE performs better than TreeTPUT and is the algorithm of choice. The optimizations themselves have a greater impact than the choice of the base algorithm: while TPUT and KLEE perform quite differently, TreeTPUT and TreeKLEE are much closer to each other. Optimizing only the scan depths in AdaptiveTPUT/AdaptiveKLEE already improves the run-times, but the full cost-based optimizations of TreeTPUT and TreeKLEE give much better results and are essential for consistently good performance.

Although the issue of exact vs. approximate results is orthogonal to the contributions of this paper, we think it is worthwhile pointing out that the approximate variant of TreeKLEE is a particularly intriguing algorithm for many practical applications. It is often a factor of 2 faster than its exact counterpart, but consistently achieves a relative recall above 90% – an excellent result quality that would be perfectly acceptable for most applications of top-k querying.

7 Conclusion and Future Work

This paper has developed and experimentally studied novel techniques for optimizing top-k aggregation queries that involve many peers in a wide-area network. Each of our main techniques can individually improve the performance of the state-of-the-art algorithms, TPUT and KLEE. Together, our techniques exhibit additional synergies and consistently outperform prior methods. Our future work will aim to eliminate the few limitations that our methods have: i) considering correlation information for the underlying peers and their value distributions in the statistical predictor models, and ii) looking for better approximation techniques for our hierarchical grouping and adaptive thresholding methods.

References

1. Akbarinia, R., et al.: Reducing network traffic in unstructured p2p systems using top-k queries. Distributed and Parallel Databases 19(2-3), 67–86 (2006)
2. Balke, W.-T., Nejdl, W., Siberski, W., Thaden, U.: Progressive distributed top k retrieval in peer-to-peer networks. In: ICDE, pp. 174–185 (2005)

3. Brijs, T., Swinnen, G., Vanhoof, K., Wets, G.: Using association rules for product assortment decisions: A case study. In: KDD, pp. 254–260 (1999)
4. Cao, P., Wang, Z.: Efficient top-k query calculation in distributed networks. In: PODC, pp. 206–215 (2004)
5. Chang, K.C.-C., won Hwang, S.: Minimal probing: supporting expensive predicates for top-k queries. In: SIGMOD, pp. 346–357 (2002)
6. Das, G., Gunopulos, D., Koudas, N., Sarkas, N.: Ad-hoc top-k query answering for data streams. In: VLDB, pp. 183–194 (2007)
7. Das, G., Gunopulos, D., Koudas, N., Tsirogiannis, D.: Answering top-k queries using views. In: VLDB, pp. 451–462 (2006)
8. Dubinko, M., Kumar, R., Magnani, J., Novak, J., Raghavan, P., Tomkins, A.: Visualizing tags over time. In: WWW, pp. 193–202 (2006)
9. Garofalakis, M.(ed.): Special issue on in-network query processing. IEEE Data Eng. Bull. 28(1) (2005)
10. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. J. Comput. Syst. Sci. 66(4), 614–656 (2003)
11. Güntzer, U., Balke, W.-T., Kießling, W.: Optimizing multi-feature queries for image databases. In: VLDB, pp. 419–428 (2000)
12. Güntzer, U., Balke, W.-T., Kießling, W.: Towards efficient multi-feature queries in heterogeneous environments. In: ITCC, pp. 622–628 (2001)
13. Information Sciences Institute. The University of Southern California. The network simulator - ns-2 (2007), `http://www.isi.edu/nsnam/ns/`
14. Marian, A., Bruno, N., Gravano, L.: Evaluating top-k queries over web-accessible databases. ACM Trans. Database Syst. 29(2), 319–362 (2004)
15. Michel, S., Triantafillou, P., Weikum, G.: Klee: A framework for distributed top-k query algorithms. In: VLDB, pp. 637–648 (2005)
16. Nepal, S., Ramakrishna, M.V.: Query processing issues in image (multimedia) databases. In: ICDE, pp. 22–29 (1999)
17. Neumann, T., Michel, S.: Algebraic query optimization for distributed top-k queries. In: BTW, pp. 324–343 (2007)
18. Soliman, M.A., Ilyas, I.F., Chang, K.C.-C.: Top-k query processing in uncertain databases. In: ICDE, pp. 896–905 (2007)
19. Theobald, M., Weikum, G., Schenkel, R.: Top-k query evaluation with probabilistic guarantees. In: VLDB, pp. 648–659 (2004)
20. Winick, J., Jamin, S.: Inet-3.0: Internet topology generator. Technical Report UM-CSE-TR-456-02, EECS, University of Michigan (2002)
21. Xin, D., Han, J., Chang, K.C.-C.: Progressive and selective merge: computing top-k with ad-hoc ranking functions. In: SIGMOD, pp. 103–114 (2007)
22. Yu, H., Li, H.-G., Wu, P., Agrawal, D., Abbadi, A.E.: Efficient processing of distributed top-k queries. In: Andersen, K.V., Debenham, J., Wagner, R. (eds.) DEXA 2005. LNCS, vol. 3588, pp. 65–74. Springer, Heidelberg (2005)
23. Zeinalipour-Yazti, D., et al.: The threshold join algorithm for top-k queries in distributed sensor networks. In: DMSN (2005)
24. Zhang, J., Suel, T.: Efficient query evaluation on large textual collections in a peer-to-peer environment. In: Peer-to-Peer Computing, pp. 225–233 (2005)

POEMS: Peer-Based Overload Management

Wee Siong Ng[1,2], Panos Kalnis[2], Kian-Lee Tan[2], and Markus Kirchberg[1]

[1] Institute for Infocomm Research (I^{2}R),
Agency for Science, Technology and Research (A*STAR), Singapore
{wsng,mkirchberg}@i2r.a-star.edu.sg
[2] School of Computing, National University of Singapore, Singapore
{kalnis,tankl}@comp.nus.edu.sg

Abstract. The Internet has become increasingly important to many emerging application such as Blog, Wikis, podcasts, and others Web-based communities and social-networking services, i.e. Web 2.0. Behind the scenes, added functionalities depend on the ability of users to work with the data stored on servers, i.e. DBMSs. However, the unpredictability and fluctuations of requests could result in overload, which can substantially degrade the quality of service. It is a challenging task to provide quality of service with inexpensive and scalable infrastructure. In this paper, we look at a new architectural design dimension, POEMS, that is online transformable between a single-node server and peer-based service network architectures. POEMS operates as a conventional DBMS under normal load conditions and transforms to peer-to-peer operation mode for processing under heavy load. In contrast to traditional distributed DBMSs, all nodes contribute their spare capacities for data manipulation. This is achieved without the need to install any DBMS at any of the contributing nodes. Data are partitioned online and operators are distributed to nodes similarly. The effectiveness of query processing is achieved by node cooperation. POEMS allows processes or operators to be dismissed online, so a user can fully utilise his/her resources.

1 Introduction

We have observed an evolution in Web technology in the past few years. Unlike the 1^{st} generation Web regarding the Web as information source, Web 2.0 provides architecture of participation that encourages user contribution. Consequently, companies such as YouTube and Flickr have been very successful by providing 'peer production' solutions to meet the demands of today's users. However, the quality of service has also become much more crucial. Any unpleasant experience may push users away to those who provide better services.

There are many aspects in providing pleasant user-experiences, responsiveness is one of the criteria. In a typical multi-tier Web application, users' data is stored in a DBMS. In order to prevent a bottleneck at the data tier, many organizations and researchers are heading toward distributed database technology. Significant advances have taken place in the development and deployment of distributed DBMSs (DDMSs). These include mechanisms to provide transparency

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 350–365, 2008.

in accessing data from multiple servers [19], and the support of distributed transactions over fragmented and heterogeneous data sources [8,15].

In this paper, we investigate a common and practical problem: Imagine a typical business environment where a medium-size organization operates with hundreds of office personal computers (PCs) and a central DBMS. Most of the office PCs are used mainly for simple word-processing and emailing, or as dumb terminals with no DBMS facilities installed at all. The central DBMS receives requests internally or externally through an Internet connection. The unpredictability and fluctuations of the requests may overload the DBMS[1]. Conventional DDMSs tackle overload by introducing additional servers to handle the extra load. However, besides the additional costs, such approaches are not flexible. The motivation of our work is that we can exploit the spare capacity of resources of office PCs, in order to assist the DBMS to maintain its performance under fluctuating overload conditions. The intuition is to distribute some of the DBMS load to the PCs during the peak periods, while not interrupting them when the DBMS's workload is normal.

Peer-to-Peer (P2P) technology provides an attractive alternative for building distributed systems. The P2P environment is dynamic and sometimes ad hoc resulting in an evolving architecture where each peer is fully autonomous. A considerable amount of research has been conducted on data integration [7,10], data mapping [5,6], data discovery and query processing [13] in P2P environments. These systems assume that all participating peers have a DBMS installed and are willing to share their data with other peers. Moreover, each peer is expected to play the role of data provider and data consumer.

In contrast to DDMSs, peer-based DBMSs offer a more cost-effective solution where each peer has a DBMS installed and load or data is distributed in order to maintain system performance. The main concern here is not additional costs, but rather the complexity of maintaining the DBMS at each peer site. This involves several difficulties especially for novice users. First, a complex DBMS would affect the ad hoc tasks running on a typical PC. Second, there is the issue of the complexity of inter-connectivity and integration between one DBMS and others. Third, introducing new functions involves the upgrade of all the participants. Furthermore, both DDMSs and peer-based DBMSs suffer from the drawback of not being capable to handle dynamically changing user requests (due to the initial data placement not being able to guarantee good load balancing for different access patterns in the future).

1.1 Our Proposal

Most existing systems employ either a client-server (CS) or P2P-based architecture. The former provides easy access to resources, data control, and integration of new technology. While these features are absent in P2P systems, they provide scalability in harnessing processing power for solving a given task.

[1] Many community Web sites, in particular those with a regional focus, experience overload during common break times and at the beginning / end of a working day.

We propose *POEMS (Peer-based OvErload Management System)*, a novel approach that handles the above-mentioned problems and inherits the advantages of both CS and P2P systems. We discuss an autonomic DBMS architecture with two operation modes: A centralized and a P2P mode. Under normal conditions, the DBMS operates in CS mode without interrupting other peers. The administrator defines a performance threshold, e.g. based on transaction throughput. On overloading (i.e. exceeding the given threshold), the DBMS seeks ways to improve its performance by means of best-effort, especially when the overload is caused by short and swiftly fluctuating requests. In such a situation, the DBMS transforms its query processing operation into P2P mode and harnesses more power from the peers to assist the DBMS during the peak period.

When POEMS operates in a P2P mode, it treats each peer as an autonomic element (AE) [11] that will contribute only its processing power and memory resources as services without any DBMS capabilities (e.g. similar to the SETI@Home project). Operators have to be shifted on demand to AEs based on the query's execution plan[2]. POEMS interacts constantly with AEs to manage their current resource status. When the system becomes overloaded, a query optimizer partitions the data considering the available AEs and the amount of underutilized resources. A root operator that consists of all the optimized sub-operators (e.g. scan, selection, join, and projection) will be generated based on the local query execution plan and sent together with the partitioned data to the AEs for processing. All processing to be carried out at an AE must be main memory based in order to reduce the effects (i.e. I/O operations) of interruption on the existing tasks running at the AE. Intuitively, data shifting may incur high communication overhead, but this concern can be easily resolved: By moving only selected operators to remote peers to cooperate with the existing operators, peers can process subsequent queries effectively without any further data shifting from the DBMS (i.e. PUSH-based prefetching).

For example, let us consider 3 queries on the `CUSTOMER` and `CART` relations:

```
(a) SELECT CUSTOMER.name, CUSTOMER.address FROM CUSTOMER
    WHERE CUSTOMER.cid > 1000
(b) SELECT CART.cartid, CART.status FROM CART
(c) SELECT CUSTOMER.cid, CUSTOMER.firstname, CUSTOMER.sex, CART.cartid,
    CART.status FROM CUSTOMER, CART WHERE CUSTOMER.cid = CART.cid
```

Assume queries (a), (b) and (c) arrive in this sequence. Let X be a cluster of peers, each of which receives a root operator `Project(Select(CUSTOMER, CUSTOMER > 100))` and a fraction of non-overlapped data from the `CUSTOMER` relation. Similarly, let Y be another cluster of peers, each of which receives a root operator `Project(CART)` and a fraction of non-overlapped data from `CART`. Incorporating the operators generated from queries (a) and (c) at multiple peers can effectively process the subsequent projection-join query that involves relation `CUSTOMER` and `CART` without the need for any data from the DBMS.

[2] It has to be acknowledged that POEMS assumes read-dominated workloads (as common in Web environments). Managing updates is beyond the scope of this paper.

An important property of POEMS is that peers are allowed to dismiss a process at will during runtime when they need more resources for their local tasks. This is in vivid contrast to traditional parallel or distributed DBMSs.

The main contributions of our proposed POEMS framework include:

- The proposal of a dynamic framework that is online transformable between CS and P2P architectures, depending on the workload of the system;
- The introduction of operator-based query processing where there is cooperation between different operators at different hosts for query answering;
- The proposal of a movable operator architecture to handle the dynamism of peers where operators can freely move to another peer on request;
- The development of several optimization techniques that decrease data transfer among nodes; and
- The evaluation of POEMS in a real environment with 26 office PCs.

The rest of this paper is organized as follows: Section 2 provides some essential background; Section 3 describes the POEMS framework and its architecture; Section 4 analyzes the proposed techniques; Section 5 presents an extensive experimental evaluation of the system; and, finally, Section 6 concludes the paper.

2 Related Work

POEMS can be categorized as an autonomic computing system, where, according to [11], self-management is envisioned as an intrinsic property that deals with the complexity of modern systems. In autonomic computing, a system maintains and adjusts its operations in order to cope with changing conditions. A similar self-management concept has been proposed for service overlays networks [20].

The goal of overload management is to maintain the system's performance close to optimal under overload conditions. Conceptually, overload management is a special case of load balancing. While overload management deals with a special set of load conditions (i.e. overload), load balancing focuses on distributing equal loads among nodes even in underload conditions. Load balancing in shared-nothing architectures has been well studied [1,2,17] and deployed in several distributed computing projects such as NOW [1], Condor [4] and Beowulf [2]. The methods can be categorized into static [3] and dynamic load-balancing [9,18]. These systems follow a common assumption that loads are distributed to a cluster of machines or processing elements that are fully dedicated to sharing loads. In the case of dynamic environments such as P2P, the notion of 'virtual server' has been used [16] to propose a load-balancing algorithm in distributed hash tables. The P2P systems mentioned above support high workload by means of increasing the number of replicas. Our approach is different in several aspects:

- Operators are distributed among peers and operators cooperate in response to a query, compared to existing techniques that route a query to a single source for processing. POEMS therefore offers the advantage of distributing the query load to multiple nodes minimizing the burden of each node.

- The granularity of data partition is finer and can be adjusted dynamically, allowing data to be partitioned online based on the available resources.
- Peers are involved in query processing when POEMS is overloaded, in contrast to existing techniques in which replicated data are cached permanently while waiting for requests, either in underloaded or overloaded conditions.

Scalability requirements for data-intensive Web applications are the driving factor behind the Database Scalability Service Provider (DSSP) approach [12,14]. DSSP exploits Web application properties, e.g. read-dominated workloads. In addition to the home server, DSSP employs a dynamic number of proxy servers to which the generation of dynamic content is offloaded. Similar to DDMSs, this requires managed servers which is fundamentally different to POEMS.

3 Prototype Design and Implementation

POEMS consists of two main components: The (custom-build) DBMS running on a central server and the AE component running on AEs. Before we consider these components in greater detail, we will define the Processor, which is a widely used object in our system. A root operator is an operator that encapsulates all optimized sub-operators. In order to distinguish it from operators commonly used in the DBMS, we name the root operator *Processor*. The Processor plays an important role in query processing in POEMS. It is a movable object that can be dispatched to AEs as requested or it can replicate itself at other AEs for load sharing. Each Processor consists of operators (all sub-optimized operators are encapsulated in a root operator), data and a Plan. Data are subsets of relations partitioned by the DBMS. A Plan is used to guide operators on how to process a query and where to retrieve other partitioned data.

Figure 1 depicts the architecture of POEMS. From the network point of view, it consists of a large number of AEs offering their spare resources and a central DBMS facilitating user services. The solid lines denote connections among AEs

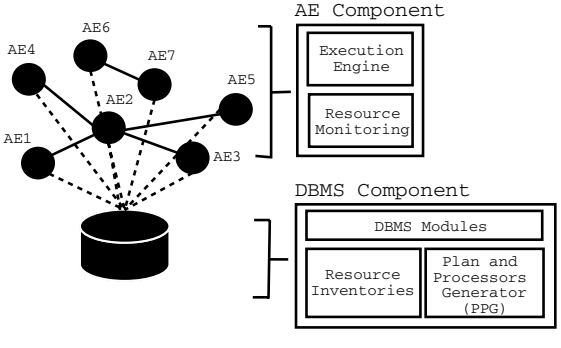


Fig. 1. POEMS Architecture and Context Environment

in the same cluster, e.g. [AE_1, ..., AE_5] and [AE_6, AE_7] are two distinct clusters (say X and Y, respectively) responsible for storing different data. Considering our sample queries from Section 1.1, each AE of X is responsible for storing the partial data of `CUSTOMER` and AEs of Y store the partial data of the `CART` relation. Each AE also connects directly (dashed lines) to the DBMS.

SQL requests are first directed to the central server. The DBMS has all the features of a traditional DBMS (i.e. query parser, query optimizer, various operators etc. as subsystems in the *DBMS Component*). In addition, the DBMS includes *Resource Inventories* and a *Plan and Processor Generator (PPG)*. The former maintain an inventory of the location and characteristics of AEs (e.g. how many memory pages an AE contributes), which is essential for scheduling. A transition from the CS to the P2P mode occurs when the system is overloaded. For each query, the PPG partitions the data, produces Processors to handle the task, and dispatches the Processors to a set of AEs. PPG tries to partition and assign all data to the available AEs. If there are not enough AEs to handle the required amount of data, the remaining data are assigned to the DBMS. In such a case, the DBMS acts as an AE that is responsible for the remaining data.

An AE_i has a very simple architecture consisting of two major modules: The Execution Engine and the Resource Monitor. The *Execution Engine* executes the Processor that it receives without having to know what operators it consists of. Other AEs can connect to AE_i and request data. AE_i may answer a query by executing the operator (or part of it) locally, if it has the required data, or by acquiring data from other AEs. All processed results will be returned directly to the requester that initiated the query without going though the DBMS. Since AEs are not fully dedicated to the task of load management and their load may vary from time to time depending on the tasks that they are currently running, a mechanism is necessary to provide the current resource status of AEs to the DBMS. In our prototype, we implemented a *Resource Monitor (RM)* module monitoring the main memory usage of its AE and calculating the number of available pages. This information is uploaded periodically to the DBMS.

4 POEMS Query Processing

Users pose queries by means of SQL statements to a central DBMS. In our implementation, a query q has the following form: `SELECT` A `FROM` R `WHERE` C, where R is a set of relations, A is the set of target attributes and C is the set of conditions. C in the `WHERE` clause supports the `<expression` *op* `expression>` form, where an `expression` is a column name, a constant or a string expression and *op* can be one of the comparison operators $\{<, <=, =, >=, >\}$.

Let v_{sys} be a function of the system load defined as:

$$v_{sys} = \frac{\texttt{Number of completed transactions per interval-}t}{\texttt{Number of incoming transactions per interval-}t}$$

where interval-t is a constant (e.g. 1min). Overload is interpreted as a condition where the $v_{sys} < v_{adm}$, where $v_{adm} \in (0, 1]$ is a parameter set by the administrator. In such a case, query processing switches to P2P mode to harness more computing power. Obviously, a large value of v_{adm} causes the system to always face an overloaded condition. Thus, shifting the system towards P2P processing. In another extreme case, a very small value of v_{adm} causes the system to retain a CS architecture. This tunable parameter that characterizes POEMS provides better adaptation to user needs based on different environment conditions.

4.1 Query Distribution

Assume that the DBMS notices an increase in load at an interval-t, and attempts to solve the problem by transforming the processing mechanism to P2P mode. Let *firstQuery* be the first query the central DBMS receives upon transformation to P2P mode or a query that requires tables not requested by previous queries while the system was in an overloaded condition. Also, let *followingQuery* be a subsequent incoming request. We shall focus on join and select-join queries that require more processing resources (simple selections are processed similarly).

The DBMS generates a set of Processors to handle each incoming query. First, we discuss the data partition mechanism. Let $R_{firstQuery} = \{R_1, ..., R_n\}$ denote the cross-product of relations R_1 to R_n for *firstQuery*. For each R_i, data have to be partitioned and distributed to a set of AEs. The data size to be assigned to each AE has to be small enough to fit into its main memory in order to avoid any heavy I/O operation causing serious interruption to the user's running tasks.

The relations in $R_{firstQuery}$ are first sorted (using information from the DBMS system catalog) on the size of each relation R_i and the total number of pages that are needed to store all the tuples of $R_{firstQuery}$ is computed. For each AE registered in the Resource Inventories, its buffer (i.e. number of pages it can contribute) is partitioned to $n+2$ segments, where n is the number of relations in $R_{firstQuery}$. Notice, there are two additional segments: One is assigned as storage for intermediate results; and another segment is used as the output buffer. Since all operations are performed in the main memory, pointers instead of real values are stored in the intermediate result buffer.

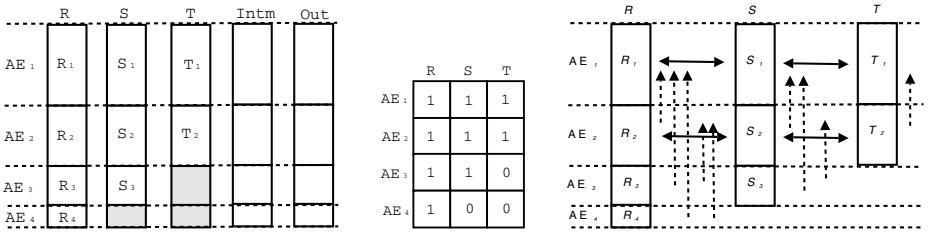


Fig. 2. (a) Data Assignment; (b) A Plan; (c) Data Retrieving

Figure 2(a) is a graphical view of how resources in AEs are partitioned and assigned. Assume R, S and T are three relations of $R_{firstQuery}$ and $Intm$ is the intermediate result buffer and Out is the output buffer. The height of each rectangle is proportional to the amount of data that has been assigned to it. AE_1 is assigned more data than the others since it contributes more resources. No data from T has been assigned to AE_3 and no data from S or T has been assigned to AE_4 (the shaded-rectangles).

Apart from assigning partitioned data to each AE, there is a need to migrate operators. An AE produces final results solely from operators that are assigned to it by the DBMS. Although query optimization is critical in a relational DBMS, we concentrate on a simple local optimization approach. The optimizer can be replaced easily in the future without affecting the general framework. For the current prototype, we employ an iterative improvement algorithm for the randomized optimization of the query plan. The output of the optimizer is an operator consisting of many sub-operators, i.e. nested-join, selection, projection, and scan operators. For each scan operator in an AE, a pointer of a corresponding set of partitioned data generated previously is assigned to it.

The Plan is a simple data structure. It has an $n \times m$ array, where n is the number of relations and m is the number of AEs participating in solving the given query. Each cell (n_i, m_j), $0 < i < n$, $0 < j < m$, in a Plan has a value of 0 or 1. A cell is set if AE_j is assigned data R_i, and it is reset otherwise. Considering Figure 2(a), the information of data distribution among the AEs is transformed to a Plan as in Figure 2(b). The objective of a Plan is to assist query processing by providing information on how and where a query should be processed. In the following, we describe the mechanisms of data retrieval and processing.

Execution of *firstQuery*. On receiving a Processor, the AE starts processing the query. This is divided into two sub-processes: Local processing (LP) and remote processing (RP). LP starts producing partial results immediately by executing the operators that are attached to the Processor if all the required data of $R_{firstQuery}$ are locally available (e.g. in Figure 2, AE_1 and AE_2 can produce partial results locally since all the relations R, S and T are available). The results are returned to the requester immediately. RP involves retrieving data from other AEs to the peer for processing. One of the simplest ways to determine the order of data retrieval from different AEs is the following: AE_j ($0 < j < m$, m is the number of AEs) retrieves the data of relation R_i from AE_{j+1} if and only if the cell of $(i, j + 1)$ in the Plan is set (e.g. in Figure 2(c), AE_1 will retrieve R_2, S_2 and T_2 from AE_2, R_3 and S_3 from AE_3 and R_4 from AE_4). Observe that AE_1 performs more operations than AE_2; similarly, AE_2 performs more operations than AE_3 and so on. The advantage of this resource-based data and task assignment is that AEs with more resources are assigned more operations than others. As a result, the usage of resources at each AE is optimized and the overloading of AEs with low resources is avoided.

Note, the operators of each AE are generated based on an execution plan produced by the DBMS query optimizer with only local information to estimate the result size and cost. Therefore, strictly speaking, the plan is optimized if the

query is run on the DBMS host. Taking R, S and T as an example, based on the associative property of joins, a query optimizer may produce any of these alternative plans: $R \ltimes (S \ltimes T)$, $(R \ltimes S) \ltimes T$ or $(R \ltimes T) \ltimes S$. Since the size of $R \geq S \geq T$, plan $R \ltimes (S \ltimes T)$ is favored by the optimizer. However, since each AE is assigned an equal data size for each relation, there are no indexes in AEs and all the operations are performed in the main memory, processing costs of any of the above-mentioned plans are similar.

However, this might not be valid in RP. RP involves retrieving data from remote AE_k, where $j < k < totalAE$, to AE_j, where the size of R_k in AE_k is smaller than or equal to the size of R_j in AE_j. In this case, assigning a smaller set of data to the outer relation may involve higher processing cost. Although we do not have to consider any I/O operation since all operations are main memory-based, accessing memory entails cost-related characteristics that are similar to disk-based I/O operations. We do not deal with main memory optimization issues in this paper, but we need to reduce L1 and L2 cache misses by avoiding the assignment of a larger set of data to an inner relation and a smaller set of data to an outer relation, as such assignments might cause more L1 and L2 cache misses since the size of the L1 and L2 caches is small compared to the main memory. For example, consider Figure 3(a) and let $R \ltimes (S \ltimes T)$ be the plan produced by the DBMS. Assume that the right-most predicates are inner relations and the left-most predicates are outer relations. Notice, since $R_3 \leq R_2 \leq R_1$, $S_3 \leq S_2 \leq S_1$ and $T_2 \leq T_1$, there are two possible cases where the plan $R \ltimes (S \ltimes T)$ is inadequate for AE_1. First, AE_1 retrieves S_2 or S_3 and replaces the local S_1 with S_2 or S_3 for processing, $R_1 \ltimes (\{S_2, S_3\} \ltimes T_1)$. In this case, since the size of $\{S_2, S_3\} \leq T_1$ and relation S is the outer relation, $\{S_2, S_3\} \ltimes T_1$ would be more costly than $T_1 \ltimes \{S_2, S_3\}$. Similarly, in the second case, AE_1 retrieves R_2 or R_3 and replaces the local R_1 with R_2 or R_3 for processing $\{R_2, R_3\} \ltimes (S_1 \ltimes T_1)$. In this case, the size of the intermediate results produced by $(S_1 \ltimes T_1)$ might be larger than the size of $\{R_2, R_3\}$ since the size of $R_1 = S_1 = T_1$ and $\{R_2, R_3\} \leq R_1$. Piping the larger intermediate results as an inner relation to its parent may affect the cost significantly due to the cache misses of L1 and L2.

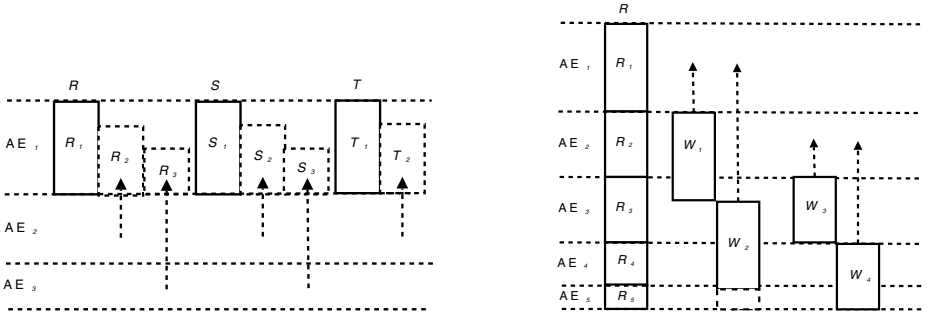


Fig. 3. (a) Remote Processing; (b) Fixed-size Window for Data Retrieval

One of the solutions is to let the optimizer consider the resources at different AEs while generating the execution plan. Different plans can then be tailored for the various AEs. However, this method adds extra load to the DBMS, which is especially undesirable under an overload condition. In POEMS, all AEs are assigned a single plan, which has the advantage of simplicity and scalability. Nonetheless, RP still suffers from the above-mentioned problem. We propose a simple yet effective mechanism to minimize the effect of assigning smaller sets of data to outer relations. Consider Figure 3(b) as an example. We remove S and T from the figure and insert the additional AE_5 for illustrative purposes. For each AE_i, a window W_j with a size that equals the segment size is created. The window W_j is dispatched to $\text{AE}_{(i+1)}$ and is filled with the data of $R_{(i+1)}$. If W_j is not full, it is dispatched to $AE_{(i+2)}$ and the filling process is repeated. The process will stop if W_j is full or AE_n is the last AE in the cluster participating in storing the data of R. An exception happens when W_j is full, AE_n is the last AE and some data remain. In this case, the size of W_j is expanded to capture the remaining data. Note, when expanding W_j, AE_i would require more buffer than the initially assigned segment to store the additional data. The extra buffer that is needed can either be obtained from *Intm* buffer or *Out* buffer, or in the worst case, the additional data may be put on disk. In any case, this may not be an issue since the size of the remaining data is always small.

Execution of *followingQuery*. When the DBMS receives a *followingQuery*, it checks the `FROM` predicates. There are three possible cases: (i) All the required relations have been previously distributed to remote AEs in one cluster; (ii) The required relations are handled by two or more clusters; and (iii) Only parts of the required relations have been distributed to AEs. The DBMS determines the case by referring to the Resource Inventories. In the first case, the DBMS generates operators as usual based on its local cost information. The operators are then attached to a new Processor. Each of the AEs involved in the processing receives a similar Processor. Unlike the Processor of the *firstQuery*, no data or Plan is assigned in the Processor of the *followingQuery* since all such information can already be found at the AEs. This is desirable as the DBMS workload is reduced.

If some of the required data are not found in the AEs (i.e case (iii)), a *followingQuery* will be processed in two steps. First, each part of the missing data is obtained from the DBMS, partitioned and then distributed to a new set of AEs. Second (this also applies to case (ii)), the major challenge is the mechanism of joining relations from two or more clusters, each with a set of AEs. Assume there are two clusters: Cluster (R, S) consisting of $\{\text{AE}_1, \text{AE}_2, \text{AE}_3, \text{AE}_4\}$ and Cluster (T) consisting of $\{\text{AE}_5, \text{AE}_6, \text{AE}_7\}$. Cluster (R, S) stores the data of R and S, while Cluster (T) stores the data of T. Observe that there are several query processing issues: (i) The cluster selection problem of defining where an execution should be performed; and (ii) The size of the to-be-fetched relation in a cluster might be larger or smaller than the available memory resources of the requester AE. Fetching data larger than memory size incurs I/O operations since data have to be stored on disk. At the other end, resources are underutilized when a small set of data is fetched. Given a query $R \bowtie (S \bowtie T)$, POEMS

selects a cluster that has the maximum number of `FROM` predicates stored (e.g. Cluster (R, S) has two predicates out of three of $R \bowtie (S \bowtie T)$). If there is a tie, random selection is used. Notice, given two clusters, each of the AEs in the first cluster has to fetch data from the AEs of the second cluster. Therefore, in order to optimize the memory resources used, each AE in the first cluster will define a window of the size of the segment as described in *firstQuery* processing.

4.2 Reducing the Network Cost

There are two points that involve data transfer: (i) When the DBMS distributes data to a set of AEs; and (ii) During data fetching among AEs. We do not restrict data flow from the DBMS to AEs since any data missing from the AEs will entail references to the DBMS again. On the other hand, data transfer among AEs should be reduced to minimize network cost. Even though the amount of data transfer from the DBMS could not be reduced, it is worthwhile grouping similar queries together and reducing the number of queries that will be posted to AEs for processing. Therefore, our optimization strategy is two-fold: (i) Reduce the number of queries posed to AEs; and (ii) Minimize data transfer among AEs.

We propose a *Windowed and Grouped (WG)* algorithm to reduce the number of queries posed to AEs. WG aims to minimize total execution cost by grouping concurrent queries that require similar resources. Let GW_n be a group window taking n incoming queries and group similar queries together. Given two queries Q_1 and Q_2 (with their respective data sources $R_{(Q_1)}$ and $R_{(Q_2)}$), they can be grouped if and only if $R_{(Q_1)} \subseteq R_{(Q_2)}$ or $R_{(Q_2)} \subseteq R_{(Q_1)}$. Freezing queries and grouping them later may not appear worthwhile at first glance because of the short lifetime of the query and an additional delay of query execution. However, there are two conditions that make WG a desirable algorithm in a situation of overload: (i) The swiftness of requests enables a large number of candidates to be considered for grouping in a very short period; and (ii) A slowdown in the average query response time is unavoidable when system overload sets in.

4.3 Processor Reallocation

Recall that users are allowed to withdraw their contributed resources anytime. When a user withdraws his/her resources, the Processor at the user's AE has to be reallocated. The AE interacts with the DBMS to find an available AE' to handle the Processor. Checking its Resource Inventories, the DBMS informs the AE to migrate the Processor if there is (i) an AE' that has enough resources to take over the Processor, or (ii) a set of AEs whose aggregate resources are enough to take over the Processor. Case (i) is simple and only involves migrating the Processor to AE'. Case (ii) requires splitting the Processor into several sub-Processors. Still, this is straightforward as we only need to split the data in the Processor to several portions according to the new AE's resources. If there is no AE' available, the Processor will be relocated to the DBMS. In either case, the DBMS generates a new Plan and updates all other AEs who are members of the same cluster as the requester on the new location of the Processor.

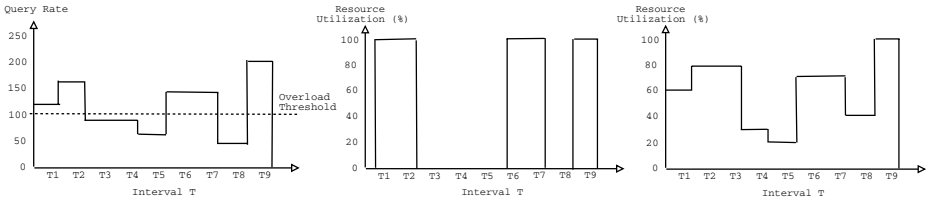


Fig. 4. Resource Utilization: (a) Server Workload; (b) Naive; (c) Microeconomics

4.4 Optimizing the Utilization of Resources

Figure 4(a) presents a server workload scenario. When the query rate is more than 100 queries per period, the server is overloaded. During such periods, it requests resources from the peers. Figure 4(b) presents the resource utilization of a naive algorithm, which utilizes 100% of the available peer resources and releases them when the peak ends. This is acceptable from the peers' perspective. However, with a new peak, the server must retransmit all required data.

To achieve a balance among the total amount of data transmitted by the server, the throughput of the system and the peers' desire to avoid contributing resources, we adopted an algorithm based on microeconomics. Each peer has a set of blocks representing its resource. Each block can be reserved for storing data by paying some amount of virtual currency. The block is reserved and valid for a period of an interval-t. On expiration, the peer has the right to regain it back if the requestor does not pay for it. In order to reserve the blocks, the requestor has to earn profit. The server earns profit for every query it manages to serve while exceeding its capacity. On each interval-t, the `OnLoad` function is called to compute the block reservation strategy. It first pays for all the reserved blocks if the account has enough currency, otherwise the peer will regain the blocks. After clearing the payment, it checks if the number of reserved blocks are enough to handle the incoming requests. If more blocks are needed and the account has enough currency, it buys extra blocks. Otherwise, it needs to loan extra blocks for storing the data. The loaned blocks have to be paid back once it has earned the profit. However, the profit may not always be enough to pay for all blocks and unpaid blocks are returned to peers (releasing cached data).

Figure 4(c) shows the behavior of the algorithm for the server workload of Figure 4(a). It is evident that our algorithm follows closely the workload and avoids using all available resources on the peers if not necessary. Additionally, it keeps some resources in the peers even when the server does not need them at the time, since these may improve query performance during the next peak.

5 Experimental Evaluation

We tested our prototype on a network with 26 Pentium IV PCs (1.6MHz, 256MB RAM) and a Sun Solaris server (two Ultra-SRARC processors, 480MHz,

4GB RAM) serving as the central DBMS. All machines were physically connected to a LAN. The server-to-AE and the AE-to-AE transfer rates were 10Mbps and 100Mbps, respectively. We used the TPC-H schema to generate the data set for our experiments. It consists of eight separate tables with a total of about 900,000 tuples. We defined a query set with a mixture of selection and join queries. We used two metrics to evaluate the system performance: (i) The number of completed transactions per interval (i.e. throughput) denoted as $CP(t)$; and (ii) The effectiveness of a system defined as: $100 * CP(t)/Q(t)$, where $Q(t)$ is the number of incoming requests per interval-t. The default value of t was 60min. Obviously, when the server is not overloaded, its effectiveness is 100%.

5.1 Client-Server vs. POEMS

In the first set of experiments, we compare a centralised DBMS against POEMS. For fairness, we do not employ any optimization strategies for POEMS. The server is considered overloaded when the rate of incoming queries $Q(t) \geq 120$. The results are presented in Figure 5.

When $Q(t)$ is low, both systems achieve best results. However, when $Q(t)$ increases to more than 120, the server can no longer handle the additional load. Therefore $CP(t)$ for CS becomes constant, while the effectiveness drops. POEMS, on the other hand, switches into the P2P mode. For $Q(t) = 120$, the server is overloaded and additionally has to pay the cost of transforming to P2P mode. This involves data partitioning and distribution, which are expensive processes since they involve scanning large portions of the data and preparing the Processor objects to be distributed to the remote AEs. For this reason, POEMS' initial performance is slightly worse than CS. After POEMS has paid the initial cost, however, it can utilize the peers' resources to increase its throughput. Since the data are already in the peers, only operators are sent to AEs for subsequent queries. Moreover the queries can be evaluated in parallel from data which reside in the peers' main memory. These facts explain the performance boost of POEMS over CS for high query rates.

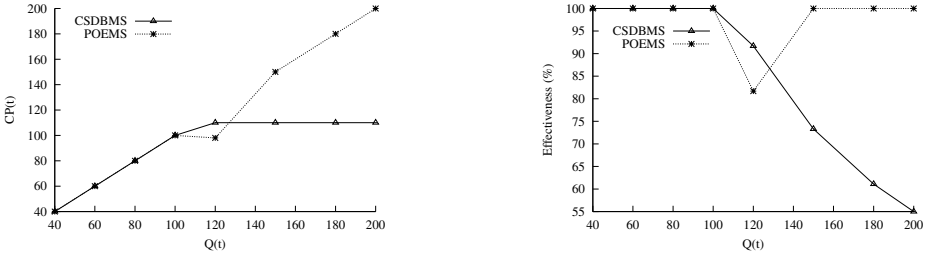


Fig. 5. CS vs. POEMS: (a) Throughput; (b) Effectiveness

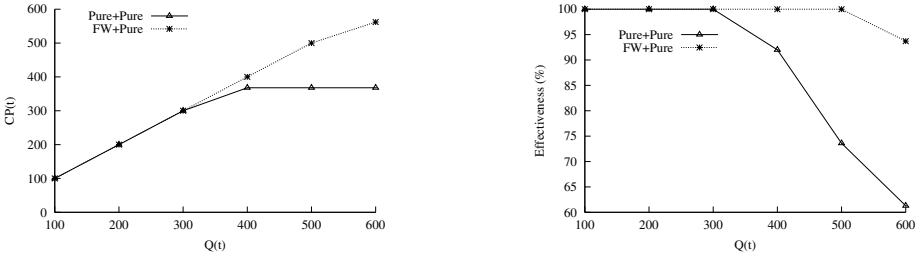


Fig. 6. POEMS Optimizations: (a) Number of Completed Queries; (b) Effectiveness

5.2 Optimized POEMS

We evaluate the query grouping optimization as presented in Section 4.2. Settings are the same as above and the results are presented in Figure 6, where 'Pure+Pure' represents a POEMS system with all the optimizations turned off while 'FW+Pure' denotes POEMS with the query grouping optimization.

Query grouping identifies similar queries and shares the execution cost among them. Essentially, this optimization decreases the number of distinct queries running simultaneously; therefore it allows the system to complete more queries in a given period. This is evident from the figures, where query grouping increases the system's performance by around 50%.

We tested other optimizations, e.g. Semi-joins and Bloom-filter joins. Both methods further improved the results, which we omit due to space constraints.

5.3 Network Load Optimization

We implemented a simulator using the parameters from the prototype in order to evaluate the effectiveness of network load optimization on a larger network.

In addition to the Microeconomics algorithm, we also tested distributed hash joins. Results show that hash joins benefit much from an increased peer number (due to joins being performed in parallel with negligible communication cost).

Effectiveness of Microeconomics Algorithm. We compare the network load of a naive resource utilization algorithm with that of the microeconomics-based resource utilization from Section 4.4. Figure 7(a) presents a scenario of server workload. When the query rate is more than 100 queries per period, the server is overloaded and utilises its peers' resources. With a naive resource utilization algorithm, the server transmits all the required data to the peer, which discards the data once the peak ends. When a new peak starts, the server must retransmit all the data. With the microeconomics-based method, some amount of data is retained at the peer even when the peak is over, so that it can be reused during the next peak. This avoids the need to retransmit all the data at each peak reducing the network load. In the simulator, the network transfer rate is set to 10Mbps and a simple selection query is being used. Figure 7(b) shows the results obtained. At intervals T_6 and T_7 the data cached at the

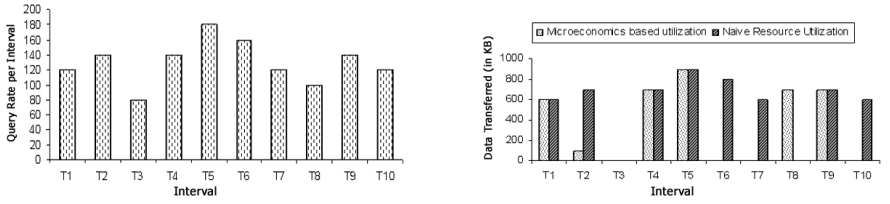


Fig. 7. Microeconomics Algorithm: (a) Server Workload; (b) Network Utilization

peer is reused and, hence, no new data needs to be transmitted in the case of microeconomics-based resource utilization, while all the data needs to be retransmitted with the naive resource utilization algorithm. At T_8 there has been a data transfer due to the microeconomics-based utilization, because the resources at the peer are not enough to handle the load. Over the periods T_1 to T_{10}, the naive resource utilization algorithm causes a transfer of a total of $5,600$KB of data, while the microeconomics one causes a transfer of $3,700$KB of data. Thus, there is about 33% of savings by the latter. This confirms that the microeconomics-based algorithm optimizes network utilization more effectively.

6 Conclusion

In this paper, we investigated the practical problem of dealing with overload in enterprise DBMSs. Motivated by the fact that current solutions are either too expensive or complicated to be deployed, we developed POEMS. Our system utilizes the available resources in numerous office PCs by delegating data manipulation in order to relieve the DBMS during peak periods. The whole procedure is transparent to the user and does not require the installation of any DBMS software on the PCs. Furthermore, PCs contribute their resources only when it is absolutely necessary minimizing the disturbance to the office user.

A POEMS prototype illustrates the feasibility of the proposed framework, and the experimental results demonstrate its potential. We also showed that under known query patterns, the system can be very scalable. We are confident that by incorporating sophisticated optimization methods, POEMS has the potential to achieve even better performance in a wide range of practical applications.

References

1. Anderson, T.E., Culler, D.E., Paterson, D.A.: A case for NOW (network of workstations). IEEE Micro 15(1), 54–64 (1994)
2. Becker, D.J., Sterling, T., Savarese, D., Dorband, E., Ranawake, U.A., Packer, C.V.: BEOWULF: A Parallel Workstation for Scientific Computation. In: ICPP, pp. 11–14 (1995)
3. Copeland, G., Alexander, W., Boughter, E., Keller, T.: Data Placement in Bubba. In: ACM SIGMOD, pp. 99–108 (1995)

4. Epema, D.H.J., Livny, M., van Dantzig, R., Evers, X., Pruyne, J.: A Worldwide Flock of Condors: Load Sharing Among Workstation Clusters. Journal on Future Generation Computer Systems 12(1), 53–65 (1996)
5. Halevy, A.Y., Ives, Z.G., Suciu, D., Tatarinov, I.: Schema Mediation in Peer Data Management Systems. In: ICDE, pp. 505–516 (2003)
6. Halevy, A.Y., Ives, Z.G., Madhavan, J., Mork, P., Suciu, D., Tatarinov, I.: The Piazza Peer Data Management System. IEEE Transactions on Knowledge and Data Engineering 16(7), 787–798 (2004)
7. Halevy, A.Y., Rajaraman, A., Ordille, J.: Data Integration: The Teenage Years. In: VLDB, pp. 9–16 (2006)
8. Haas, L.M., Miller, R.J., Niswonger, B., Roth, M.T., Schwarz, P.M., Wimmers, E.L.: Transforming Heterogeneous Data with Database Middleware: Beyond Integration. IEEE Data Engineering Bulletin 22(1), 31–36 (1999)
9. Helal, A., Yuan, D., El-Rewini, H.: Dynamic Data Reallocation for Skew Management in Shared-nothing Parallel Databases. Distributed and Parallel Databases 5(3), 271–288 (1997)
10. Kementsietsidis, A., Arenas, M., Miller, R.J.: Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues. In: ACM SIGMOD, pp. 325–336 (2003)
11. Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. IEEE Computer 36(1), 41–50 (2003)
12. Manjhi, A., Ailamaki, A., Maggs, B.M., Mowry, T.C., Olston, C., Tomasic, A.: Simultaneous Scalability and Security for Data-intensive Web Applications. In: ACM SIGMOD, pp. 241–252 (2006)
13. Ng, W.S., Ooi, B.C., Tan, K.-L., Zhou, A.: PeerDB: A P2P-based System for Distributed Data Sharing. In: ICDE, pp. 633–644 (2003)
14. Olston, C., Manjhi, A., Garrod, C., Ailamaki, A., Maggs, B.M., Mowry, T.C.: A Scalability Service for Dynamic Web Applications. In: CIDR, pp. 56–69 (2005)
15. Parent, C., Spaccapietra, S.: Database Integration: An Overview of Issues and Approaches. Communications of the ACM 41(5), 166–178 (1998)
16. Rao, A., Lakshminarayanan, K., Surana, S., Karp, R., Stoica, I.: Load Balancing in Structured P2P Systems. In: International Workshop on Peer-to-Peer Systems (2003)
17. Stonebraker, M.: A Case for Shared Nothing. Database Engineering 9(1), 4–9 (1986)
18. Scheuermann, P., Weikum, G., Zabback, P.: Data Partitioning and Load Balancing in Parallel Disk Systems. VLDB Journal 7(1), 48–66 (1998)
19. Tomasic, A., Raschid, L., Valduriez, P.: Scaling Heterogeneous Databases and the Design of Disco. In: ICDCS, pp. 449–457 (1996)
20. Liang, J., Gu, X.H., Nahrstedt, K.: Self-configuring Information Management for Large-scale Service Overlays. In: IEEE INFOCOM, pp. 472–480 (2007)

Improving Web Service Discovery by Using Semantic Models

Aishwarya Bose, Richi Nayak, and Peter Bruza

Faculty of Information Technology, Queensland University of Technology, Australia
{a.bose,r.nayak,p.bruza}@qut.edu.au

Abstract. With the advent of service oriented architecture, Web services have gained tremendous popularity. This warrants the need to establish an effective and reliable process of Web service discovery. This paper presents a novel approach to enhance the accuracy of Web service discovery by finding semantically similar Web services for a user query using the support-based latent semantic kernel. The empirical evaluation confirms that the accuracy of Web service discovery with the proposed method shows a significant improvement over traditional discovery methods.

1 Introduction

Web Service (WS) is a buzz word of today. With corporate world moving towards Service Oriented Architecture (SOA), Web services are currently widely in use. In most service oriented architectures, business-to-business as well as business-to-customer systems, Web services play a vital role for conducting daily transactions and information exchange. A Web service is a public interface of remotely invoking an application to perform a business function or a set of functions.

As the number of Web services increases because of inexpensive technical resources, the problem of locating Web services of interest from a large pool of Web services becomes prominent [8, 24]. Most of the service discovery mechanism use traditional attribute-based matchmaking algorithms that fall short of capturing the semantics for service discovery [11] and/or partially satisfy the need of user search. Due to a weak searching mechanism, the desired service(s) is often not returned and the user may have to look for different search terms to achieve the result. Hence, finding the appropriate Web services according to the need of the users is still a challenge.

Universal Description, Discovery and Integration (UDDI) registry, which is a collection of all registered Web services available on the Internet, enables discovery of Web service providers. Each Web service is attached to a Web Service Description Language (WSDL) [7] document. The WSDL document contains the information about the description of a Web service and how to access the service using XML tags. The potential to achieve dynamic, scalable and cost-effective infrastructure for electronic transactions in business and public administration has driven recent research efforts towards semantic Web services that is enriching Web services with semantics. Semantic information aims to enhance the integration and Web service discovery by utilizing the machine readable constructs of the representation. A number of ontologies like OWL-S

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 366–380, 2008.

[15] and WSDL-S [4] have been proposed to address the semantic heterogeneity among Web resources and services. However, a majority of the Web services available over Internet are not annotated using any of these ontologies. This is mainly because WSDL is the standard language to express a service and any semantically enhanced language is not yet a standard that must be followed. Secondly, a number of ontologies and frameworks have been proposed each having its own advantages and disadvantages. This makes hard to select one as a standard. Moreover, there exist a large number of Web services on the Internet that have been already created long before any of these annotations were proposed.

This warrants a need to discover the Web services accurately using smart searching techniques. In this paper, we propose a methodology for discovering Web services adapting semantic models and data mining techniques. Previous researchers have proposed a number of ideas to enhance the accuracy of Web service discovery by applying data mining approaches[16, 17], singular vector decomposition [22], graph based methods, various ontology based discovery frameworks and others [5, 25]. However, most of these approaches have not been thoroughly tested and evaluated.

The contribution of this paper is two-fold. Firstly, a novel Web service discovery method based on the semantic similarity derived from the trained support-based kernel is introduced. We propose the creation of latent semantic kernel with the support-based algorithm using the concept of binning & merging, and then utilise the kernel to find semantically similar Web services for a user query. Secondly, a thorough practical experimentation and evaluation have been performed. The empirical analysis confirms that the proposed method is able to find semantic relations and thereby to improve the process of Web service discovery in comparison to traditional methods.

2 Related Work

A considerable body of research has emerged proposing different methods of improving accuracy of Web service discovery. Web service search engine-Woogle [8] utilizes clustering and association mining to find similarity between Web services based on common user queries. Researchers have proposed Web service discovery frameworks using various ontologies [6, 9, 10, 19, 21] to enhance the semantics in WSDL. These frameworks may enhance the semantic part but the major concern remains how to deal with the existing ones that already have been published.

Recently a Web service discovery method combining semantic and statistical association with hyperclique pattern discovery [18] has been proposed. Algorithms using singular vector decomposition (SVD) [22] and probabilistic latent semantic analysis [14] have been proposed to find the similarity between the Web services to enhance the accuracy of service discovery. However none of these methods provides empirical and theoretical analysis showing that these methods improve the process of Web service discovery. Our approach is an extension of SVD [22] to support-based latent semantic kernel to further increase the accuracy of Web service discovery.

A common problem with the SVD based approaches is that the computation of the high dimensional matrix representing the training documents is expensive. There have been some attempts to reduce the dimensionality of matrix prior to applying SVD. One such solution is using random projection [20]. In random projection, the initial

corpus is projected to l dimensions, for some $l > k$, where k is the dimension of the semantic kernel, to obtain a smaller representation which is close to the original corpus and then perform SVD on the reduced dimension matrix [13, 20]. The proposed approach of dimensionality reduction in this paper has outperformed the random projection method as shown in section 4.4. In the proposed approach we have created the semantic kernel on a large Wikipedia corpus. Our particular contributions are in terms of dimensionality reduction by introducing the concept of merging documents as well as using the constructed kernel on a general-purpose corpus to find semantically similar Web services for a user query.

3 The Proposed Web Service Discovery Method

In this paper, we propose a Web service discovery method based on semantic kernels to find the most relevant Web services for a user query. The use of semantic kernels in the Web service discovery method helps to locate semantically similar Web services that were otherwise not found. For example, a user who is looking for Web services related to 'weather' may also be interested in Web service related to 'climate' or 'rainfall', since they are semantically related.

A term-document matrix representing all training documents is used to construct the semantic kernel for finding different topics (related terms). The constructed semantic kernel, which represents each document as a set of topics, discovers these hidden topics and their relationships to the term and document set. Due to the huge number of the terms present in the documents and the large number of training documents present in the dataset, the construction of kernel poses a problem for large data sets due to the large dimensionality of matrix. We propose the dimensionality reduction of the term-document matrix by binning & merging the training documents to create the semantic kernel which is one of the innovations in the paper.

Fig. 1 depicts the overview of the proposed Web service discovery method. The similarity between the user query and Web services is calculated using the support-based latent semantic kernel. To create the kernel, the first step is text pre-processing which includes extracting the content of the training documents followed by the standard processes such as stop-word removal and stemming. During the pre-processing stage, the words that are hybrid in nature are also processed. This is done to convert a hybrid word into multiple standard dictionary words wherever possible. A hybrid word can be a compound word (e.g. sandpaper) or a composite word (e.g. BookPriceCheck) or a joint word with a connected symbol such hyphen (e.g. Book_Price). A WSDL document may follow any of the naming conventions to name a variable such as Camel case, Pascal case, joining words using underscore or other available standard naming conventions. For example, 'stockBroker' or 'StockBroker' or 'stock_broker' all being the same becomes stockbroker.

An additional process is added for calculating the importance of documents in a corpus (more detail is in section 3.1.2). On the basis of document importance, the training documents are assigned into bins such that each bin contains equally important documents. The documents of a bin are then merged to form a single document representing the content of the bin. The content of the documents from different bins are processed to form the term-document matrix where each row represents a unique

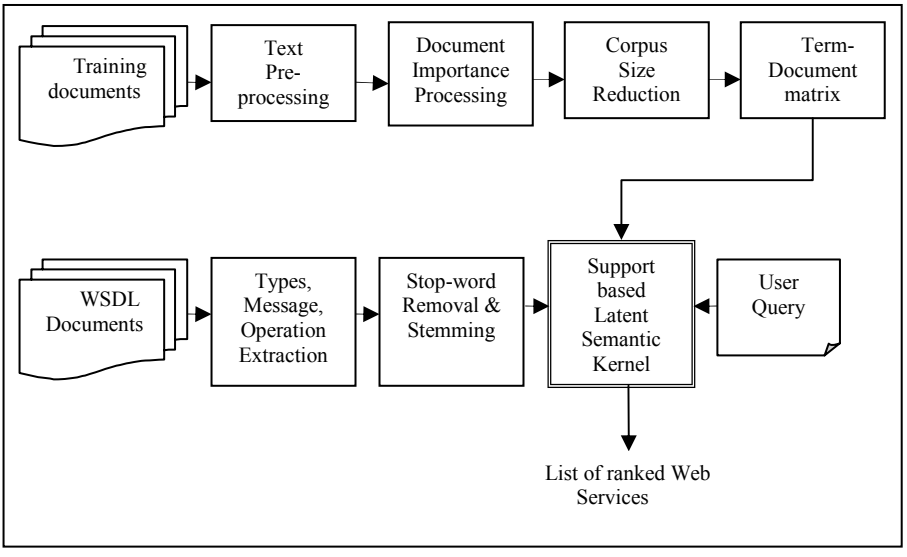


Fig. 1. Overview of Web service discovery approach

term and each column represents a merged document. The cell represents the relative importance of the term in the document. A latent semantic kernel is created performing SVD on this term-document matrix.

In order to find the similarity between the user query and the Web services, the first step is extracting the content from the WSDL documents followed by stop-word removal & stemming. The constructed support-based semantic kernel in the training phase is then used to find the similarity between WSDL documents and a query when the query is provided. The topics of WSDL documents which are most related to the query topics are considered to be the most relevant. Based on the similarity computed using the support-based semantic kernel, the WSDLs are ranked and a list of appropriate Web services is returned to the user.

The selection of training corpus to build the latent semantic kernel is crucial. If the training corpus does not cover the WSDL topics, the results will not be satisfactory. We chose Wikipedia representing the world of knowledge for constructing the kernel. The selection of this training data corpus ensures that the WSDL data topics will be covered to a large extent. So it is not too specific to any particular WSDL or topic and the same kernel can be used for all WSDL documents. In this way we take benefit of having a general-purpose training corpus independent of the nature of WSDL documents.

3.1 Building of Support-Based Latent Semantic Kernel

We now explain the process of creating the support-based latent semantic kernel.

3.1.1 Background Information

Let P be a matrix that transforms documents from the higher-order input space to a lower-order feature (or topic) space. To compute P, singular vector decomposition

(SVD) is performed on the term-document matrix A where m and n represent the number of documents and terms present in the training data set respectively.

$$A_{n\times m} = U_{n\times r} \sum_{r\times r} V^T_{r\times m} \tag{1}$$

where $\sum$ is a rxr diagonal matrix composed of non-zero eigen values of AA^T. The columns of U and V are orthogonal eigen matrices associated with r non-zero eigen values of AA^T.

The original document vectors are projected into the subspace created by the first k singular vectors of the reduced space. Hence, the dimension of the original space is reduced to k and this dimension can be controlled by varying k. By selecting a k singular value, $k < r$, a k-dimensional space can be defined. Using the dimensions nxk and mxk, matrices U_k and V_k can be redefined along with kxk diagonal matrix $\sum$. Thus

$$A_k = U_k \sum_k V_k^T \tag{2}$$

is the nearest matrix of rank k to the original matrix A, and P becomes U_k [12], the latent semantic kernel .

3.1.2 Methods for Kernel Creation

As stated earlier, due to the large number of terms and documents present in the training dataset, the dimensionality of the term-document matrix is extremely large which makes the matrix computations very time and space consuming. We propose a novel way of reducing the dimensionality of the term-document matrix by binning & merging. The proposed approach reduces the dimensionality without any information loss since all pre-processed terms (after stemming and stop-word removal) are used in building the kernel. We have used two different methods to create the semantic kernel, one by random document selection and the other by support-based document selection. The support-based document selection aims to remove any bias which might have been introduced because of the random selection of documents.

Let the training dataset be $D = \{D_1, D_2, \ldots, D_m\}$. Let a document D_i contains a maximum of n unique terms ($t_{i1}, t_{i2},\ldots, t_{in}$) after the stemming and stop-word removal, having the corresponding frequencies ($f_{i1}, f_{i2},\ldots, f_{in}$). Let T_j be a unique term in the dataset, the total frequency of T_j is denoted by $F_j = \sum_{i=1}^{m} f_{ij}$.

We propose the concept of computing "document importance" for merging several documents in order to reduce the dimensionality of term-document matrix. The documents are weighted according to their document importance and equally distributed across several bins. Documents of each bin are then merged to form a single document. This process significantly reduces the dimensionality of the term-document matrix depending upon the required number of bins. In order to compute the document importance, we define weightage and support of a term. Support of a term denotes the relative importance of the term in the whole corpus and weightage of a term represents the importance of the term within the document. Inclusion of support and weightage both in calculating document importance removes the bias of a frequent term present only in a document.

The weightage W_j, of a term T_j in a document shows the relative importance of the term in the document. It is defined as the ratio of the frequency of the term in the document to the frequency of all terms present in that document, i.e.

$$W_j = \frac{f_j}{\sum_{j=1}^{n} f_j} \tag{3}$$

The size of the dataset, F, is the sum of the frequencies of all terms present in the dataset i.e.: $\sum_{j=1}^{n}\sum_{i=1}^{m} f_{ij}$.

Let $S = \{S_1, S_2, \ldots, S_n\}$ be the collection containing the support of n unique terms present in the dataset. The support S_j of a term T_j in a corpus shows the relative importance of the term in the corpus. S_j of term T_j is the ratio of the total frequency of T_j to the size of the dataset. It is calculated as

$$S_j = \frac{F_j}{F} = \frac{\sum_{i=1}^{m} f_{ij}}{\sum_{j=1}^{n}\sum_{i=1}^{m} f_{ij}} \tag{4}$$

Using the weightage and support of all terms, the document importance DI_i is calculated as:

$$DI_i = \sum_{j=1}^{n} W_j * S_j \tag{5}$$

The task now is to selectively choose the documents and place them into bins and then merge all the documents of a bin to form a single document. The rational behind this is to reduce the dimensionality of the term-document matrix without having loss of terms. Thus the number of documents is reduced but the number of terms does not change.

Let B be the collection of q bins, $B = \{B_1, B_2, \ldots, B_q\}$ where each bin contains equal number of documents, dividing the dataset D into q bins. Let BD be a collection of q number of documents, $BD = \{BD_1, BD_2, \ldots, BD_q\}$, where BD_i is the result of merging all the documents present in the bin B_i. After merging, the frequency of the terms in the merged document is computed to remove very low and very high frequency terms as they could be possible outliers.

Let K be the revised term-document matrix, containing x rows and y columns which will be the input for constructing the latent semantic kernel. Each row represents a unique term, each column represents a merged document and each cell contains the support value (equation 4) of a term in a merged document. Let U_k represent the semantic kernel which is the output of SVD on matrix K with a suitable value of k selected during experiment.

Fig 2 describes the algorithm for creating the support-based semantic kernel using the concept of binning & merging.

Random Selection of Documents	Support Based Selection of Documents
Input: D: Training Dataset, m: Number of Documents, q: Number of Bins, K: term-document matrix Output: U_k: Latent Semantic Kernel	
1. Remove the stop words and perform stemming for each document in D 2. /* Binning Documents */ 3. Divide the dataset into q bins by selecting equal number of documents at random 4. /* Merging Documents */ 5. for each $B_q \in B$ a. BD_q = empty b. for each $D_i \in B_q$ i. $BD_q = BD_q \cup D_i$ c. end for 6. end for 7. /* Initialize the term-document matrix K */ 8. for each $x = 0$ to T_j a. for each $y = 0$ to D_i i. $K[x][y] = 0$ b. end for 9. end for 10. for each $BD_q \in BD$ *a.* for each $T_j \in BD_q$ 1. $K[T_j][BD_q] = K[T_j][BD_q] + S_j$ b. end for 11. end for 12. U_k = SVD (K)	1. Remove the stop words and perform stemming for each document in D 2. Calculate weightage W_j (equation 3) and support S_j (equation 4) for each term T_j 3. Calculate document importance DI (equation 5) for each document in D 4. Sort DI in ascending order 5. /* Binning Documents */ 6. for $i = 0$ to q a. $B_i = 0$ b. for $j = 0$ to m/q i. $B_i = B_i \cup D_{j*q+i}$ c. end for 7. end for 8. /* Merging Documents */ 9. for each $B_q \in B$ a. BD_q = empty b. for each $D_i \in B_q$ i. $BD_q = BD_q \cup D_i$ c. end for 10. end for 11. Initialize the term-document matrix K 12. for each $BD_q \in BD$ *a.* for each $T_j \in BD_q$ i. $K[T_j][BD_q] = K[T_j][BD_q] + S_j$ b. end for 13. end for 14. U_k = SVD (K)

Fig. 2. Algorithms for Random & Support based document selection for semantic kernel creation

Once the kernel is created, it is used in calculating the similarity between Web services (WSDL documents) and the user query. The WSDL documents are processed to extract the meaningful information for further calculation.

3.2 Finding Similarity between a Query and WSDLs with Using the Kernel

In the proposed approach, we utilised the semantic kernels to find the similarity between the user query (Q') and a Web service (W') using the following model:

$$sim(Q',W') = \cos(Q',W') = \frac{Q'^{T} PP^{T} W'}{\left\| Q'^{T} P \right\| \left\| P^{T} W' \right\|} \tag{6}$$

where P is the latent semantic kernel constructed on the training documents. The similarity between the query and all the WSDLs present in the repository are computed for a query. The WSDLs are then ranked in the order of similarity value. A list of top-n Web services is returned to the user.

A WSDL document is defined using Types, Message, Operation, Port Type, Binding, Port and Service [7] and thus requires processing of various components for similarity calculation. The content of a WSDL document is extracted from the tags such as Types which contains the parameters and the data type of the operations, Port Type which contains the operation name, input & output messages and the documentation which is the description of the Web service. The binding information is ignored since it deals mainly with integrating Web services and may not provide semantic information. The extracted content is then subjected to stop word removal & stemming. Using equation 6, we compute the similarity between the query and the operation name (N'), input & output parameters (P') and description (D') of the Web service. These component similarities are aggregated to compute the total similarity (S_{sum}) between the query and the Web service using the following equation:

$$S_{sum} = w_1 * sim(Q',N') + w_2 * sim(Q',P') + w_3 * sim(Q',D') \tag{7}$$

where, w_1, w_2 and w_3 are the assigned weights to give different weightage to each of the operation name (N'), parameters (P') and description (D') components of the Web service respectively while calculating similarity between a query and the Web service. After careful consideration of various scenarios, equal weightage is assigned to each of the components such that:

$$w_1 = w_2 = w_3 \text{ and } w_1 + w_2 + w_3 = 1 \tag{8}$$

Equal weightage scheme is able to pick up the difference in two Web services even if they are different only in some components. Consider the following example. A price check service from an online bookstore and a stock market service may have similar input and output parameters (e.g. input: name (string) & output: price (float)). However, the method name will be different such as BookPriceCheck and StockPriceCheck. Computing the total similarity using equation 7, $sim(Q',P')$ (similarity between the query and parameters) will be equal for both the services. However, $sim(Q',N')$ (similarity between the query and operation name) and $sim(Q',D')$ (similarity between the query and service description) will be different.

4 Empirical Evaluation

To validate the proposed approach, we have performed extensive experiments. We have obtained WSDL documents from the real-life Web services to perform experiments. We have thoroughly evaluated our approach considering all live Web services.

4.1 Dataset

A general-purpose Wikipedia dataset [1] is used in creating the semantic kernel. This dataset has been used since it contains varied topics from different fields like art, law, history, aviation, archaeology, chemistry, sports, music, literature etc. Thus, the kernel represents the knowledge of many possible domains that a Web service may belong to. It contains 48306 documents that are too many for any SVD based technique to handle on a PC of Intel Core Duo 1.86 GHz processor and 2 GB of memory. The original term-document matrix represented by this data set is reduced to a size utilising the method proposed in section 3.1.2 so that SVD can be performed successfully.

The kernel constructed from this dataset is used to find the similarity between a query and Web services. The Web service dataset contains 873 WSDL documents from XMethods [2] and QWS Dataset [5] representing a variety of Web services such as stock market, music, literature, sms, translation, sports, geographic location etc.

4.2 Experiment Design

We have performed several experiments to evaluate the performance of the proposed support-based semantic kernel based Web service discovery method. The proposed method is compared with the standard keyword (tf*idf) [23] based information retrieval method. The proposed method is also compared with the query expansion based approach using Wordnet [3]. Wordnet expands the query by adding the semantically similar terms in the query so that the precision of the search can be improved.

For representing the matrix to construct the semantic kernels, two measures have been employed in experiments, namely the standard tf*idf values [23] and the proposed support measure as shown in equation 4 of section 3.1.2.

Experiments have also been performed to evaluate the effectiveness of dimensionality reduction by the proposed binning & merging according to document importance measure. Two types of kernels are constructed utilising the term-document matrix based on the merged documents using the support-based distribution and using the random distribution. The performance of semantic kernels constructed with the proposed merged documents is compared with the semantic kernels constructed with (1) the support-based selection (selecting documents based on document importance as defined in equation (5)), (2) the arbitrary document selection (selecting documents arbitrarily from the dataset), (3) the document-size based selection (selecting documents having the highest content measured using document size) and (4) the random projection method [20].

Each of the experiments have been performed using 50 user-defined queries, each query having at least two terms and the average term size is three.

4.3 Evaluation Measure

To evaluate the accuracy of the proposed method, *precision, recall* and *F-score* measures are used. Precision is defined as the ability to provide the relevant Web services from a set of retrieved Web services. Recall is the ability to provide maximum number of relevant Web services from a set of relevant Web services. F-score is the harmonic mean of precision and recall. Mathematically, they are defined as follows:

$$\text{Precision} = \frac{Number\ of\ relevant\ Web\ servicves\ retrieved}{Total\ number\ of\ retrieved\ Web\ services\ from\ the\ dataset} \tag{9}$$

$$\text{Recall} = \frac{Number\ of\ relevant\ Web\ services\ retrieved}{Total\ number\ of\ relevant\ Web\ services\ in\ the\ dataset} \tag{10}$$

$$\text{F-Score} = \frac{2 * Precision * Recall}{Precision + Recall} \tag{11}$$

The reported results are averaged over 50 queries. While evaluating the approach, top-*n* precision has been considered since the user is most likely to be interested in a list of first *n* Web services. The value of *n* has been selected as 10 & 20 for practical experimentation.

4.4 Results

We constructed several semantic kernels based on the varying size of input matrix (term-document) according to the methods of document size reduction as discussed in section 4.2. Table 1 summarises the number of documents and the number of words after stop-word removal, stemming and removing very high (greater than 10000) and very low (less than 5) frequency terms as they could be outliers in creating the semantic kernels. These semantic kernels are then utilised in finding the similarity between the query and the WSDL documents for Web service discovery.

Table 1. Table showing number of documents and words used in different methods

Method	Types	No. of Documents	Reduced No. of Documents	No. of Words
Latent Semantic Kernel	Random Selection (Merging)	48306	967	34065
	Support Based Selection (Merging)	48306	967	34082
	Support Based Selection	967	-	5576
	Arbitrary Document Selection	967	-	5768
	Document Size Based Selection	967	-	21787
	Random Projection	2500	967	10257

Table 2 lists the Web service discovery results in terms of precision, recall and F-score. These are average results over 50 queries. This table shows the results of the proposed method utilising two ways of size reduction in making semantic kernels along with the standard information retrieval methods based on tf*idf and tf*idf with semantic enhancement using Wordnet [3]. Results show that the proposed Web service discovery method utilising the semantic kernel performs significantly better in comparison to tf*idf as well as tf*idf with Wordnet enhancement.

Table 2. Comparing Latent Semantic Kernel with Information Retrieval based tf*idf methods based on Precision, Recall and F-Score

Methods	Types (Merging)	Sub-types (U_k=300)	Precision		Recall	F Score	
			Top 20	Top 10		Top 20	Top 10
Latent Semantic Kernel	Support Based Selection	**tf*idf**	**34.9**	**48.4**	**63.24**	**44.98**	**54.84**
		Support	**33.9**	**46.8**	**61.53**	**43.72**	**53.16**
	Random Selection	tf*idf	34.1	48.4	61.85	43.96	54.30
		Support	33.8	44.2	61.08	43.52	51.29
Traditional methods	tf*idf (with Wordnet enhancement)		31.4	41.4	56.90	40.47	47.93
	tf*idf		30.4	40.8	54.68	39.08	46.73

Support-based semantic kernel built with the term-document matrix representing tf*tdf measure seems to perform better than the kernel built with matrix using the support measure. This is because some words are present sparsely in the corpus and the support value obtained from equation 4 is too low.

Fig 3 provides a sample of the queries used during experimentation. From the figure it is evident that support-based method is much more efficient compared to traditional keyword based methods for finding similar Web services.

Thus, from Table 2 and Fig 3, it is clear that the support-based selection method outperforms random selection and any standard information retrieval method in terms of accuracy.

The selection of the lower-size dimension (k) with which the semantic kernel is created is crucial because Web services encompass various domains, and the constructed kernel should be able to find the semantics variances in all WSDL documents. Experiments have performed to find out the trade-off between accuracy (precision, recall, F-score) and computation constraint (response time, disk space).

From Table 3 it is evident that a value of k=300 performs well in terms of F-score as well as average response time and disk space requirement. Even though k=700, performs best in terms of F-Score value, the computation is too expensive considering the time and resource constraint. Hence, for all the experiments, k=300 have been chosen for creating the semantic kernel.

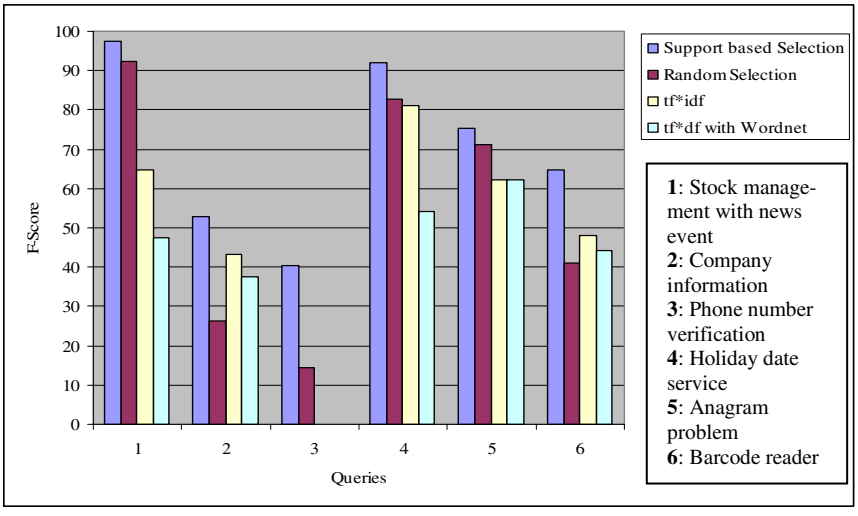


Fig. 3. Comparing F-Score values of different methods for a set of queries

Table 3. Response time and required disk space along with F-Score for different value of U_k

Kernel (U_k)	Precision		Recall	F Score		Average Response time in seconds	Disk Space in MB
	Top 20	Top 10		Top 20	Top 10		
200	29.1	37.2	66.05	40.4	47.594	57.27	63.3
300	**31**	**38.4**	**71.61**	**43.27**	**49.99**	**84.66**	**95**
400	30.2	37.6	70.72	42.33	49.10	102.13	126
500	29.9	38	70.00	41.90	49.26	121.96	158
600	30.1	38	70.27	42.15	49.33	146.4	189
700	30.6	38.6	71.63	42.88	50.17	164.8	221

One of the goals of the proposed method is to build a semantic kernel that is scalable and efficient for any given training documents. Semantic kernel has also been created without merging, by selecting a specified number of documents (Table 1) that a PC can handle to check whether dimensionality reduction by merging is having any positive effect. Table 4 compares the different types of kernels created without merging the documents but using the selective number of documents with different approaches (details in section 4.2). The best F-score value is obtained by using the documents with the document size based selection method. This is because it contains more words (Table 1) and hence is able to find semantically similar words in a better way. But even this result (F-Score value) is far below the results shown in Table 2. This shows that semantic kernels only improve the results in comparison to standard retrieval methods when the training documents used in constructing the kernel reflect the domains completely.

Support-based selection which is based on finding the support of a term (equation 4) and then calculating the importance of the document (equation 5) performs better than the rest, because the randomness induced by selecting documents arbitrarily is being removed**.** Reducing the dimensionality by random projection and then using SVD on the reduced matrix performs the worst.

Comparing the F-Score values in Tables 2 and 4, it is evident that merging is having a two fold positive effect: it is able to reduce the dimensionality and is able to find the similarity between Web services in more accurate way.

Table 4. Comparing different ways of creating kernel without merging with their Precision, Recall and F-Scrore measure

Methods	Types (Without Merging)	Sub-types	Precision		Recall	F Score	
			Top 20	Top 10		Top 20	Top 10
Latent Semantic Kernel	Document Size Based Selection	tf*idf	26.7	38.6	51.10	35.07	43.98
		Support	27.2	38.4	51.72	35.65	44.08
	Support Based Selection	tf*idf	22.9	30.2	43.26	29.95	35.57
		Support	22.6	28.6	41.51	29.27	33.87
	Arbitrary Document Selection	tf*idf	20.56	26.48	38.75	26.87	31.46
		Support	20.44	26.36	38.85	26.79	31.41
	Random Projection	tf*idf	2.8	3.4	6.52	3.92	4.47
		Support	2.8	3.8	6.78	3.96	4.87

Based on experiments it can be said that the proposed Web service discovery method using support-based semantic kernel with reduced matrix size performs better compared to traditional retrieval methods to find most relevant services. Also the proposed method is very efficient in terms of scalability and thereby offering a two fold advantage in terms of accuracy and efficiency.

5 Conclusion and Future Work

In this paper, we have proposed a novel approach to find semantically similar Web services for a user request using the support-based semantic kernel with reduced matrix size. Results clearly show that the accuracy of Web service discovery has improved and the proposed method outperforms traditional keyword based methods to find most relevant Web services. The approach which is based on an innovative concept of dimensionality reduction by binning & merging has been evaluated thoroughly. Experiments ascertain that the kernels created with reduced size term-document matrix using binning & merging are able to explore semantic relationships in Web services in a more efficient way in comparison to the kernel built with the same number of selected documents.

The proposed methodology can be used to enhance the search mechanism in UDDI registry. Existing Web service discovery process in UDDI registry is a basic key-word based method. Performing semantic analysis by using the proposed support based latent semantic kernel will help to find semantically similar Web services by exploring the hidden meaning of the query terms. This approach will increase the accuracy of discovering semantically similar Web services to fulfill the requirement of the user.

In future, we plan to extend this approach to link semantically similar Web services so that a Web service which can partially fulfil what a user is looking for can be linked with another Web service so as to achieve the overall objective of the user and thus increasing the overall accuracy of Web service discovery.

References

[1] INEX: INitiative for the Evaluation of XML Retrieval, http://inex.is.informatik.uniduisburg.de/2007/

[2] XMethods, http://www.xmethods.net/ve2/index.po

[3] Wordnet, http://wordnet.princeton.edu/

[4] Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.-T., Amit Sheth, L.L., Kunal Verma, L.L.: Web Service Semantics - WSDL-S (2005)

[5] Al-Masri, E., Mahmoud, Q.H.: QoS-based Discovery and Ranking of Web Services. In: IEEE 16th International Conference on Computer Communications and Networks (ICCCN), pp. 529–534 (2007)

[6] Chaiyakul, S., Limapichat, K., Dixit, A., Nantajeewarawat, E.: A Framework for Semantic Web Service Discovery and Planning. In: IEEE Conference on Cybernetics and Intelligent Systems, p. 5 (2006)

[7] Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language (WSDL) 1.1. In: World Wide Web Consortium (2001)

[8] Dong, X., Halevy, A., Madhavan, J., Nemes, E., Zhang, J.: Similarity Search for Web Services. In: 30th VLDB Conference, Toronto, Canada (2004)

[9] Fan, J., Ren, B., Xiong, L.-R.: An Approach to Web Service Discovery Based on the Semantics. In: Wang, L., Jin, Y. (eds.) FSKD 2005. LNCS (LNAI), vol. 3614, pp. 1103–1106. Springer, Heidelberg (2005)

[10] Klusch, M., Fries, B., Sycara, K.: Automated Semantic Web Service Discovery with OWLS-MX. In: Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, Hakodate, Japan, pp. 915–922 (2006)

[11] Lamparter, S., Schnizler, B.: Trading Services in Ontology-driven Markets. In: 2006 ACM symposium on Applied computing, Dijon, France, pp. 1679–1683 (2006)

[12] Landauer, T.K., Foltz, P.W., Laham, D.: An Introduction to Latent Semantic Analysis. Discourse Processes 25, 259–284 (1998)

[13] Lin, J., Gunopulos, D.: Dimensionality Reduction by Random Projection and Latent Semantic Indexing. In: Third SIAM International Conference on Data Mining, San Francisco, CA, USA (2003)

[14] Ma, J., Cao, J., Zhang, Y.: A Probabilistic Semantic Approach for Discovering Web Services. In: 16th International Conference on World Wide Web, Banff, Alberta, Canada, pp. 1221–1222 (2007)

[15] Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K.: OWL-S: Semantic Markup for Web Services. In: W3C (2004)

[16] Nayak, R.: Using Data Mining In Web Services Planning, Development and Maintenance. International Journal of Web services Research 5, 62–80 (2008)
[17] Nayak, R., Lee, B.: Web Service Discovery with additional Semantics and Clustering. In: 2007 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2007), Silicon Valley, USA (2007)
[18] Paliwal, A.V., Adam, N.R., Xiong, H., Bornhovd, C.: Web Service Discovery via Semantic Association Ranking and Hyperclique Pattern Discovery. In: 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 649–652 (2006)
[19] Paolucci, M., Srinivasan, N., Sycara, K., Nishimura, T.: Toward a Semantic Choreography of Web services: from WSDL to DAML-S. In: International Conference on Web Services (ICWS) (2003)
[20] Papadimitriou, C.H., Raghavan, P., Tamaki, H., Vempala, S.: Latent Semantic Indexing: A Probabilistic Analysis. In: Seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems Seattle, Washington, United States, pp. 159–168 (1998)
[21] Pathak, J., Koul, N., Caragea, D., Honavar, V.G.: A Framework for Semantic Web Services Discovery. In: 7th Annual ACM International Workshop on Web Information and Data Management, Bremen, Germany, pp. 45–50 (2005)
[22] Sajjanhar, A., Hou, J., Zhang, Y.: Algorithm for Web Services Matching. Advanced Web Technologies and Applications, pp. 665–670 (2004)
[23] Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24, 513–523 (1988)
[24] Sapkota, B., Roman, D., Ryszard, S., Fensel, K.D.: Distributed Web Service Discovery Architecture. In: Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006) (2006)
[25] Yu, J., Su, H., Zhou, G., Xu, K.: SNet: Skip Graph based Semantic Web Services Discovery. In: 2007 ACM Symposium on Applied computing, Seoul, Korea, pp. 1393–1397 (2007)

BPEL4RBAC: An Authorisation Specification for WS-BPEL

Xin Wang[1], Yanchun Zhang[1], Hao Shi[1], and Jian Yang[2]

[1] School of Computer Science and Mathematics
Victoria University, Australia
xin@csm.vu.edu.au, {Yanchun.Zhang,Hao.Shi}@vu.edu.au
[2] Department of Computing, Macquarie University,
Sydney, NSW2109, Australia
jian@ics.mq.edu.au

Abstract. Business process management is designed to make business activities and trade easier and more cost effective. The increasing business integration and legal requirements raise the need for secure business processes. However, the openness and distribution nature of inter-organisational business processes may result in more security breaches. As a widely accepted standard, WS-BPEL does not support for business process security protection even if the participating organisations already have working security policies. To address this problem, we have developed an authorisation specification BPEL4RBAC for WS-BPEL. Through BPEL4RBAC access control model, with an extension for WS-BPEL, called BPEL4RBAC policy language, the secure WS-BPEL is then achievable. The former introduces the access control capability into business process environment while the latter is used to represent the authorisation information in WS-BPEL.

1 Introduction

In today's business world, with the development of globalisation and the constant optimising of enterprise business process, organisations become more dynamic and the underlying business process are frequently changing [1]. Automating business services on demand and adapting market changes are main necessities to facilitate collaboration among business partners. As Web services become widely accepted, business process is taken as a new paradigm for business collaboration from different organisations instead of middleware [2]. In business world, a unified process specification language is significantly crucial in terms of collaboration. WS-BPEL is one such language that provides the syntax for specifying business processes behaviour based on Web services.

The security of business process is crucial for the business success as security problems would affect companies and their stakeholders in terms of profit and reputation. In most cases, business process management approaches and security solutions are developed separately [3]. Existing BPM methodologies seldom consider security issues while increasing business integration and legal requirements

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 381–395, 2008.

raise the need for secure business processes. Moreover, the security concern is enforced with the growing integration among organisations. The openness and distribution nature of business processes results in more security breaches [3]. The WS-BPEL language itself does not support security protection even if the participating organisations already have working security policies.

Access control and authorisation concerns are one of the major challenges in business systems. Role based access control (RBAC) [4] has emerged in 1990s in order to solve this kind of problems. However, as a widely accepted security paradigm, RBAC is not realisable to apply role based model to business process systems directly. For instance, the inherited roles might be stored remotely and permissions constraints will consequently require several remote invocations [5]. In nature, the business process and Web services environment is typically dynamic and distributed. The adaptive access control models are required to reinforce security features of business process systems [6].

To address these problems, this paper provides a theoretical foundation for realising effective access control in BPM systems that can adequately meet the distinctive security challenges in Web services environment. We introduce BPEL4RBAC, an authorisation specification, to provide access control and authorisation constraints ability to existing WS-BPEL standard. The remainder of this paper is organised as follows: Section 2 describes the related works. The access control requirements are illustrated with a running example in Section 3. Section 4 introduces the BPEL4RBAC model and policy language in detail. Section 5 compares other research works with BPEL4RBAC. The last section, Section 6, summarises the contribution and discusses future works.

2 Related Works

Balancing business collaboration and system security are competing goals [7]. Business applications contain information with variable levels of sensitivity in nature. However, in the real world, business activities are highly unpredictable comparing with single user applications [8]. In contrast, the open access in business process requires higher level of integrity and confidentiality.

2.1 Role Based Access Control

Access control mechanism aims at protecting information at different levels of granularity by configuring security policies [9]. In RBAC, the security policy does not directly grants permissions to users but assigned to appropriate roles on the basis of specific policy [10]. The assignment of users to roles is separated from the assignment of permissions to roles [5].

Several constraints may apply to an RBAC model. For example, Separation of Duty (SoD) is one of the well-known security principles. By partitioning related tasks and privileges, SoD reduces the possibility of fraud or errors. To protect the interest of organisations, the conflicting roles must not be assigned to the same user in a business process [11]. While in some other cases, the same user is required to perform two different activities. This is considered as a binding

of duty constraint. The security policy is embodied in RBAC to specify these access control constraints.

Although the concept of role has existed for a long time in systems security, the work presented by Sandu in [4] described this approach in detail. RBAC model is now adopted in many commercial products since access control is an important requirement of information systems. RBAC was found to be the most attractive solution for providing security characteristics in inter-organisational business systems [12]. Moreover, it would be much easier for organisations to enhance security protection from existing RBAC based systems.

2.2 WS-BPEL and Web Service Security

Service-oriented methodologies, associated with XML related technologies and standards, are applied to facilitate business Service-orientated methodologies, associated with XML related technologies, are applied to facilitate business collaboration with partners and customers [13]. This emerging paradigm provides loosely coupled and distributed business services across organisational boundaries [14].

Compared with traditional business applications, Web services aggregate isolated business functionalities in a standardised way to achieve a significant reduction in development cost and easier deployment for participating business partners [15]. Business process based collaboration is constructed by combining Web services through one of the process specification languages. WS-BPEL 2.0[16] fills this requirement gap and covers the ideas of two rivals, WSFL [17] and XLANG [18], developed by IBM and Microsoft respectively from 2001. WS-BPEL, initially named BPEL4WS, is built on top of several Web services and XML standards, including SOAP[19], WSDL[20], UDDI[21], XML Schema[22] and XPath[23].

The existing security standards for Web services should be also taken into consideration when providing security features for WS-BPEL. A variety of security standards have been proposed for Web service architecture at different levels. WS-Security[24] is the foundation for building secure Web services. It aims to realise message-level security for exchanging SOAP messages. Based on WS-Security, WS-Policy[25] provides a general purpose model and corresponding syntax for expressing Web services policies. The WS-Policy is constructed by a set of messaging-related assertions. The assertions can be defined in a set of security policy assertions related to supporting the WS-Security specification.Besides WS-Policy based architecture, there are some other XML-based languages that can be used to express Web services policies, such as SAML[26] and XACML[27]. With these languages we can specify access control rules that protect Web services from unauthorized access and ensure integrity and confidentiality of exchanged messages[11].

3 Business Process and Access Control

In this section, we illustrate WS-BPEL and RBAC model separately by running examples. For easy understanding, we use a common scenario: bank loan

```
<!-- WSBPEL syntax for loan process -->
<process name = "loanApprovalProcess"
  targetNamespace = "http://myloan.example.com/loanprocessing"...>
  <!-- Define the partners involved in -->
  <partnerLinks>
    <!--Customer submit application,Agency provides credit rating,
      Approver makes decision-->
    <partnerLink name="customer"  partnerLinkType="lns:loanPartnerLT"
      myRole="loanService"/>...
  </partnerLinks>
  <!-- Variables -->
  <variables>
    <variable name="loanRequest" messageType="lns:loanRequestMessage" />...
  </variables>
  <!-- receive loan request from customer -->
  <sequence>
    <receive partnerLink = "customer" portType = "lns:loanPartnerLT"
      operation="request" variable="loanRequest" createInstance="yes">
    </receive>
    <assign>
      <copy>
        <from partnerLink="customer"/><to variable="loanRequest"/>
      </copy>
    </assign>
    <flow>
      <links><link name = "receive-to-assess" />  ... </links>
      <!-- creditCheck, high risk go to approver -->
      <invoke partnerLink = "creditCheck" portType = "lns:creditCheckLT"
        operation="checkCredit" inputVariable="loanRequest"
        outputVariable="creditRequest">
        <targets> <target linkName="receive-to-assess"/></targets>
        <sources>
          <source linkName="assess-to-approval">
            <transitionCondition> $loanRisk.level='low'
            </transitionCondition></source>
          <source linkName="assess-to-approver">
            <transitionCondition> $loanRisk.level!='low'
            </transitionCondition>
          </source>
        </sources>
      </invoke>
      <!--Approver makes decision-->
      <invoke partnerLink = "approver"
        portType = "lns:loanApprovalLT" operation="approve"
        inputVariable="creditRequest" outputVariable="loanDecision">
        <targets><target linkName="approver-to-approval" /></targets>
        <sources><source linkName="approval-to-reply" /></sources>
      </invoke>
      <!--Reply to costomer-->
      <reply partnerLink="customer"  portType="lns:loanPartnerLT"
        operation="response"  variable="loanDecision">
        <targets>
          <target linkName="approver-to-reply" />
          <target linkName="approval-to-reply" />
        </targets>
      </reply>
    </flow>
  </sequence>
</process>
```

Fig. 1. BPEL Code for Bank Loan Process

application. The processes in this scenario are quite straightforward. First, a customer applies for bank loan. Then the bank conducts credit check according to this applicant and the risk level is also assessed comprehensively. Finally the loan application is approved to the eligible applicant while unacceptable high risk applications are rejected. In this scenario, the business processes are formalised by original WS-BPEL 2.0 code. The access control requirements and constraints are described in plain language.

3.1 Business Process in BPEL

WS-BPEL is designed to describe business processes in a structured way. The business logic is expressed as a group of activities and performed by invoking web services. The <process> element is on the top level of WS-BPEL specification. The attributes of <process> specifies the process name and related namespace. The <partnerLink> element indicates the external Web services to be invoked from this process. The <variable> element defines the data variables involved in this process. The <sequence> element contains sequentially executed activities while the <flow> element specifies concurrently performed activities. These elements may contain one or more basic elements such as <receive> and <reply> which define the message flows in a process. In bank loan application scenario, we can divide the whole process into six activities:

1. The customer applies for bank loan (apBL)
2. Loan officer conducts credit check based on customer's application (ccAP)
3. Low risk application is automatically approved (rkBL)
4. High risk application is re-assessed by loan admin or bank manager (reAP)
5. Some high risk applications are rejected while others are approved based on reassessment result. (dcBL)
6. The application result is sent to customer by loan officer. (rtBL)

Based on this process, we can work out the WS-BPEL code for bank loan application, as shown in Fig.1. The <partnerLinks> indicates the participators in the bank loan application process: the customer, the credit agency who conducts the credit check and the approver. In <sequence>, the bank loan application process is conducted in the following order: the <receive> designates loan request received from a customer, the credit check is invoked by <invoke> element, the risk level is described by <transitionCondition>, the approver makes a decision for the loan application by another <invoke>, finally the loan response is sent back to the customer by <reply>.The flowchart of this process is shown in Fig.2.

3.2 Access Control with RBAC

Although RBAC have been implemented by varieties of applications and can be represented in many ways, we choose to express our example in plain English

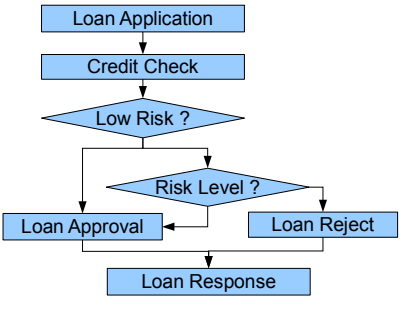


Fig. 2. Bank Loan Application Flowchart

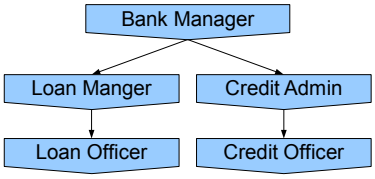


Fig. 3. Role Hierarchy in a Bank

to provide a universal understanding in this example. We also strictly adhere to the original RBAC to avoid any specific problems in particular systems.

In the bank loan application process, we can set some practical permissions and constraints according to RBAC model. Since the role is the core element in RBAC, we first describe the role hierarchy of a bank in the Fig.3.

As we discussed above, the main idea in RBAC considers simple constraints that can be effectively checked and implemented. We can describe the access control requirements and constraints as follows:

1. Role assignment and permissions:
   (a) Bank Manager is on top of the role hierarchy. Bank Manager governs Loan Manager and Credit Admin. Loan Manger governs Loan Officer while Credit Admin governs Credit Officer.
   (b) Loan Manager role and Credit Admin role can only be assigned to department managers.
   (c) Only Loan Officer is permitted to handle loan application.
   (d) Only Credit Officer is permitted to conduct credit check.
   (e) High risk loan applications must proceed to Loan Manager.
   (f) Loan Officer provides final result to the applicant.
2. Mutually exclusive roles:
   (a) The Loan Officer, who receives loan application from customer A, must be the Loan Officer provides loan response to customer A.
   (b) Loan Officer and Credit Officer must be assigned to different staff.
   (c) Loan Officer and Loan Manager must be assigned to different staff.
3. Cardinality:
   (a) There must be at least one staff assigned as Loan Officer.
   (b) There must be one staff and only one assigned as Bank Manager.
4. Prerequisite roles:
   (a) Assign a staff to Credit Officer only when new application received.
   (b) Assign a manager to Loan Manager only when application is rated as high risk.

4 BPEL4RBAC Model and Policy Language

We extend from original RBAC model to provide access control and authorisation constraints ability to existing WS-BPEL. This extended RBAC model is called BPEL4RBAC model in our proposed architecture. Moreover, in order to provide WS-BPEL compatible access control policy specification, we facilitate the WS-BPEL extension mechanism to build up our BPEL4RBAC policy language. Firstly, we provide formal definition of BPEL4RBAC model. Then, the bank loan application process is represented in this way as a running example. The BPEL4RBAC policy language is introduced in next subsection.

4.1 BPEL4RBAC Model

We extend existing RBAC model with considerations of business process. Fig.4. shows RBAC4BPEL model. In BPEL4RBAC, a user is human being belongs to an organisation. A role is a named job function within the business process context that regards the authority and responsibility. A permission is an approval of actions granted to specific roles. A constraint regulates the relations between different elements.

In order to provide access control capability to WS-BPEL, we add two elements to original RBAC model, namely organisation and business process. An organisation is a group of users with structure of roles and responsibilities functioning to participating business processes. A business process is "a set of logically related tasks performed to achieve a well defined business outcome" [12]. User assignment (UA) and permission assignment (PA) are both many-to-many relationship since a user can be assigned to many roles and a role can have one

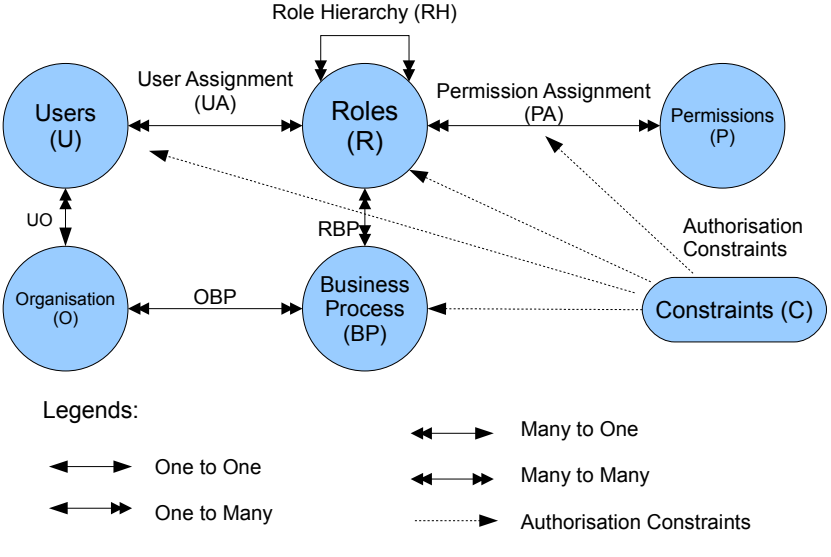


Fig. 4. BPEL4RBAC Model

or more users. Role hierarchy (RH) maps the nature structure of an organisation. User organisation (UO) relationship indicates which user belongs to which organisation. Organisation business process (OBP) relationship specifies which business process is developed or consumed by which organisation. Role business process (RBP) relationship describes which role is involved in which business process. We can define the RBAC4BPEL model as follows:

Definition 1. *Basic Elements:*

- $U = \{u_1, u_2, u_3, \cdots\cdots, u_i\}$, *set of users;*
- $R = \{r_1, r_2, r_3, \cdots\cdots, r_j\}$, *set of roles;*
- $P = \{p_1, p_2, p_3, \cdots\cdots, p_j\}$, *set of permissions;*
- $UA \subseteq U \times R$, *a many-to-many user to role assignment relation;*
- $PA \subseteq P \times R$, *a many-to-many permission to role assignment relation;*
- $RH \subseteq R \times Ra$ *partial order on R, represents the role hierarchy.*

Definition 2. *BPEL4RBAC Business Process:*

- $BP = \{bp_1, bp_2, bp_3, \cdots\cdots, bp_m\}$, *set of business processes;*
- $BA = \{ba_1, ba_2, ba_3, \cdots\cdots, ba_n\}$, *set of business activities;*
- $BPBA \subseteq BP \times BA$, *a many-to-many business process to business activities assignment relation, Where* $bpba = \{(ba, bp) \in BPBA \mid ba \in BA, bp \in BP\}$;
- $bp : BP \rightarrow 2^{BA}$, *a function mapping each business process (bp) to the set of business activities;*
- $RBP \subseteq R \times BP$, *a many to many role to business process, Where role :* $BP \rightarrow 2^R$ *is a function mapping each business process to the set of roles, Where* $roles(r_i) = \{r \mid (u_i, r_j) \in UA\}$.

Definition 3. *BPEL4RBAC Organisation:*

- $O = \{o_1, o_2, o_3, \cdots\cdots, o_l\}$, *set of organisations;*
- $UO \subseteq U \times O$, *a many to one user to organisation assignment relation, Where* $user : U \rightarrow O$ *is a function mapping user* (u_i) *to organisation* (o_l), *Where* $uo = \{(u, o) \in UO \mid u \in U, o \in O\}$;
- $OBP \subseteq O \times BP$, *a many to many organisation to business process assignment relation, Where* $o : BP \rightarrow 2^O$ *is a function mapping organisation* (o_l) *to business process* (bp_m), *Where* $obp = \{(o, bp) \in OBP \mid o \in O, bp \in BP\}$.

4.2 BPEL4RBAC Policy Language

As an extension to WS-BPEL, BPEL4RBAC policy language is layered on top of WS-BPEL. Its features can be aggregated with WS-BPEL features during the business processes. The extension introduces a set of elements to provide role based access control capability. The root element in BPEL4RBAC is <policy>. The basic elements in BPEL4RBAC language are <user>, <role>, <permission>, <organisation>, <business process> and <constraints>. In order to differentiate BPEL code and BPEL4RBAC extension, we use "b4r" prefix to indicate BPEL4RBAC namespace and "bpel" prefix to designate BPEL code.

With the extensibility of WS-BPEL, the <bpel:extensions> element imports the BPEL4RBAC extension to BPEL code. The <b4r:policy> is the root element of proposed extension. All other elements and activities in BPEL4RBAC are enclosed. The overall syntax is shown in Fig. 5:

```
<bpel:process name = "NCName" ...
 xmlns:b4r = "http://myloan.example.com/bpel4rbac"...>
 <bpel:extensions
  namespace = "http://myloan.example.com/bpel4rbac"
  mustUnderstand = "yes">
 </bpel:extensions>
 <bpel:extensionActivity>
  <b4r:policy>
   <b4r:roles>
    <b4r:role ID = "NCName">role</b4r:role>+
   </b4r:roles>
   <b4r:roleHierarchys>
    <b4r:roleHierarchy ID = "NCName">+
     <b4r:seniorRole>senior role</b4r:seniorRole>
     <b4r:juniorRole>junior role</b4r:juniorRole>
    </b4r:roleHierarchy>
   </b4r:roleHierarchys>
   <b4r:userAssignments>
    <b4r:userAssignment ID = "NCName">+
     <b4r:role>role</b4r:role>
     <b4r:assignment>user to role assignment</b4r:assignment>
    </b4r:userAssignment>
   </b4r:userAssignments>
   <b4r:permissions>
    <b4r:permission ID = "NCName">+
     <b4r:role>role</b4r:role>
     <b4r:permittedActivity>permitted activity for this role
     </b4r:permittedActivity>
    <b4r:bpelActivity>related BPEL activity</b4r:bpelActivity>
    </b4r:permission>
   </b4r:permissions>
   <b4r:constraints>
    <b4r:constraint ID = "NCName">+
     <b4r:object>object name</b4r:object>
     <b4r:activity>business process activity</b4r:activity>?
     <br4:bpelActivity>related BPEL activity</br4:bpelActivity>?
     <b4r:consequentObject>object name</b4r:consequentObject>?
     <b4r:subsequentActivity>?
      business process activity
     <b4r:subsequentActivity>
     <br4:bpelActivity>related BPEL activity</br4:bpelActivity>?
     <b4r:constraintAssertion>condition</b4r:constraintAssertion>
    </b4r:constraint>
   </b4r:constraints>
   <b4r:businessProcesses>
    <b4r:businessProcess ID ="NCName">+
     <b4r:activity>business process activity</b4r:activity>
     <br4:bpelActivity>related BPEL activity</b4r:bpelActivity>
     <bpel:partnerLink name = "NCName"/>
    </b4r:businessProcess>
   </b4r:businessProcesses>
  </b4r:policy>
 </bpel:extensionActivity>
</bpel:process>
```

Fig. 5. BPEL4RBAC Code

Role Hierarchy. The <b4r:roles>, <b4r:roleHierarchys>, <b4r:userAssignments>, <b4r:permissions>, <b4r:constraints> and <b4r:businessProcesses> indicate group of roles, role hierarchy, user assignment, permissions, constraints and business processes respectively. Table 1(a) and 1(b) illustrate the roles and role hierarchy in the bank loan application example. The <b4r:role> element

is used to define a role in an organisation while the <b4r:roleHierarchys> element specifies the role hierarchy. The roles definition hierarchy can be encoded as:

```
<b4r:roles>
    <b4r:role ID = "R1">Bank Manager</role>
    ...
    <b4r:role ID = "R5">Loan Officer</role>
</b4r:roles>
<b4r:roleHierarchys>
    <b4r:roleHierarchy ID = "RH1">
        <b4r:seniorRole>Bank Manager</b4r:seniorRole>
        <b4r:juniorRole>Loan Manager</b4r:juniorRole>
    </b4r:roleHierarchy>
    ...
</b4r:roleHierarchys>
```

User Assignment and Permissions. The <b4r:userAssignment> element defines user assignment in a business process while <b4r:permission> element describes the permission imposed on specific role or activity. The user assignment

Table 1. Roles and Role Hierarchy

Role ID	*Role Name*
R1	Bank Manager
R2	Loan Manager
R3	Credit Admin
R4	Credit Officer
R5	Loan Officer

(a) Roles

RH ID	*Hierarchy*
RH1	Bank Manager > Loan Manager
RH2	Bank Manager > Credit Admin
RH3	Loan Manager > Loan Officer
RH4	Credit Admin > Credit Officer

(b) Role Hierarchy

UA ID	*Role*	*Position*
UA1	Loan Manager	only to department manager
UA2	Credit Manager	only to department manager

(c) User Assignment

Pm. ID	*Role*	*Activity*	*BPEL Activity ID*
P1	Loan Officer	handle loan application	apBL
P2	Credit Officer	conduct credit check	ccAP
P3	Loan Manager	handle high risk loan application	reAP
P4	Loan Officer	provide final result to applicant	rtBL

(d) Permissions

and permissions illustrated in Table 1 (c) and (d), which illustrate the user assignment and permissions in this example, can be encoded as:

```
<b4r:userAssignments>
    <b4r:userAssignment ID = "UA1">
        <b4r:role>Loan Manager</b4r:role>
        <b4r:assignment>Only to department managers</b4r:assignment>
    </b4r:userAssignment>
    ...
</b4r:userAssignments>
<b4r:permissions>
    <b4r:permission ID = "P1">
        <b4r:role>Loan Officer</b4r:role>
        <b4r:permittedActivity>handle loan application
        </b4r:permittedActivity>
        <b4r:bpelActivity>apBL</b4r:bpelActivity>
    </b4r:permission>
    ...
</b4r:permissions>
```

Constraints. The <b4r:constraint> element is used to define access constraint in business process. The <b4r:object> element and <b4r:consequentObject> element designate the role or user whom the constraint applies on. The <b4r:activity> element and <b4r:subsequentActivity> element indicate which activity or process is restrained by this constraint. The <b4r:bpelActivity > element describes the association between the activity or process in WS-BPEL with the activity in BPEL4RBAC.

The <b4r:constraintAssertion> is the assertion describes the constraint condition. The <b4r:object> and <b4r:constraintAssertion> elements are compulsory while other elements are optional according to the constraint. The following code illustrates the constraints in Table 2:

Table 2. Access Control Constraints

Const. ID	*Object(Activity)*	*Consequent Object (Subsequent Activity)*	*Condition*	*Related BPEL Activity ID*
C1	Loan Officer (Handles loan application from customer A)	Loan Officer (Provides response to customer A)	Same User	apBL, rtBL
C2	Loan Officer	Credit Officer	Different User	apBL, ccAP
C3	Loan Officer	Loan Manager	Different User	rkBL, reAP
C4	Loan Officer		At least one user assigned to this role	apBL
C5	Bank Manager		Only one user assigned to this role	
C6	Credit Officer (A staff assigned to this role)		Only when a new loan application received	rkBL
C7	Loan Manager (A staff assigned to this role)		Only when application is rated as high risk	reAP

```
<b4r:constraints>
    <b4r:constraint ID = "C1">
        <b4r:object>Loan Officer</b4r:object>
        <b4r:activity>
            Handles loan application from customer A
        </b4r:activity>
        <br4:bpelActivity>apBL</b4r:bpelActivity>
        <b4r:consequentObject>Loan Officer</b4r:consequentObject>
        <b4r:subsequentActivity>
            Provides loan response to customer A
        </b4r:subsequentActivity>
        <br4:bpelActivity>rtBL</b4r:bpelActivity>
        <b4r:constraintAssertion>Same User</b4r:constraintAssertion>
    </b4r:constraint>
    ...
    <b4r:constraint ID = "C7">
        <b4r:object>Loan Manager</b4r:object>
        <b4r:activity>A staff assigned to this role<b4r:activity>
        <br4:bpelActivity>reAP</b4r:bpelActivity>
        <b4r:constraintAssertion>
            Only when loan application is rated as high risk
        </b4r:constraintAssertion>
        </b4r:constraint>
</b4r:constraints>
```

The <b4r:businessProcess> element associates the activity in WS-BPEL and BPEL4RBAC by <b4r:activity>, <br4:bpelActivity> and <bpel:partnerLink> elements correspondingly. The following is an illustrating example code:

```
<b4r:businessProcesses>
    <b4r:businessProcess ID ="BP1">
        <b4r:activity>Handle loan application</b4r:activity>
        <br4:bpelActivity>apBL</b4r:bpelActivity>
        <bpel:partnerLink name = "customer"/>
    </b4r:businessProcess>
    ...
    <b4r:businessProcess ID ="BP6">
        <b4r:activity>Send application result to customer
        </b4r:activity>
        <br4:bpelActivity>rtBL</b4r:bpelActivity>
        <bpel:partnerLink name = "customer"/>
    </b4r:businessProcess>
</b4r:businessProcesses>
```

4.3 BPEL4RBAC System Architecture

In accordance to our proposed specification, the BPEL4RBAC-based system architecture is illustrated in detail in this section. Since BPEL4RBAC extends WS-BPEL, its architecture is WS-BPEL enabled and compatible with existing Web services standards.

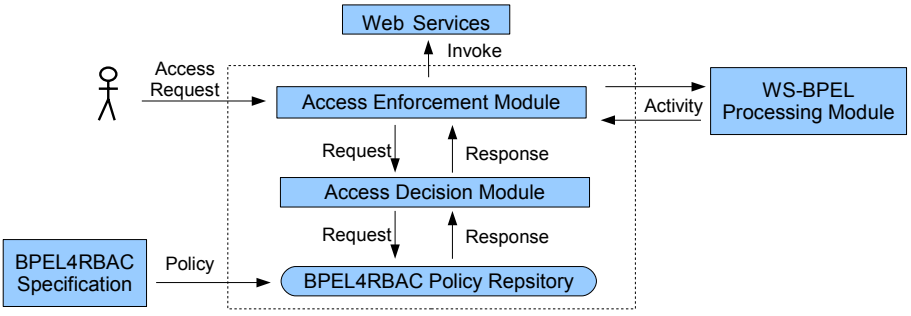


Fig. 6. BPEL4RBAC System Architecture

The Access Enforcement Module (AEM) is the key component of the entire system. It handles user request, them AEM forwards the request to Access Decision Module (ADM). After the response received from ADM, AEM contacts the WS-BPEL module to get the information for requested business activities. Then the corresponding Web services are invoked by AEM.

After received the request from AEM, ADM contacts BPEL4RBAC policy repository and perform a check against the user who send this request. The decision is made according to the pre-defined BPEL4RBAC policy which is represented by BPEL4RBAC model.

Based on BPEL4RBAC policies, AEM and ADM work together to ensure the integrity and confidentiality of WS-BPEL represented business processes. Taking the advantage of easy integration with WS-BPEL, BPEL4RBAC can also integrated with message level security standards, such as WS-SecurityPolicy and XACML that we have discusses in Section 3.2.

Another benefit comes from the modular design of the system architecture. The BPEL4RBAC policy specification is separated from access decision module. The access decision processing mechanisms for ADM are more flexible with the same security policy. The AEM is separated from ADM as well. Thus the AEM focuses only on coordination with Web services and access control enforcement without touching about any change in access decision strategies and security policies.

5 Comparison of Related Works

There are many research works [4][11] focus on enhancing security features for BPM. Some of these works addressed on access control ability to current WS-BPEL.

Bertino, Crampton and Paci [4] developed RBAC-WS-BPEL and BPCL languages. The RBAC-WS-BPEL is an adapted version of RBAC model with business process element introduced within. The authorisation specification is composed from authorisation schema, represented by XACML, and authorisation constraints, represented by BPCL which is an XML based language.

However, XACML does not directly support the notation of roles, and lacks some essential features in RBAC such as separation of duty and role hierarchy.

Liu and Chen [11] developed an extended RBAC model, WS-RBAC. Three new elements are introduced into the original RBAC model, namely enterprise, business process and Web services. The authorisation constraints are described in WS-Policy and WS-PolicyAttachment. However, these two standards are designed for message level security. It is difficult to express some access control constraints on this layer such as role hierarchy and permissions. This also increases the complexity of performing authorisation constraints.

Comparing with these works, BPEL4RBAC extends its ability from both RBAC side and WS-BPEL side. BPEL4RBAC is compatibility with WS-BPEL standard since BPEL4RBAC policy language is an extension of WS-BPEL. This ensures the access control functions can be seamlessly integrated into WS-BPEL. The system architecture also provides the adaptability with other security standards to enhance its security level further. BPEL4RBAC is able to be extended with other standards as long as they are compatible with WS-BPEL.

6 Conclusion and Future Works

In this paper, we propose BEPL4RBAC authorisation specification which supports the access control capability in business process environment.

The BPEL4RBAC extends the classical RBAC model with organisation and business process elements. These two elements are essential for representing access control information in business process scenario. The BPEL4RBAC policy language is also formally defined. The access control and authorisation requirements illustrated in BPEL4RBAC model can be mapped into this policy language. All these information are integrated with WS-BPEL seamlessly. The system architecture investigates the feasibility of BPEL4RBAC. With the separation of AEM and ADM modules, access decision strategies and security policies can be developed by physically isolated users or organisations. These strategies and policies might be changed frequently according to the real world need. Thus, BPEL4RBAC system ensures the availability in heavy duty business process environment.

We are extending this work in many tracks. The BPEL4RBAC policy language can be further improved by taking consideration of more complicated organisation behaviours. In particular, an organisation may define different roles and permissions to the same process when cooperating with different partners. Another direction intends to provide connectivity with message level security standards, such as XACML and WS-Policy as we discussed above.

References

1. Wei, X., Jun, W., Yu, L., Jing, L.: SOWAC: a service-oriented workflow access control model. In: Proceedings of the 28th Annual International Computer Software and Applications Conference. COMPSAC 2004, pp. 128–134 (2004)

2. Wang, X., Zhang, Y., Shi, H.: Scenario-Based Petri Net Approach for Collaborative Business Process Modelling. In: IEEE Asia-Pacific Service Computing Conference, pp. 18–25 (2007)
3. Neubauer, T., Klemen, M., Biffl, S.: Secure Business Process Management: A Roadmap. In: First International Conference on Availability, Reliability and Security (ARES 2006), Washington, DC, USA, pp. 457–464. IEEE Computer Society, Los Alamitos (2006)
4. Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-based access control models. IEEE Computer 29(2), 38–47 (1996)
5. Sloman, M., Lupu, E.: Security and management policy specification. Network, IEEE 16(2), 10–19 (2002)
6. Kim, H., Lee, R., Yang, H.: Frameworks for Secured Business Process Management Systems. In: Fourth International Conference on Software Engineering Research, Management and Applications (2006)
7. Tolone, W., Ahn, G., Pai, T., Hong, S.: Access control in collaborative systems. ACM Computing Surveys (CSUR) 37(1), 29–41 (2005)
8. Siponen, M., Oinas-Kukkonen, H.: A review of information security issues and respective research contributions. ACM SIGMIS Database 38(1), 60–80 (2007)
9. Wang, H., Cao, J., Zhang, Y.: A flexible payment scheme and its role-based access control. IEEE Transactions on Knowledge and Data Engineering 17(3), 425–436 (2005)
10. Kalam, A., Baida, R., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miege, A., Saurel, C., Trouessin, G.: Organization based access control. In: Proceedings of Policies for Distributed Systems and Networks. POLICY 2003. IEEE 4th International Workshop, pp. 120–131 (2003)
11. Liu, P., Chen, Z.: An Access Control Model for Web Services in Business Process. In: Proceedings of Web Intelligence, 2004. WI 2004, IEEE/WIC/ACM International Conference, pp. 292–298 (2004)
12. Yang, C.: Designing secure e-commerce with role-based access control. International Journal of Web Engineering and Technology 3(1), 73–95 (2007)
13. Wang, H., Zhang, Y., Cao, J., Varadharajan, V.: Achieving Secure and Flexible M-Services Through Tickets. IEEE Transactions On Systems, Man, And Cyberneticspart A: Systems and Humans 33(6), 697 (2003)
14. Chang, J.: Business Process Management System - Strategy and Implementation. Auerbach Publications (2006)
15. Papazoglou, M., Georgakopoulos, D.: Service-oriented computing: Introduction. Communications of the ACM 46(10), 24–28 (2003)
16. OASIS: Web Services Business Process Execution Language v2.0 (2007)
17. IBM: Web Services Flow Language (2001)
18. Corporation, M.: XLANG: Web Services for Business Process Design (2001)
19. W3C: SOAP Specification V1.2 (2007)
20. W3C: Web Services Description Language (WSDL) V1.1 (2001)
21. OASIS: UDDI Version 3.0.2 (2004)
22. W3C: XML Schema (2004)
23. W3C: XML Path Language (XPath) (1999)
24. OASIS: WS-Security Core Specification V1.1 (2006)
25. W3C: Web Services Policy 1.2 - Framework (WS-Policy) (2006)
26. OASIS: Security Assertion Markup Language (SAML) v2.0 (2005)
27. OASIS: eXtensible Access Control Markup Language TC v2.0 (XACML) (2005)

A Workflow-Based Approach for Creating Complex Web Wrappers*

Paula Montoto, Alberto Pan**, Juan Raposo, José Losada,
Fernando Bellas, and Javier López

Department of Information and Communication Technologies
University of A Coruña, Spain
{pmontoto,apan,jrs,jlosada,fbellas,jmato}@udc.es

Abstract. In order to let software programs access and use the information and services provided by web sources, wrapper programs must be built to provide a "machine-readable" view over them. Although research literature on web wrappers is vast, the problem of how to specify the internal logic of complex wrappers in a graphical and simple way remains mainly ignored. In this paper, we propose a new language for addressing this task. Our approach leverages on the existing work on intelligent web data extraction and automatic web navigation as building blocks, and uses a workflow-based approach to specify the wrapper control logic. The features included in the language have been decided from the results of a study of a wide range of real web automation applications from different business areas. In this paper, we also present the most salient results of the study.

Keywords: web wrappers, data mining, web automation, web information systems.

1 Introduction

Most of today's Web sources are designed to be easily used by humans, but they do not offer suitable interfaces to allow software programs to interact with them. During the last years, a growing interest has arisen in automating the interactions with web sites. Most previous research works in this field have focused on the concept of wrapper. A wrapper abstracts the complexities involved in automating a certain task on a web source, providing a programmatic interface to external applications. As related work, see [3],[4],[5],[9],[10],[11],[12],[13],[14],[15],[17],[18]. [8] provides a survey. Notice that, since data extraction from HTML pages is one of the key tasks involved in wrapper generation, some researchers use the term wrapper to mean only the extraction phase; in this paper we consider wrapper as a more general term implying all the tasks involved in web automation.

* This research was partially supported by the Spanish Ministry of Education and Science under project TSI2005-07730.
** Alberto Pan's work was partially supported by the "Ramón y Cajal" programme of the Spanish Ministry of Education and Science.

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 396–409, 2008.

Most previous works on web wrappers assume a particular underlying model we will call 'query wrapper' model. Query wrappers consider a web source as a special kind of database where queries can be posed using a form, obtaining a result set composed of structured data records. The query wrapper model assumes a pre-defined list of execution steps: first, a navigation sequence is used to automatically fill in some query form. Secondly, data extraction techniques are used to obtain the list of results from the response HTML pages. In addition, query wrappers may also support paginated result listings and accessing 'record detail' pages to extract further information.

While query wrappers are useful, they do not fit some important web automation applications. For instance, many tasks involve taking decisions depending on the retrieved data to continue the navigation in a way or another.

The objective of this paper is to propose a graphical language for creating wrappers able to automate any web automation process on a given website. It is important to note that, in our view, a wrapper automates the interaction with a single website and for a single purpose. For tasks involving the combination and/or orchestration of several web sources, our approach consists of enabling the wrappers to participate as basic components in usual data and process integration architectures such as data mediators [16] or Business Process Management systems. Other key objective for our language is being simple: wrappers should be created graphically and the language should not include features that introduce unnecessary complexity. Programming-skills should not be necessary in the majority of cases.

As a source of inspiration for our model, we have studied BPM orchestration patterns [1] and technologies (such as BPMN [7] and WS-BPEL [6]). As in our case, BPM technologies are also concerned with graphically specifying complex logic.

To base our proposal on firm roots, we have studied a wide range of real web automation tasks, which are being used by corporations from different business areas, trying to capture all the requirements needed in real applications. Some of the main results of the study are also reported in the paper. The language has been implemented in a fully functional prototype.

The paper is structured as follows. Section 2 reports some of the most important results and conclusions of the motivating study. Section 3 describes the proposed language. Section 4 discusses related work.

2 Motivation

To guide the development of the language, we have studied a wide range of real web automation applications. We have studied 391 wrappers belonging to 24 applications. These applications are a sub-set of those developed by a European enterprise specialized in web automation applications during the last three years. We have chosen applications in different business areas to increase the generality of our approach: B2B web automation (i.e. automating repetitive operations with other organizations through a web interface), batch data extraction, internet metasearch applications, web account integration, and technology and business watch (i.e. monitoring web information relevant for business and/or research purposes, such as competitor prices).

As a previous step before the study, we analyzed existing workflow technologies for BPM (such as [1],[6],[7]) to identify some features which could apply to web automation applications. Then we analyzed the wrappers in the study to check if they required or could benefit from them. The features we found useful are: conditional bifurcations, error management, parallelism, asynchronous events and sub-processes. We also analyzed if the wrappers conformed to the query wrapper model. Now, we report and discuss some of the main results of the study:

1. A first conclusion is that only the 57% of the wrappers conform to the *query wrapper* model. The percentage varies in function of the application type: 100% of the wrappers in metasearch applications conform to that model, while in B2B applications the percentage only reaches 53%. The study allows concluding that the query wrapper model is too simple for many real web automation applications.
2. Regarding *conditional bifurcations*, 54% of the wrappers that do not conform to the query wrapper model require them. Therefore, the language should support them.
3. Regarding *error management*, most of the wrappers considered require or could benefit from: 1) On the apparition of an error in the process, indicating which action to perform: either ignore it or halt the process and return the error to the invoking application, 2) Executing retries (e.g. when executing web navigation sequences). In addition, 37% of the wrappers in the B2B application area require or could benefit from user-defined, application-specific exceptions to return to the calling application. Therefore, the language includes support for these features.
4. Regarding *parallelism*, we observed it is very useful in two cases: 1) When a wrapper needs to process a list of records extracted from a web page, and the processing of each record involves executing one or more navigation sequences (e.g. accessing a detail page). Since navigation sequences can be relatively slow, processing the records in parallel can greatly improve performance, and 2) We have also observed that some applications execute the same query wrapper multiple times using different query parameters and then merge the obtained results. Therefore, it is useful to include in the language specific support to allow specifying the parallel execution of multiple queries on the same web form, thus alleviating the invoking application of this task. 84% of the studied wrappers could benefit from either one or both of these kinds of parallelism. On the other hand, no wrapper required other types of parallelism. Recall that, in our model, wrappers abstract the interactions with a single source for a given task. More room for parallelism would undoubtedly arise if we considered web automation tasks involving the combination and/or orchestration of several sources. Nevertheless, we follow the common approach in integration architectures of separating access and coordination layers. To coordinate and/or integrate several sources (which may or may not be web sources), our approach consists of enabling the wrappers to participate as components in usual data and process integration architectures such as data mediators [16] or Business Process Management systems. Therefore, we conclude the language should not include more general support for parallelism because it would considerably increase the complexity and its benefits would be unclear.

5. Regarding *asynchronous operations*, some web pages may dynamically change its content without requiring a full reload of the page (e.g. sources which use Javascript / AJAX technology to update its content). In these situations, it would be useful if the wrapper could be asynchronously notified of changes in the content of the areas of interest in the page. 6 wrappers in the study have to deal with these sources but, since AJAX sources are gaining prominence, we expect this feature to increase its importance. The wrappers we examined are not notified asynchronously. Instead, they access the target regions at specified intervals (i.e. "polling"). The reason is that most automatic navigation systems use browsers as basic components for navigating and hosting HTML pages, and with current browser APIs it is difficult to identify content-change events at the desired granularity. Polling can be easily supported by including a *wait* activity in the language.
6. Regarding *subprocesses*, we have detected that the most complex wrappers studied could be greatly simplified by using sub-processes. We have also observed there are several *structural patterns* that occur in a great number of wrappers (e.g. most wrappers have to deal with different types of results pagination, many wrappers need to poll a list of results for changes, etc.). These patterns appear many times, sometimes with slight variations. We conclude a desirable feature for the language is to allow creating reusable components to support them.

3 Language Description

In this section, we describe the proposed language. Section 3.1 describes the overall structure of a workflow in our approach. The pre-defined activities that can be used in the workflows are described in section 3.2. Section 3.3 describes how users can create reusable components. Section 3.4 presents an example.

3.1 Workflow Model

The data instances handled in the process flow (we will call them values) belong to a structured type [2]. A structured type can be atomic, a record type or a list type. The language specifies support for the atomic types commonly found in programming languages (*string*, *int*, *long*, *double*, *float*, *date*, *boolean*, *binary*, *url*, *money*) and for a specific type called *page*, which encapsulates the information needed to allow the web automation system to access a page: that is, an URL and the required cookies .

Fig. 1 shows an UML diagram describing the basic structure of the language. A workflow receives a set of variables as input parameters and returns a single variable as output. The value of a variable is an instance of a valid data type. The input parameters can be mandatory or optional. A workflow is composed of a set of ordered *activities*. Activities can be either basic or structured. Structured activities include those used for loops and bifurcations and enclose one ordered sequence of activities (bifurcations enclose one sequence for each execution path). A workflow can be seen simply as a sub-class of structured activity. Basic activities perform the actions in the workflow. Although not shown in the diagram due to space constraints, certain activities require some of the variables they use to be of a certain data type.

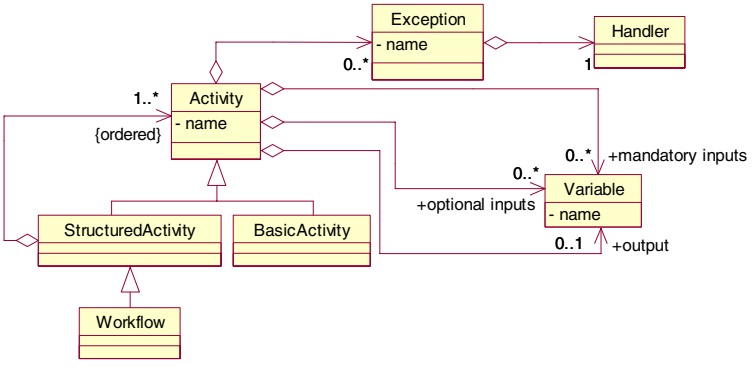


Fig. 1. Basic structure of the language

To handle error management, the language leverages on the concept of *exceptions*, that can be either pre-defined or user-defined. There exist pre-defined exceptions to represent generic runtime errors, typical errors produced while executing a navigation sequence (http error, timeout, connection error) and while extracting data records (e.g. a record does not match the expected type). User-defined exceptions are generated using the *THROW* activity. Each exception has assigned one in a set of pre-defined handlers. Handlers exist for throwing and ignoring the exception. It is also supported to configure retries before the error is handled.

3.2 Pre-defined Activities

This section describes the activities that can be used to create the workflows. We begin describing the basic activities and then describe the structured activities:

- *SEQUENCE*: An instance of the *SEQUENCE* activity executes a navigation sequence and returns a *page* value representing the final web page reached. Optionally, it can receive the following input parameters: 1) One *page* value. If provided, the page is loaded in the automatic navigation component before executing the sequence. This is useful if the configured sequence needs to start from a given page. 2) One or more values of either atomic or record type. These are needed because navigation sequences are often expressed in function of variables. For instance, a sequence automating a query on an Internet bookshop can receive as input the title and author to search. Our system uses an extension of the techniques proposed in [14] to implement the *SEQUENCE* activity, but any other method could be used.
- *EXTRACTOR*: An instance of the *EXTRACTOR* activity receives a *page* value and outputs a list value containing a list of records in the page. Our implementation uses wrapper induction techniques to generate extraction programs.
- *I/O Activities*: Set of activities for reading/writing data from/to files and databases.
- *WAIT*. It causes the workflow to wait the specified number of milliseconds.
- *THROW*: This activity throws user-defined exceptions.

- *EXPRESSION*: It receives zero or more values of any data type as input and outputs a single value. The output value is computed using an expression that can use constants, functions and the input values. The implementation of this activity supports arithmetic operations, text processing and regular expressions, date manipulation, textual similarity functions and functions to manage list values.
- *RECORD_CONSTRUCTOR*: This is the basic activity for transforming and combining data records. It receives zero or more values of any data type as input and outputs a record. The workflow creator defines the fields that form the output record. For each field, she/he needs to provide an expression to compute its value, expressed in function of the input values. The expressions used should support the same operations supported by the *EXPRESSION* activity.
- *CREATE_LIST/ADD_RECORD_TO_LIST*: The *CREATE_LIST* activity creates an empty list value. The *ADD_RECORD_TO_LIST* activity receives as inputs a list value and a record value and outputs a list value containing the result of adding the input record to the input list.
- *OUTPUT*: This activity produces the workflow output. Although a workflow usually returns a list of records, the output activity allows returning each data record to the invoking application as soon as it is available. Since web navigations can be slow, the lapse between obtaining the first output record and the last can be big.
- *Custom Activities:* It is useful to allow developers to create new activities by using a standard programming-language. For instance, this allows invoking external applications. In our implementation, custom activities are created using Javascript.

The structured activities included in the language are:

- *SWITCH*: This activity implements conditional bifurcations in the workflow. It receives as inputs zero or more values of any type. Each output arrow from the activity represents a possible execution path. Each path has an associated Boolean condition (expressed in function of the input values) which triggers its activation.
- *LOOP/REPEAT*: These activities allow creating conditional loops. They specify an exit condition typically expressed in function of the input values.
- *ITERATOR*: It allows specifying a non-conditional loop by iterating on a list of records. It receives a list value as input and, in each iteration, outputs a record contained in the list. It allows configuring parallel execution of its iterations. According to the results of the experimental study, this is very useful, for instance, to access in parallel detail pages of a list of extracted data records.
- *FORM_ITERATOR*: According to the results of the experimental study, web automation tasks frequently need to execute several queries on the same web form using different combinations of query parameters. While this can be done with the basic activities, it is very useful to have a specific activity for this purpose.

Basic and structured activities must be configured to set their specific information composed by: input/s of the activity (whenever is/are necessary), output of the activity (whenever the activity has output) and additional information based on the type of the activity (i.e. the *SEQUENCE* activity needs as additional information the sequence navigation to execute and the *SWITCH* activity needs as additional information the boolean condition which triggers its activation).

3.3 User-Defined Reusable Components

The proposed model allows users to create reusable components of two kinds: *binary-reuse* components and *source-reuse* components. *Binary-reuse* components allow exporting an existing workflow as an activity (we call such activities "Workflow Activities") that can then be used to create new workflows. This way, sub-processes implementing functionality common to several wrappers can be easily reused.

Source-reuse components allow defining reusable templates to represent frequent *structural patterns*. Templates are reused at the source level because the implementation of structural patterns in each wrapper may suffer slight variations that prevent reuse at the binary level. The remaining of this section describes the use of templates.

At workflow creation time, workflow creators can drag and drop templates to the workspace and compose several templates to easily create wrappers that need to implement common structural patterns. A template is created in a similar way as a workflow, with the following differences:

- The template creator does not need to provide configuration information for every activity in the template. The configuration of these activities will be "filled in" when the template is used to create a workflow.
- As well as workflows, templates return an output value and can require mandatory and optional parameters. Nevertheless, when instantiating a template to create a workflow, the workflow creator may add as many new input parameters as wished. This is allowed because those additional parameters may be needed as inputs for the activities of the template left without configuration at template-creation time.
- Templates can include special activities called "*Interface Activities*". Interface activities specify a list of input parameters and one output result, but they do not specify any particular implementation. When the workflow creator uses the template to create a new workflow, she/he will specify an implementation for the Interface Activity. This implementation can be any activity or complete workflow having entries and outputs conforming to the ones defined by the interface activity. The workflow creator can also implement an Interface Activity by using another template. As well as with templates, at workflow creation time, the workflow creator may add as many new input parameters as needed to the Interface Activity.

Now, we introduce some example templates. Fig. 2 shows a template called *Simple_Pagination* used to process the common kind of paginated result intervals, where the next interval is accessed by clicking on a 'Next' link or button (**NOTE**: in all workflow figures, the arrows connecting the activities represent the execution flow and the dotted lines represent the data flow: that is, how values are produced and consumed by the activities. Also notice the legend on the lower right corner of Fig. 2, indicating how the values of the different data types are represented in the figures). The template receives as input a *page* value and returns a list of records. The activities the workflow creator needs to configure appear in grey in the figure. The template iterates through the result listing pages until there are no more intervals left (this is detected by a *SWITCH* activity). The *SEQUENCE* activity called *Go_to_Next_Interval* navigates to the next result interval. The *EXTRACTOR* activity obtains the list of data records in each page. The workflow creator also needs to provide an implementation for the *Process_Record* Interface Activity, which is in charge of processing each record.

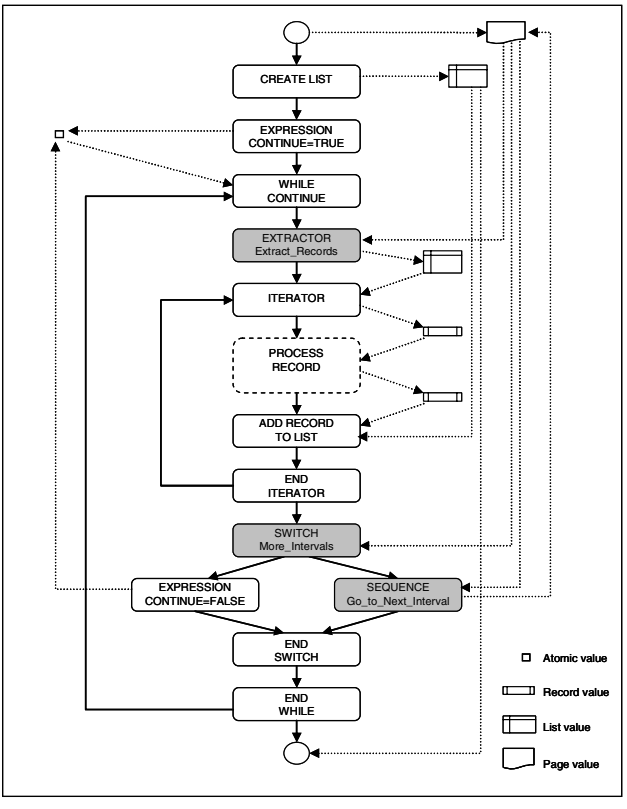


Fig. 2. *Simple_Pagination* template

We consider three example implementations of *Process_Record*:

- The first one is directly using the *OUTPUT* activity. This can be used when the workflow only needs to return the extracted records to the invoking application.
- The second example implementation is built using a template called *Filter_and_Transform* (see Fig. 3) that: 1) Filters the extracted records according to a condition specified in the configuration of the *SWITCH* activity, and 2) Transforms the records that passed the filter according to the expressions specified in a *RECORD_CONSTRUCTOR* activity. The template also uses the *Process_Record* Interface Activity to allow further processing of the records.
- The third example implementation is a template called *Detail* (see Fig. 3). This template allows accessing a 'detail' page in order to complete the data extracted for each item. It receives as input the record extracted from the result listing page. The template starts navigating to the detail page of the item (*SEQUENCE* activity). Then, it extracts the detail information (*EXTRACTOR* activity) and combines it with the input record to form a single record (*RECORD_CONSTRUCTOR* activity). The template uses a *Process_Record* Interface Activity to process the record containing the complete item. Therefore, the possible implementations for the interface activity

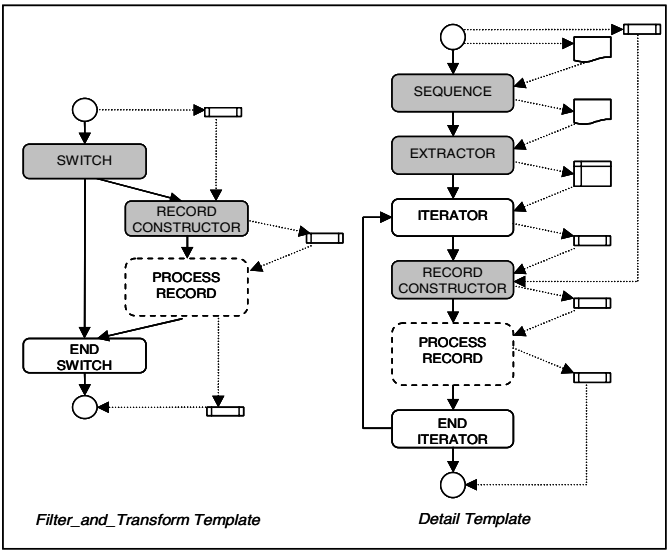


Fig. 3. *Filter_and_Transform* and *Detail* templates

include the three discussed options. For instance, if it is needed to access several levels of detail pages, the *Detail* template can be used recursively.

3.4 Example

This section illustrates some of the main features of the language through a wrapper automating the interaction with a web portal providing information about the incidences reported by the clients of an Internet Service Provider. When an incidence requires an intervention in the user's home, the ISP subcontracts an enterprise partner to perform it. The example wrapper is used by an enterprise partner of the ISP to automate the retrieval of the incidences data that a given worker could attend.

The wrapper has the following inputs: the login/password to access the portal, the zipcode indicating the current location of the worker of the enterprise partner, the maximum distance the worker could travel to solve an incidence, and the type of incidences the worker can solve. The wrapper should perform the next steps:

1. Authenticate in the ISP web portal using the login/password pair.
2. Fill in a search form to obtain all the active incidences located near the input zipcode. The incidence listing is paginated using a 'Next' link.
3. Extract all the incidences data. The data shown in the result listing includes the incidence type. If the incidence is of the type the worker can attend, then it is needed to access a detail page to obtain additional information, such as the distance with respect to the input zipcode. The returned incidence data must include a derived field indicating the deadline for attending the incidence; it is computed from two extracted items: the date when the incidence was open and the maximum number of days agreed between client and ISP.

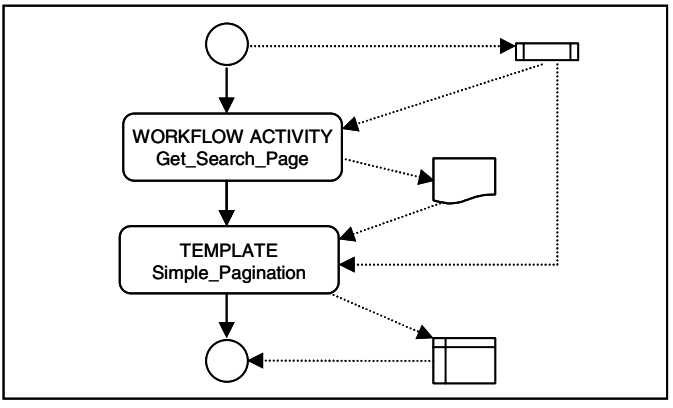


Fig. 4. High-level Activities

4. Return the incidences of the input type having distance less than the input maximum distance. In addition, the wrapper should be able of dealing with one error condition: the incidences search form can return a message error when the input zipcode is outside the area assigned to the partner.

In addition, the wrapper should be able of dealing with one error condition: the incidences search form can return a message error when the input zipcode is outside the geographical area assigned to AcmeInstall. The process flow of the wrapper executes two high-level sub-processes: one *workflow activity* (recall section 3.3) called *Get_Search_Page* and an implementation of the *Simple_Pagination* template (see Fig. 2). Fig. 4 shows the complete wrapper.

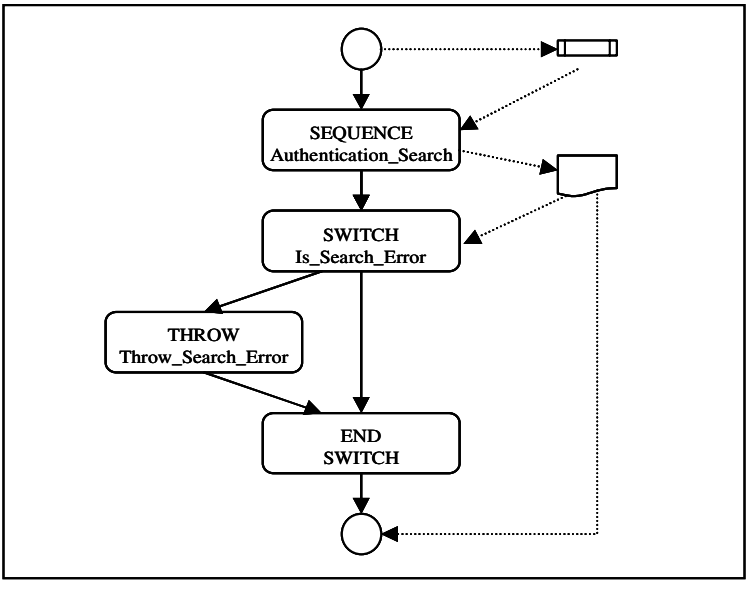


Fig. 5. *Get_Search_Page* Workflow Activity

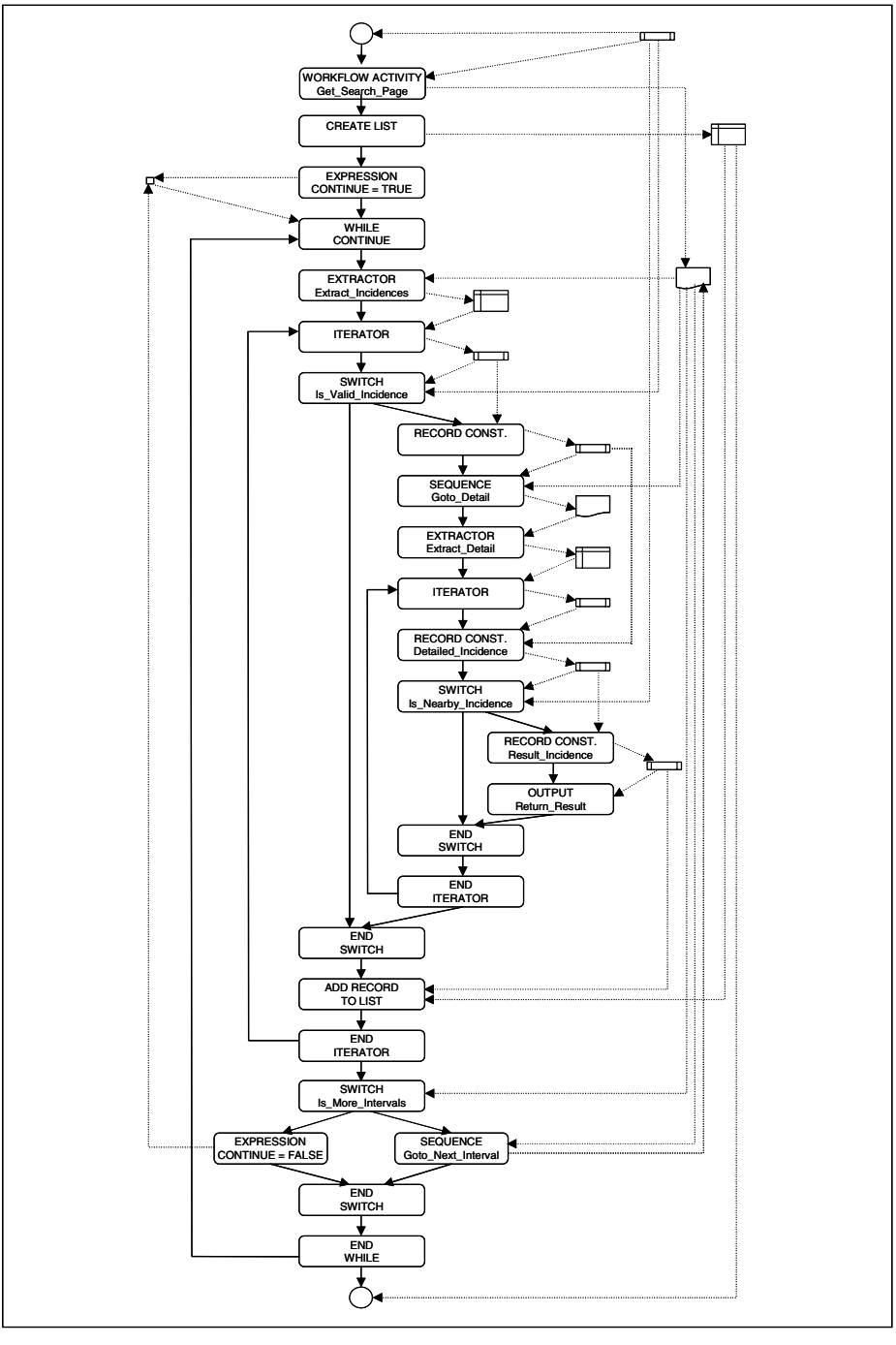


Fig. 6. Full wrapper using the *Simple_Pagination* template

As it is shown in Fig. 5, *Get_Search_Page* subprocess first performs the authentication process and the search using a *SEQUENCE* activity. The sequence accesses the page containing the authentication form, fills in the *LOGIN* and *PASSWORD* fields and submits the form. Then, it executes the incidences search by accessing the query form, filling in the *ZIPCODE*. Then, a *SWITCH* activity called is used to check if the page source code contains the text '*Incorrect zipcode'*. If the message is found, the *Throw_Search_Error* activity ends the process returning an exception. Notice that this sub-process could be reused by other wrappers using the same source and search form but processing the results differently.

Now, we describe how to fill in the *Simple_Pagination* template to extract and process the incidences in the desired way. The input page for the activity is the page outputted by the *Get_Search_Page* subprocess. It returns a list of records of INCIDENCE type, containing the data from the desired incidences.

The steps needed to fill in the template are (see Fig. 6):

1. We need to "fill in" the template by configuring the *Go_to_Next_Interval SEQUENCE* activity with the sequence for navigating to the next result interval (e.g. clicking on an anchor) and the *Extract_Records EXTRACTOR* activity with the needed extraction rules to obtain the incidences list from the search result.
2. The *Process_Record* Interface Activity of *Simple_Pagination* can be implemented using an instance of the *Filter_and_Transform* template to filter the incidences of the input type using the *SWITCH* activity. It is not needed to configure the *RECORD_CONSTRUCTOR* activity since its default settings are valid.
3. The *Process_Record* Interface Activity from the *Filter_and_Transform* template used in step 2 can be implemented using the *Detail* template (the incidence detail page is accessed only for the incidences of the input type). We need to provide the sequence for navigating to the detail page and the extraction rules to obtain the detail data. We also need to configure the *SWITCH* activity to filter the incidences according to the input *DISTANCE*, and the *RECORD_CONSTRUCTOR* activity to add the additional field *DEADLINE_DATE*.
4. The *Process_Record* Interface Activity from the previous *Detail* template can be implemented using again *Filter_And_Transform* to filter all the incidences of the input type verifying that its distance is less than the input maximum distance. Finally, the *Process_Record* Interface Activity from the *Filter_And_Transform* template is implemented by simply using the *OUTPUT* activity.

4 Related Work

Most previous works on wrapper generation have focused on the building blocks for web automation: web data extraction and automatic web navigation. The web data extraction problem has been addressed for instance in [4],[5],[9],[10],[11],[12],[13], [14],[17],[18]. [8] provides a survey. Techniques for automatic generation of web navigation sequences were proposed in [3] and [14]. These works do not consider the problem of how to specify the logic of the complete wrapper. Nevertheless, they provide the foundations for the *SEQUENCE* and *EXTRACTOR* activities of the proposed model. There are two kinds of works addressing the problem of building complete wrappers:

- Some works such as [11] define specialised languages for programming wrappers. Our proposal has several advantages: 1) it allows graphically specifying the wrapper logic: this way, wrappers are simpler to create and maintain and programming skills are not required; 2) it encapsulates the data extraction and automatic navigation tasks, leveraging on semi-automatic methods.
- Other works such as [5],[10],[13],[15] propose techniques to create complete wrappers without needing programming skills. Nevertheless, these systems implicitly assume the query wrapper model. As it has been previously discussed, this model is not suitable for a substantial number of web automation applications.

In the industrial arena, many web automation tools have appeared. QL2 (http://www.ql2.com) and NewBie (http://www.newbielabs.com) follow the approach of providing specialised programming languages. Our proposal has the same advantages over these tools already mentioned for the research systems using the same approach. Another interesting tool is Dapper (http://www.dapper.net) which allows creating and sharing wrappers between final users. The wrappers that can be created using Dapper are roughly equivalent to those supported by the query wrapper model. As it has been already discussed, this is not enough for enterprise-class web automation. The Kapow Robomaker tool (http://www.openkapow.com) also uses a workflow approach for web automation. The approach proposed in this paper has a number of advantages with respect to Robomaker: 1) Robomaker does not encapsulate complex data extraction tasks in activities. The extraction of a list of data records requires an activity in the workflow to extract each record field. Optional attributes in the records require bifurcations in the workflow. This leads to large workflows even for relatively simple tasks. In addition, their model does not support using semi-automatic methods for extraction, 2) Robomaker does not support defining reusable components, 3) Robomaker does not support other functionalities such as user-defined exceptions.

5 Conclusions

This paper describes a new graphical language for designing web automation applications. Our approach models each task as a high-level workflow composed of specialized activities which leverage on the previous research work in automatic web navigation and web data extraction techniques, but allow for complex logic-control features such as branching or parallelism.

The model also allows the easy creation of reusable components. On one hand, subprocesses common to several wrappers can be easily created and reused through workflow activities. On the other hand, it is allowed to easily create templates that implement structural patterns at the source-level that repeat across many applications. Reusable components also increase simplicity of development, since advanced users can create complex patterns reused by less sophisticated users.

The proposed language has been designed from the study of real-world web automation tasks, to ensure it captures all the needed requirements.

References

1. Aalst, W., Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow patterns. Distributed and Parallel Databases 14(1), 5–51 (2003)
2. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley, Reading (1995)
3. Anupam, V., Freire, J., Kumar, B., Lieuwen, D.F.: Automating Web navigation with the WebVCR. Computer Networks 33(1-6), 503–517 (2000)
4. Arasu, A., Garcia-Molina, H.: Extracting Structured Data from Web Pages. In: Proceedings of the 2003 ACM SIGMOD International Conference, pp. 337–348 (2003)
5. Baumgartner, R., Flesca, S., Gottlob, G.: Declarative Information Extraction, Web Crawling and Recursive Wrapping with Lixto. In: Proceedings of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNR) (2001)
6. Oasis WS-BPEL. Web Services Business Process Execution Language, `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel`
7. BPMN: Business Process Modelling Notation, `http://www.bpmn.org`
8. Chang, C.-H., Kayed, M., Girgis, M.R., Shaalan, K.F.: A Survey of Web Information Extraction Systems. IEEE Transactions on Knowledge and Data Engineering 18(10), 1411–1428 (2006)
9. Crescenzi, V., Mecca, G.: Automatic Information Extraction from Large Websites. J. ACM 51(5), 731–779 (2004)
10. Doorenbos, R., Etzioni, O., Weld, D.S.: A Scalable Comparison-Shopping Agent for the World-Wide Web. Agents, 39–48 (1997)
11. Kistlera, T., Marais, H.: WebL: A Programming Language for the Web. In: Proceedings of the 7th International World Wide Web Conference, pp. 259–270 (1998)
12. Wang, J., Lochovsky, F.: Data Extraction and Label Assignment for Web Databases. In: Proceedings of the 12th World Wide Web Conference, pp. 187–196 (2003)
13. Knoblock, C.A., Lerman, K., Minton, S., Muslea, I.: Accurately and Reliably Extracting Data from the Web: A Machine Learning Approach. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering (1999)
14. Pan, A., Raposo, J., Álvarez, M., Hidalgo, J., Viña, A.: Semi Automatic Wrapper-Generation for Commercial Web Sources. In: Proceedings of IFIP WG8.1 EISIC (2002)
15. Sahuguet, A., Azavant, F.: WysiWyg Web Wrapper Factory (W4F). In: Proceedings of the 8th International World Wide Web Conference (1999)
16. Wiederhold, G.: Mediators in the architecture of future information systems. Computer 25(3) (March 1992)
17. Zhai, Y., Liu, B.: Structured Data Extraction from the Web Based on Partial Tree Alignment. IEEE Trans. Knowl. Data Eng. 18(12), 1614–1628 (2006)
18. Zhai, Y., Liu, B.: Extracting Web Data Using Instance-Based Learning. In: Proceedings of the 16th International World Wide Web Conference (2007)

Contained Rewritings of XPath Queries Using Views Revisited

Junhu Wang[1], Jeffrey Xu Yu[2], and Chengfei Liu[3]

[1] Griffith University, Gold Coast, Australia
J.Wang@griffith.edu.au
[2] Chinese University of Hong Kong, Hong Kong, China
yu@se.cuhk.edu.hk
[3] Swinburne University of Technology, Australia
cliu@ict.swin.edu.au

Abstract. We revisit the problem of finding maximal contained rewritings of tree pattern queries using views, and extend previous work to the case where both the query and the view involve *, with the restriction that no *-node is incident on any //-edge, and there are no leaf *-nodes. When there is no DTD, we show how to modify the approach of useful embedding in a previous work to find the maximal contained rewriting. When there is a DTD that is represented as an acyclic schema graph G, we show how to efficiently transform the view V to an equivalent DAG-pattern V', and reduce the problem of finding the maximal contained rewriting of query Q using V under G to that of finding the maximal contained rewriting of Q using V' without DTDs.

1 Introduction

Query rewriting using views has many applications including data integration and query optimization [3]. A view is an existing query whose answer may or may not have been materialized. Given a new query, the problem is to find another query using only the views that will produce correct answers to the original query. Usually two types of rewritings are sought: *equivalent rewritings* and *contained rewritings*. An equivalent rewriting produces all answers to the original query, while a contained rewriting may produce only part of the answers. Both types of rewritings have been extensively studied in the relational database context [3].

More recently rewriting XML queries using XML views has attracted attention because of the rising importance of XML data [8,6,5,2]. Since XPATH lies in the center of all XML languages, the problem of rewriting XPATH queries using XPATH views is particularly important. Some major classes of XPATH expressions can be represented as tree patterns [1,7]. Among previous work on rewriting XPATH queries using views, [8] studied equivalent rewritings of tree patterns discussed in [7], [6] presented results on equivalent rewritings of tree patterns when the tree patterns are assumed to be minimized, [2] studied a different type of equivalent rewritings of tree patterns under structural summaries, and [5] studied maximal

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 410–425, 2008.

contained rewritings of tree patterns where both the view and the query involve /,// and [] only (these XPATH expressions correspond to tree patterns in $P^{\{//,[]\}}$ [7]), both in the absence and in the presence of non-recursive schema graphs - a restricted form of DTDs.

In this paper we extend the work of [5] by considering queries and views involving *, with the following restrictions: no nodes labeled * are incident on //-edges, and no leaf nodes are labeled *. When there is no DTD, we show how to modify the useful embedding of [5] to find the maximal contained rewritings. When there is a DTD that is represented as an acyclic schema graph G, we show how to efficiently transform the view V into an equivalent DAG-pattern V', and reduce the problem of finding the maximal contained rewriting of Q using V under G to that of finding the maximal contained rewriting of Q using V' without G.

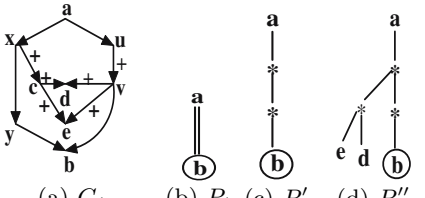
The rest of the paper is organized as follows. In Section 2 we provide the background and notations. Section 3 deals with contained rewritings when there are no DTDs, and Section 4 presents the view transformation algorithms under an acyclic DTD, as well as the algorithms for finding the maximal contained rewriting when such a DTD exists. Finally Section 5 concludes the paper.

2 Preliminaries

2.1 DTD, XML Tree and Tree Patterns

Let Σ be an infinite set of tags. We model a non-disjunctive DTD as a connected directed graph G such that (1) each node is labeled with a distinct tag, (2) each edge is labeled with one of 1, ?, +, and $*$, which indicate "exactly one", "one or zero", "one or many", and "zero or many", respectively. Here, *the default edge label is* $*$, and (3) there is a unique node, called the root, which has an incoming degree of zero. The set of tags occurring in G is denoted Σ_G. Because a node in a DTD G has a unique label, we also refer to a node by its label. In this paper we will implicitly assume all DTDs are acyclic. A DTD example is shown in Fig.1 (a).

An XML *tree* is an unordered tree[1] with every node labeled with a tag in Σ. A *tree pattern (*TP*)* is a tree with a unique *distinguished node*, and with every



(a) G_1 (b) P_1 (c) P_1' (d) P_1''

Fig. 1. DTD G_1 and TPs P_1, P_1', P_1''

[1] In this work order of nodes is disregarded.

node labeled with a symbol in $\Sigma \cup \{*\}$ (here * is the wildcard which represents any tag), every edge labeled with either / or //. The path from the root to the distinguished node is called the *distinguished path*. Fig.1 (b), (c), (d) show three TPs P_1 and P_1' and P_1'', where single and double lines are used to represent /-edges and //-edges respectively, and a circle is used to indicate the distinguished node. A TP corresponds to an XPATH expression. The TPs in Fig.1 correspond to the expressions $a//b$, $a/*/*/b$ and $a/*[*[e]/d]/*/b$ respectively. Let P be a TP. We will use $\mathtt{DN_P}$, and $\mathtt{DP_P}$ to denote the distinguished node and the distinguished path of P respectively. Note: the TPs in our discussion correspond to the fragment $P^{\{//,[],*\}}$ defined in [7]. In this paper we are interested in the subset of all TPs in $P^{\{//,[],*\}}$ such that no *-node is incident on a //-edge, and no leaf node is labeled *, and we denote this subset by $\widehat{P}^{\{//,[],*\}}$.

Below, for any tree or DTD T, we will use $N(T)$, $E(T)$ and $rt(T)$ to denote the node set, the edge set, and the root of T respectively. We will also use $label(v)$ to denote the label of node v, and call a node labeled a an *a-node*. An XML tree t is said to *conform* to DTD G if (1) for every node $v \in N(t)$, $label(v) \in \Sigma_G$, (2) $label(rt(t)) = label(rt(G))$, (3) for every edge (u, v) in t, there is a corresponding edge $(label(u), label(v))$ in G, and (4) for every node $v \in N(t)$, the number of children of v labeled with τ is constrained by the label of the edge $(label(v), \tau)$ given in G. We denote the set of all XML trees conforming to G by T_G.

A *matching* of a TP, P, in an XML tree, t, is a mapping δ from $N(P)$ to $N(t)$ satisfying the following conditions: (1) root-preserving, i.e., $\delta(rt(P)) = rt(t)$, (2) label-preserving, i.e., $\forall v \in N(P)$, either $label(v) = *$ or $label(v) = label(\delta(v))$, and (3) structure-preserving, i.e., for every edge (x, y) in P, if it is a /-edge, then $\delta(y)$ is a child of $\delta(x)$; if it is a //-edge, then $\delta(y)$ is a descendant of $\delta(x)$, i.e, there is a path from $\delta(x)$ to $\delta(y)$. Each matching δ produces a subtree of t rooted at $\delta(\mathtt{DN_P})$, denoted $\mathsf{sub}^t_{\delta(\mathtt{DN_P})}$, which is also known as an *answer* to the TP. We use $P(t)$ to denote the set of all answers of P over t. A tree pattern P is said to be *satisfiable* under DTD G if there exists $t \in T_G$ such that $P(t) \neq \emptyset$.

2.2 Containment and Containment Mapping

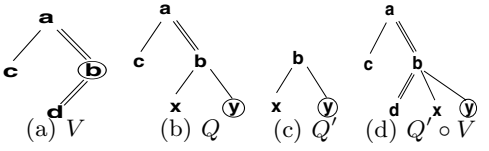
Let P and Q be TPs. P is said to be *contained* in Q, denoted $P \subseteq Q$, if for every XML tree t, $P(t) \subseteq Q(t)$. Let G be a DTD. P is said to be *contained* in Q under G, denoted $P \subseteq_G Q$, if for every XML tree $t \in T_G$, $P(t) \subseteq Q(t)$. It is shown in [7] that when Q has no *-nodes, or Q has no //-edges and no leaf *-nodes, $P \subseteq Q$ iff there is a containment mapping from Q to P. Recall: A *containment mapping (*CM*)* from Q to P is a mapping δ from $N(Q)$ to $N(P)$ that is label-preserving, root-preserving as discussed in the last section, structure-preserving (which now means for any /-edge (x,y) in Q, $(\delta(x), \delta(y))$ is a /-edge in P, and for any //-edge (x, y) in Q, there is a path from $\delta(x)$ to $\delta(y)$ in P) and is output-preserving, which means $\delta(\mathtt{DN_Q}) = \mathtt{DN_P}$. A closer inspection of the proof in [7] shows an extension of the above result as follows.

Proposition 1. *For any $P \in P^{\{//,[],*\}}$, $Q \in \widehat{P}^{\{//,[],*\}}$, $P \subseteq Q$ iff there is a* CM *from Q to P.*

2.3 Rewriting TP Using Views

A *view* is a pre-defined TP. Let V be a view and Q be a TP. A *contained rewriting* (CR) of Q using V is a TP Q' such that when Q' is evaluated on the subtrees returned by V, it gives a set of correct answers to Q. More precisely, the following two conditions hold: (1) for any XML tree t, $Q'(V(t)) \subseteq Q(t)$, and (2) there exists an XML tree t such that $Q'(V(t))$ is non-empty. The *maximal contained rewriting* (MCR) of Q using V, denoted $\mathsf{MCR}(Q, V)$, is the union of all CRs of Q using V.

Let Q' be a CR of Q using V. We use $Q' \circ V$ to represent the *expansion* of Q', which is the TP obtained by merging the root of Q' and the distinguished node of V (the merged node is labeled with $label(rt(Q'))$ if $label(rt(Q')) \neq *$, otherwise it is labeled with $label(\mathtt{DN_V})$). Fig.2 shows a view V, a TP Q, a CR Q' of Q using V, and the expansion $Q' \circ V$ of Q'. Note that condition (1) in the definition of CR is equivalent to $Q' \circ V \subseteq Q$.



(a) V (b) Q (c) Q' (d) $Q' \circ V$

Fig. 2. View (a), Query (b), CR (c), and Expansion (d)

Given TP Q and view V in $P^{\{//,[]\}}$, [5] shows that the existence of a CR of Q using V can be characterized by the existence of a *useful embedding* of Q in V. In brief, an embedding of Q in V is a partial mapping f from $N(Q)$ to $N(V)$ that is root-preserving, label-preserving, structure-preserving and upward-closed. Here, root-preserving, label-preserving, and structure-preserving are the same as in a containment mapping, except that they are required only for the nodes of Q on which the function f is defined; and upward-closed means that if $f(x)$ is defined, then f is defined on every node from $rt(Q)$ to node x. An embedding f of Q in V is said to be *useful* if (1) for every node x in the distinguished path $\mathtt{DP_Q}$, if $f(x)$ is defined, then $f(x) \in \mathtt{DP_V}$, and if $f(\mathtt{DN_Q})$ is defined, then $f(\mathtt{DN_Q}) = \mathtt{DN_V}$; (2) for every path $p \in Q$, *either* p is fully embedded, i.e., f is defined on every node in p, *or* if x is the last node in p such that $f(x)$ is defined and $f(x) \in \mathtt{DP_V}$ (x is called the *anchor* of p) and y the node immediately following x (called the *successor* of x), then either $f(x) = \mathtt{DN_V}$, or the edge (x, y) is a //-edge.

Note that the condition "if $f(\mathtt{DN_Q})$ is defined, then $f(\mathtt{DN_Q}) = \mathtt{DN_V}$" in the above definition of useful embedding is important although it appears to be missing from [5]. Consider the two TPs in Fig.3. If we treat Q_1 as V and Q_2 as Q, then the mapping $\{v_1 \rightarrow u_1,\ v_2 \rightarrow u_2\ \}$ is not a useful embedding according to the definition above, because the distinguished node of Q is not mapped to the distinguished node of V. Without the above condition it will be a useful embedding, but obviously there is no CR of Q using V. On the other hand, if we treat Q_1 as Q and Q_2 as V, then the mapping $\{u_1 \rightarrow v_1,\ u_2 \rightarrow v_2\ \}$ will be a useful embedding, although the mapping $\{u_1 \rightarrow v_1,\ u_2 \rightarrow v_2,\ u_3 \rightarrow v_3\ \}$ is not.

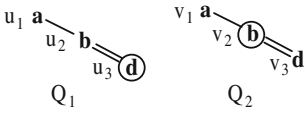


Fig. 3. TPs to illustrate useful embedding

Given a useful embedding f from Q to V, a CR Q_f can be constructed as follows. The root of Q_f is labeled $label(\mathtt{DN_V})$. For each path $p \in Q$ that is not fully embedded in V, find the anchor node x and its successor y, then connect the subpattern of Q rooted at y to $rt(Q_f)$ with the same type of edge as the edge (x, y).

A contained rewriting of Q using view V under DTD G is a TP Q' such that (1) for any XML tree $t \in T_G$, $Q'(V(t)) \subseteq Q(t)$, (2) there exists an XML tree $t \in T_G$ such that $Q'(V(t))$ is non-empty. A MCR of Q using V under DTD G is the union of all CRs of Q using V under G. Given an acyclic DTD G, a view V and a TP Q, [5] uses five types of constraints (IC, PC, SC, CC, and FC constraints, hereafter referred to as the LWZ *constraints*), that are implied by G, to chase V into another TP V', such that $V' =_G V$ ($V' \subseteq_G V$ and $V \subseteq_G V'$), and V' can not be chased further. To find a CR of Q using V under G, it finds a useful embedding from Q to V'. The useful embedding satisfies all conditions of a useful embedding when DTD G is absent with an additional condition to ensure satisfiability of the expansion of the rewriting under G.

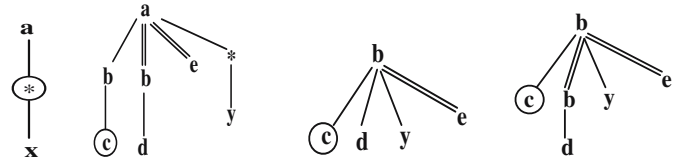
In what follows, we will implicitly assume the views and queries are all in $\widehat{P}^{\{//,[],*\}}$. We focus on efficient algorithms to find MCRs of queries using views for the class $\widehat{P}^{\{//,[],*\}}$ of TPs.

3 Rewriting without DTDs

When there are no DTDs available, the LWZ approach of [5] can be easily modified to find CRs of Q using V, for $Q, V \in \widehat{P}^{\{//,[],*\}}$. The modification we need is to relax the label-preserving requirement of a useful embedding to *weak label-preserving* defined below, to handle the case where $label(\mathtt{DN_V}) = *$. Correspondingly, if $label(\mathtt{DN_V}) = *$, and there is a useful embedding of Q in V with some non-* node u in Q mapped to $\mathtt{DN_V}$, then the root of the rewriting generated by this useful embedding should be labeled with $label(u)$.

Definition 1. *A mapping h from $N(Q)$ to $N(V)$ is* weak label-preserving *if (1) for every $v \in N(Q)$, either $label(v) = *$, or $label(v) = label(h(v))$, or $h(v) = \mathtt{DN_V}$ and $label(\mathtt{DN_V}) = *$, (2) all non-* nodes that are mapped to $\mathtt{DN_V}$ have the identical label.*

Example 1. Consider the view V in Fig.4 (a) and query Q in Fig.4 (b). There are two useful embeddings of Q in V: The first maps both b-nodes in Q (as well as the *-node) to the $*$-node in V, and the second maps only the left b-node (and the *-node) in Q to the $*$-node in V. The corresponding CRs of Q using V



(a) V (b) Q (c) Q'_1 (d) Q'_2

Fig. 4. View V, Query Q, and CRs Q'_1 and Q'_2

constructed are shown as Q'_1 and Q'_2 in Fig.4 (c) and (d), respectively. One can verify that $Q'_1 \circ V \subseteq Q$ and $Q'_2 \circ V \subseteq Q$. The MCR is the union of the two.

Note that in the above example, there are no useful embeddings from Q to V according to the original definition.

The correctness of the above modification is straightforward from Proposition 1 and the definition of CR: a useful embedding is part of a CM from Q to $Q' \circ V$ for some CR Q' of Q using V. Finding all such useful embeddings can be done in time $O(|N(Q)|^2 \times |N(V)|^2)$, because we can do so by "instantiating" V into a set $\mathcal{V}$ of views by relabeling $\mathtt{DN_V}$ with each label which appears in Q, and then finding all useful embeddings of Q in each view in $\mathcal{V}$ using the $O(|N(Q)| \times |N(V)|^2)$ algorithm of [5].

4 Rewriting under DTDs

When a DTD is available, like [5], we first transform V into another pattern, V', such that $V' =_G V$, and for every TP $Q \in \widehat{P}^{\{//,[],*\}}$, $V \subseteq_G Q$ iff $V' \subseteq Q$; then we find the MCR of Q using V' as if there was no DTD. The main trick is how to transform V to V'. It turns out that this issue is quite intricate.

Our method of view transformation is divided into the following steps:

1. relabeling *-nodes. In this step we relabel the *-nodes in V according to G, a *-node will be relabeled with c iff the resulting pattern is equivalent to V under G.
2. merging *-nodes. In this step, we try to merge *-nodes that have a common *-parent.
3. extending //-edges. In this step, we first try to chase the view with the IC and PC constraints of [5], i.e., try to insert non-* nodes between the nodes connected by the //-edge, and change //-edges to /-edges; then we check whether a //-edge can be replaced with a *-path, here a ***-path** means a /-path (a /-path is a path consisting of only /-edges) that starts and ends with non-* nodes, but the intermediate nodes are all labeled *.
4. adding descendants to nodes having //-children. In this step, we check nodes which have children connected to them by //-edges, and try to add descendants to these nodes.

5. adding descendants to nodes on *-paths. In this step, we try to add descendant nodes to nodes (except the last node) on *-paths that existed before Step 4.
6. expanding *-paths. In this step, we try to add an additional path, $aspath(p)$, from the start node to the end node of a *-path p, possibly with some descendants of the nodes on the new path as well. This may result a pattern that is a directed acyclic graph (DAG), which we call a DAG-pattern. If some *-paths $p_1, \ldots, p_n$ share common *-nodes, we try to merge the nodes with identical labels in $aspath(p_1), \ldots, aspath(p_n)$.
7. chasing using LWZ constraints. In this step we will use the CC, SC, PC, and FC constraints of [5] to chase the view until no more change can be obtained.

Next we explain each step in more detail. We assume the root of the view is not labeled * (In the case $label(rt(V)) = *$, relabel $rt(V)$ with $label(rt(G))$).

4.1 Relabeling $*$-nodes

To relabel *-nodes, we make use of a procedure `FindLabel`(V,G), modified from a similar procedure in [4], which was originally used to test the satisfiability of tree pattern V under DTD G. The procedure finds a set $L(v)$ of possible labels for each node v in V. For non-* nodes, $L(v) = \{label(v)\}$, and for *-nodes, $L(v)$ contains the label c iff every matching δ of V in any XML tree $t \in G$ maps v to a c-node, and there is at least one such t and one such δ. Due to space limit, we omit the details, and assume such a procedure exists.

***-node relabeling:** For each *-node v in V, relabel v with c if $L(v) = \{c\}$.

Similar to [4], we can show that `FindLabel`(V,G), and hence the entire relabeling process, runs in $O(|N(G)|^3 + |N(G)|^2 \times |N(V)|)$.

4.2 Merging *-nodes

We explain the intuition using the following example first.

Consider the view V_2 shown in Fig. 5(a) under the DTD G_2 shown in Fig. 5(b). In the figure the numbers beside the *-nodes are used as node identifiers. Using `FindLabel`, we can find $L(1) = \{x, y, g\}$, $L(2) = \{g, c\}$, $L(3) = \{c, z\}$, $L(4) = L(6) = \{g, v\}$, $L(5) = L(7) = \{v, u\}$. When the *-node 1 is replaced with a g-node, the node 4 and node 6 must be replaced with a v-node, because there is no edge (g, g) in G_2. Similarly, when node 1 is replaced with an x-node (or y-node), both node 4 and node 6 must be replaced with a g-node. In G_2 the edges (g, v), (x, g) and (y, g) are labeled ?. Therefore, in every possible instance[2] of V, the nodes corresponding to 4 and 6 can be merged. Therefore, we can merge node 4 and 6 to obtain the new pattern V_2' shown in Fig. 5(c).

Generally, if a *-node v has two $*$-children v_1 and v_2, then v_1 and v_2 can be merged provided they satisfy the following condition: For every $x \in L(v)$, there is a unique label y in $L(v_1)$ and $L(v_2)$ such that (x, y) is an edge in G, and the

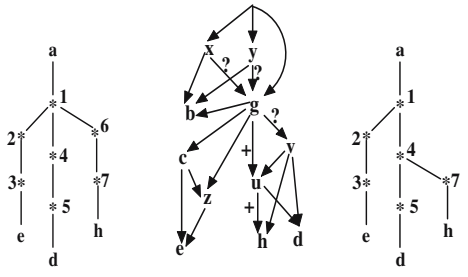
[2] A possible instance of V is an XML tree in which V has a matching.

Algorithm 1. Merging *-nodes

1: **repeat**
2: **for** each *-node v that has two children v_1 and v_2 such that $L(v_1) = L(v_2)$ **do**
3: **if** $\forall x \in L(v)$, there is a unique label y in $L(v_1)$ and in $L(v_2)$ such that (x, y) is an edge in G, and this edge is labeled ? or 1 **then**
4: merge v_1 and v_2
5: **until** no more *-nodes can be merged

FC constraint $x \twoheadrightarrow y$ is implied by G. Based on this observation, we present our algorithm for merging *-nodes in Algorithm 1.

Note it is not possible to merge two *-children of a non-*-node, because otherwise the two *-children would have been relabeled in Step 4.1.

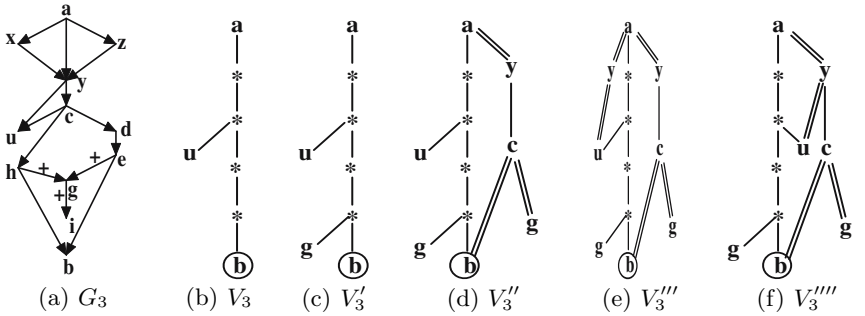


(a) V_2 (b) G_2 (c) V_2'

Fig. 5. Merging *-children of a *-node

4.3 Extending //-edges

In this step, we first use the LWZ IC constraints to chase V as in [5]. We use an example to explain the process. Consider the DTD G_3 in Fig.6. The DTD implies the IC constraint $a \xrightarrow{y} b$, and $y \xrightarrow{c} b$, that is, every path from a to b passes through y, and every path from y to b passes through c. Therefore, the



(a) G_3 (b) V_3 (c) V_3' (d) V_3'' (e) V_3''' (f) V_3''''

Fig. 6. DTD G_3, view V_3 and the transformed views

TP $a//b$ can be chased to $a//y//b$, and then to $a//y//c//b$. Then we try to change each //-edge to a /-edge if there is a relevant PC constraint. After that, we try to extend the remaining //-edges using a new type of constraints and a corresponding chase rule, given below.

Fixed length constraints (FLC): $x//y \rightarrow x/^k y$ ($k \geq 1$). The constraint means that in G, every path from x to y is of length exactly k (i.e., it has k edges).

FLC-rule: if $G \models x//y \rightarrow x/^k y$, then in V, replace every //-edge from an x-node x_0 to an y-node y_0 with a *-path from x_0 to y_0 with $k-1$ intermediate *-nodes.

For example, the DTD G_1 in Fig.1 implies the FLC $a//b \rightarrow a/^3 b$, thus the tree pattern $a//b$ is transformed to $a/*/*/b$.

Finding all LWZ constraints and FLC constraints implied by G can be done in $O(|N(G)|^3)$, therefore this step takes $O(|N(G)|^3 \times |E(V)| \times H)$, where H is the length of the longest path in G.

4.4 Adding Descendants to Nodes Having //-children

In this step, we try to add descendants to nodes that have //-children. For each //-edge e, we call the procedure `HandleEdge` as shown in Algorithm 2 to add descendants to x_1 (suppose e=$x_1//y_1$). There are two types of descendants we may add: fixed length descendants and variable length descendants. The former will be connected to x_1 via a *-path, and the later by a //-edge. Before explaining the algorithm, we need to define some terms and notations.

Let p be a path in G or V. We will use $|p|$ to denote the *length* of p, namely the number of edges in p. We also use $p[i]$ to denote the ith node on p, for $i = 1, \ldots, |p|+1$. In addition, an $x//y$-edge refers to a //-edge from an x-node to a y-node.

Definition 2. *Let $x_1//y_1$ be an $x//y$-edge. We call any path from x to y, in G, an* **image of** *$x_1//y_1$ in G. Let $p = u_0/u_1/\cdots/u_k$ be a *-path in V, and $q = l_0.l_1.\cdots.l_k$ be a path in G such that $l_0 = label(u_0)$ and $l_k = label(u_k)$. We say q is an* **image** *of p in G, and u_i* corresponds *to l_i, if $l_i \in L(v_i)$ for all $i \in [1, k-1]$.*

For any *-path or //-edge e, we use $image(e, G)$ (or $image(e)$ for short, when G is known) to denote the set of all images of e in G.

Definition 3. *A* mandatory path *in G is a path of length 0, or where every edge is labeled + or 1. Let x, y be nodes in G. If there is a mandatory path from x to y of length K, we say y is a* distance K mandatory descendant *of x. A* mandatory child *means a distance* 1 *mandatory descendant.*

Note that a distance 0 mandatory descendant of x is x itself.

Algorithm 2. `HandleEdge`(e, S), where e is an $x//y$-edge, S is a set of paths from x to y in G

1: let $l = min\{|q| \mid q \in S\}$
2: add a distance $l-1$ *-descendant s to x_1, denote the path from x_1 to s by p
3: **for** $i = l-1$ to 2 **do**
4: let $N_i = \bigcap_{q \in S} N_{q[i]}$
5: **if** there is integer $d \geq 0$ and node τ in N_i, such that $\forall q \in S$, there is a derived path from $q[i]$ to τ of length d **then**
6: add a distance d τ-descendant to $p[i]$ (provided no distance $d-1$ τ-descendant of $p[i+1]$ has been added before)
7: suppose $\tau_1, \ldots, \tau_k$ are all descendants of $p[i]$ added above
8: **for** $s, t \in [1, k]$ **do**
9: let $u(q)$ be the last common node on the derived paths $q[i] \rightsquigarrow \tau_s$ and $q[i] \rightsquigarrow \tau_t$ /* if there are multiple such derived paths consider each one in turn*/
10: let $d' = min\{|q[i] \rightsquigarrow u(q)| \mid q \in S\}$
11: **if** $d' > 0$ **then**
12: merge the d'th *-nodes (below $p[i]$) on the *-path from $p[i]$ to τ_s and that from $p[i]$ to τ_t
13: remove all leaf *-nodes (and their incident edges) from V
14: **for** each label $\tau \in N_2$ **do**
15: **if** $\exists \tau' \in N_2$ such that there is a mandatory path from τ' to τ in G **then**
16: let $N_2 = N_2 - \{\tau\}$;
17: **for** each $\tau \in N_2$ **do**
18: add a τ-child v_τ under x_1 and label the edge (x_1, v_τ) with //, if x_1 does not have a τ-descendant already;

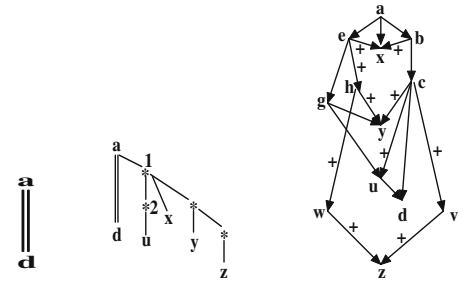
Definition 4. *Given a node $x \in G$, the* mandatory descendant set *of x, denoted M_x, is the set of all nodes in G reachable by some mandatory path from x, including x itself. Let q be an image of a *-path or a //-edge. For $i = 1, \ldots, |q|$, define*

$$N_{q[i]} = M_{q[i]} \cup M_{q[i+1]} \cup \cdots \cup M_{q[|q|]}.$$

We call every node in $N_{q[i]}$ a **derived descendant** of $q[i]$ (wrt q). By definition, for every node $\tau \in N_{q[i]}$, there is a path from $q[i]$ to τ that consists of two segments: the first segment is the path from $q[i]$ to $q[j]$ (for some $j \in [i, |q|]$) on q; the second segment is a mandatory path from $q[j]$ to τ. Such a path will be called a **derived path** from $q[i]$ to τ, denoted $q[i] \rightsquigarrow \tau$.

Let v' be a descendant of v in TP V. If the path from v to v' is of length K, we say v' is a *distance K descendant* of v (A distance 0 descendant of v is v itself). To add a distance K descendant v' to v, we add a path from v to v' of length K where all nodes between v and v' are *-nodes.

We now explain the procedure in Algorithm 2 using the view V_4 and DTD G_4 in Fig. 7 (c). For every //-edge $x_1//y_1$ in V, we add a temporary *-descendant s to x_1 (note s is a leaf *-node now). The distance from x_1 to s is the same as the length of shortest image minus 1. We denote the path from x_1 to s by p (line 1-2). In our example, s is node 2. Then we check each node $p[i]$ on p from bottom-up to see whether any fixed length descendants can be added to



(a) V_4 (b) V_4' (c) G_4

Fig. 7. Fixed length descendants are added to the a-node to obtain V_4' from V_4

it. For each image q, we check whether there is a derived path from $q[i]$ to τ of length d, for a fixed integer $d \geq 0$ and τ in G (the set N_i of possible τs are calculated first). If the condition is true, we will add a distance d τ-descendant to $p[i]$ provided no distance $d-1$ τ-descendant of $p[i+1]$ has been added earlier (line 8-9). In our example, we can add a u-child under node 2. Similarly we can add an x-child, a distance 2 y-descendant, and distance 3 z-descendant to node 1. After adding descendants $\tau_1, \ldots, \tau_k$ to $p[i]$, we need to check whether some *-nodes on the path from $p[i]$ to τ_s and that from $p[i]$ to τ_t can be merged (line 10-15). In the example, we can merge the first *-nodes under node 1 on the derived paths leading to y and z (Fig.2 (b)). If no descendant is added to the last *-node on the temporary path q, there will be some dangling *-nodes, and we must remove them (line 13). Finally, we add a variable length τ-descendant to x_1 for each $\tau \in N_2$ (line 17-18). In the example, no such descendants can be added.

4.5 Adding Descendants to Nodes on *-paths

In this step, we try to add descendants to nodes (except the last node) on each *-path. The descendants added to the start node of the path can be fixed-length descendants or other descendants, while those added to the *-nodes are always fixed-length descendants. Basically, for every *-path p that existed before Step 4.4, we consider the *-nodes on p one by one from bottom up (see the procedure `HandleStarPath` in Algorithm 3). If for a fixed integer $d > 0$ and node τ in G, there is a derived path from $q[i]$ to τ of length d, for every image $q \in image(p)$, we can add a distance d τ-descendant to $p[i]$. If there are multiple descendants $\tau_1, \ldots, \tau_k$ added to $p[i]$ this way, we need to check whether the corresponding nodes on the derived paths from $q[i]$ to $\tau_1, \ldots, \tau_k$ can be merged. Finally we add other descendants to the start node of p. For example, in G_1 of Fig.1 the paths from a to b of length 2 are $a.x.y.b$ and $a.u.v.b$ (which are images of $a/*/*/b$), and from either x or u, there is a mandatory path of length 2 to d. Therefore, we can add a distance 2 d-descendant for the first *-node of $a/*/*/b$, changing it to $a/*[*/d]/*/b$. We can also add a distance 2 e-node to the same *-node.

Algorithm 3. `HandleStarPath`(p, S) where p is a *-path, S is a set of images of p

1: **for** $i = |p| - 1$ to 2 **do**
2: let $N_i = \bigcap_{q \in S} N_{q[i]}$
3: **if** $\exists d > 0$, $\tau \in N(G)$, such that $\forall q \in S$, there is a derived path from $q[i]$ to τ of length d **then**
4: add a distance d τ-descendant to $p[i]$ (provided no such distance $d-1$ descendant of $p[i+1]$ has been added)
5: suppose $\tau_1, \ldots, \tau_k$ are all descendants of $p[i]$ added above
6: **for** $s, t \in [1, k]$ **do**
7: let $u(q)$ be the last common node on the derived paths $q[i] \rightsquigarrow \tau_s$ and $q[i] \rightsquigarrow \tau_t$ /* if there are multiple such derived paths consider each one in turn*/
8: let $d' = min\{|q[i] \rightsquigarrow u(q)| \mid q \in S)\}$
9: **if** $d' > 0$ **then**
10: merge the d'th *-nodes (below $p[i]$) on the *-path from $p[i]$ to τ_s and that from $p[i]$ to τ_t
11: **for** each label $\tau \in N_2$ **do**
12: **if** $\exists \tau' \in N_2$ such that there is a mandatory path from τ' to τ in G **then**
13: let $N_2 = N_2 - \{\tau\}$;
14: **for** each $\tau \in N_2$ **do**
15: add a τ-child v_τ for u_0 and label the edge (u_0, v_τ) with //, if u_0 does not have a τ-descendant already;

Since the derived paths from x to e and from x to d share the node c, and the derived paths from u to e and from u to d share the node v, we can merge the *-node immediately above e and that immediately above d, changing the pattern further to $a/*[*[e]/d]/*/b$ (See Fig.1(d)).

Similarly, for the view V_3 in Fig.6, we can add a g-child to the last *-node of the longer *-path, changing it to V_3', as shown in Fig.6 (f), because the nodes corresponding to the *-node in G, e and h, both have a mandatory child g.

4.6 Expanding *-paths

The procedure `Expand`(V,G) to expand all *-paths is shown in Algorithm 4. It works in two stages. In stage 1, it calls `ExpandPath` to expand each individual *-path $p = u_0/u_1/\cdots/u_k$ by adding an additional path from u_0 to u_k, referred to as the **associate path** of p and denoted $aspath(p)$, and adding descendants to the nodes on the associate path. The basic idea is as follows. If every image of p passes through a sequence of nodes $a_1, \ldots, a_n$, then we add a new path, $aspath(p)$, which passes through a sequence of a_1-node, ..., a_n-nodes (line 1-6). Each edge on the new path is initially labeled //, but it may be changed to /-edge (line 8-9) or a *-path (line 12-13). After that, it calls the procedures `HandleEdge` and `HandleStarPath` to add derived descendants to the nodes on each //-edge or *-path in $aspath(p)$. In the second stage, `Expand` tries to merge the nodes with identical label $g \neq *$, in $aspath(p_1), \ldots, aspath(p_m)$, if the *-paths $p_1, \ldots, p_m$ share some common *-nodes. It does this by checking every

Algorithm 4. `Expand`(V,G)

1: **for all** *-path p in V that existed before Step 4.3 **do**
2: `ExpandPath` (p, G);
3: **repeat**
4: **if** two *-paths p_1 and p_2 share some common *-nodes and $aspath(p_1)$ and $aspath(p_2)$ both have a g-node ($g \neq *$) **then**
5: assume $p_1[i] = p_2[i]$ is the last common *-node of p_1 and p_2
6: **if** Condition A or Condition B is true (see text for description of the conditions) **then**
7: merge the g-nodes on $aspath(p_1)$ and $aspath(p_2)$;
8: **until** no more nodes can be merged as above

Procedure `ExpandPath` (p, G), where $p = u_0/u_1/\cdots/u_k$ is a *-path in V.

1: Find the set C of common nodes, excluding $label(u_0)$ and $label(u_k)$, on all paths in $image(p, G)$;
2: **if** $C \neq \emptyset$ **then**
3: Suppose $C = \{a_1, \ldots, a_n\}$;
4: $a_0 \leftarrow label(u_0)$; $a_{n+1} \leftarrow label(u_k)$; $v_0 \leftarrow u_0$; $v_{n+1} \leftarrow u_k$;
5: Take any path from $image(p, G)$ and check the order of occurrence of $a_1, \ldots, a_n$ on the path; Assume the order is $a_1, \ldots, a_n$.
6: Add an additional path $u_0//v_1//\cdots//v_{n-1}//v_n//u_k$ in V, where $label(v_i) = a_i$ (for all $i \in [1, n]$);
7: **for** ($i = n$ to 0, $i - -$) **do**
8: **if** on every path in $image(p, G)$, a_i is immediately before a_{i+1} **then**
9: change (v_i, v_{i+1}) to /-edge;
10: **else**
11: let $S = \{q[a_i, a_{i+1}] \mid q \in image(p, G)\}$
12: **if** on every path in $image(p, G)$, the distance from a_i to a_{i+1} is m **then**
13: replace (v_i, v_{i+1}) with a *-path of length m, p_i, from v_i to v_{i+1};
14: HandleStarPath(p_i, S)
15: **else**
16: HandleEdge((v_i, v_{i+1}), S)

pair, p_1 and p_2, of *-paths that share some *-nodes as follows. Suppose the last common *-node occurs at position i, that is, $p_1[1] = p_2[1], \ldots, p_1[i] = p_2[i]$ but $p_1[i+1] \neq p_2[i+1]$, and there is a g-node on both $aspath(p_1)$ and $aspath(p_2)$. These two g-nodes can be merged if either of the following conditions are true:

Condition A $\forall q \in image(p_1)$, $g = p_1[j]$ for some $j \leq i$.
Condition B $\forall q \in image(p_1)$ such that $g = p_1[j]$ for some $j > 1$, there is path $q[i].....q[k].g$ in G such that (1) $q[i].....q[k].g$ is on every image q' of p_1 and p_2 that satisfying the condition $q'[i] = q[i]$, and (2) the edges on $q[i].....q[k].g$ are all labeled ? or 1.

Note that $aspath(p_1)$ and $aspath(p_2)$ have the same start node, and merging the g-nodes implies merging the segments from the start node to the g-nodes.

Example 2. Consider the view V_3' and the DTD G_3 in Fig.6. The images of the *-path from the a-node to the b-node in V_3' are $a.x.y.c.h.b$, $a.z.y.c.h.b$, and

$a.y.c.d.e.b$. All of them have $y.c$, thus we can add a path from the a-node to the b-node that passes through a y-node, followed by a c-node, and the edge between the y-node and the c-node is changed to /-edge. We add a g-child to the c-node because there is a mandatory path from e to g and there is a mandatory path from h to g, and for every image the segment from c to b contains h or e. Note that we do not need to add a descendant i to the c-node, because the i-node can be added later using the SC constraint. The resulting view V_3'' is shown in Fig.6 (d). Similarly, for the *-path from a to u, we can add a path from the a-node to the u-node that passes through a y-node, resulting V_3''' as shown in Fig.6 (e). Because the two *-paths share some *-nodes, and their associate paths both have a y-node, and Condition A is satisfied, we can merge the y-nodes in the two new paths. This results the pattern V_3'''' shown in Fig.6 (f).

DAG-patterns. Observe that expanding a *-path may result in an additional path (that does not have $*$-nodes) between the first node and the last node of the $*$-path, thus make the transformed pattern a directed acyclic graph, which we call a DAG-pattern (see V_2' to V_2'' in Fig.6). DAG-patterns are a generalization of TPs, and the definitions of matching, containment mapping, containment, and equivalence (of TPs) can all be trivially extended to DAG-patterns.

4.7 Chasing Using the LWZ CC, SC, PC and FC Constraints

After all the steps above, suppose we have obtained a DAG-pattern V'. We can now use the CC, PC, SC, and FC constraints and chase rules to further chase V', until no more changes can be made. Note the *-nodes in V' can be ignored during the chase. See the details in [5].

Let us denote $chased(V)$ the final view transformed from V. Clearly $chased(V) =_G V$. Furthermore, we have

Theorem 1. *Let* $V, Q \in \widehat{P}^{\{//,[],*\}}$ *be* TP*s satisfiable under* G. $V \subseteq_G Q$ *iff* $chased(V) \subseteq Q$.

Proof outline. Proof of "if" is straightforward. Proof of "only if" is outlined below (the full proof is in extended version). A boolean pattern is a pattern with no distinguished nodes. Given an XML tree t and a pattern P, $P(t)$ returns TRUE if there is an embedding of P in t, and returns FALSE otherwise. Let G be a schema graph, P and Q be TPs satisfiable under G. Suppose $label(\mathtt{DN_P}) = \mathtt{x}$. We modify G into G' by adding a node $z \notin \Sigma_G$ and the edge (x, z) (label this edge with ?, for example), and modify P by adding a z-node as a /-child of $\mathtt{DP_P}$. Similarly, we modify Q by adding a z-node as a /-child of $\mathtt{DP_Q}$. Denote the boolean patterns corresponding to the modified P and Q by P^b and Q^b respectively. Then (1) P^b, Q^b are satisfiable under G'. (2) $P \subseteq_G Q$ iff $P^b \subseteq_{G'} Q^b$. Therefore, for TP containment under schem graphs, we can focus on boolean TPs. For boolean patterns Q and P, a homomorphism from Q to P is the same as a CM except it does not need to be output-preserving. In this proof all patterns are boolean patterns.

Let $Q \in \widehat{P}^{\{//,[],*\}}$. Given a node v in Q, the subtree of Q rooted at v, denoted Q_v, is called a *node subtree* of Q. Given an edge (u, v) in Q, the subtree of Q consisting of the edge (u, v) and Q_v, denoted $Q_{(u,v)}$, is called an *edge subtree* of Q. Let x be a node in schema graph G. Denote G_x the subgraph rooted at x, i.e, the subgraph consisting of all nodes and edges reachable from x.

Lemma 1. *Let v be an x-node in $P \in \widehat{P}^{\{//,[],*\}}$. If P is satisfiable under G, then P_v is satisfiable under G_x. Furthermore, if P cannot be transformed under G, then P_v cannot be transformed under G_x.*

The theorem can be proved using induction on the height of Q based on the following lemma.

Lemma 2. *Let $V, Q \in \widehat{P}^{\{//,[],*\}}$ be boolean patterns, and $V' = chased(V)$.*

1. *For every /-child u of $rt(Q)$, if $label(u) \neq *$, then there is /-child u' of $rt(V')$ such that $label(u') = label(u)$, and $V'_{u'} \subseteq_{Glabel(u)} Q_u$.*
2. *For every //-child u of $rt(Q)$, if $label(u) \neq *$, then there is descendant u' of $rt(V')$ such that $label(u') = label(u)$, and $V'_{u'} \subseteq_{Glabel(u)} Q_u$.*
3. *For each set of *-paths $p_1, \ldots, p_n$ in V' that start from $rt(Q)$ and share common *-nodes, suppose $u_1, \ldots, u_n$ are the end nodes of these paths. Then there is an homomorphism h from $tree(p_1, \ldots, p_n)$, the subpattern consisting of $p_1, \ldots, p_n$, to V' such that $V'_{h(u_i)} \subseteq_{label(u_i)} Q_{u_i}$ for all $i \in [1, n]$.*

4.8 Finding MCRs under DTDs

Let $V, Q \in \widehat{P}^{\{//,[],*\}}$ be the view and the query, respectively, under DTD G. Let $V' = chased(V)$. We need to consider two different cases: when $label(\mathtt{DN}_{V'}) \neq *$ and when $label(\mathtt{DN}_{V'}) = *$.

First consider the case $label(\mathtt{DN}_{V'}) \neq *$. In this case, we can easily prove the following result using Theorem 1:

Corollary 1. *A* CR *of Q using V under G is a* CR *of Q using V' (without* DTD*s). A* CR *Q' of Q using V' (without* DTD*s) is a* CR *of Q using V under G provided $Q' \circ V$ is satisfiable under G.*

Therefore, to find the MCR of Q using V under G, we only need to find the MCR of Q using V'. Note that although V' may be a DAG-pattern, the method of useful embedding can still be used without change.

Next consider the case $label(\mathtt{DN}_{V'}) = *$. In this case, we need to find the set $L(\mathtt{DN}_V)$ of all possible labels for $\mathtt{DN}_V$, and then break the view V into the union of $V_1, \ldots, V_n$ such that each V_i is identical to V except $\mathtt{DN}_{V_i}$ is labeled with a distinct label in $L(\mathtt{DN}_V)$. We chase each V_i to V'_i, and find the MCRs of Q using each V'_i as in the case where $label(\mathtt{DN}_{V'}) \neq *$, and union all of them. The following example demonstrates the necessity of breaking V into $V_1, \ldots, V_n$ before transforming them using G.

Example 3. Consider the DTD G_1 in Fig.1. Suppose we have the view $V = a/*/*[b]$, and the query $Q = a/x[/c]/y/b$. Under G_1, we will transform V into

$V' = a/*[*[d]/e]*[b]$ (the tree is similar to P_1'' in Fig.1 except the *-node above b, rather than the b-node, is circled). Thus we cannot find a useful embedding of Q in V'. However, under the DTD G_1, we can break V into the union of $V_1 = a/*/y[b]$, $V_2 = a/*/v[b]$. Now V_1 and V_2 can be transformed into $V_1' = a/x[c/[d]/e]/y[b]$ and $V_2' = a/u/v[d][e][b]$ respectively. Thus we can find the CR y/b using V_1', which is also a CR of Q using V under G_1.

5 Conclusion

We revisited the issue of finding the maximal contained rewriting using views for tree pattern queries, and extended the work of [5] to TPs in $\widehat{P}^{\{//,[],*\}}$. As can be seen, the additional * in the queries and views makes the rewriting process considerably more complicated in the presence of DTDs. Our algorithms also provide a means for testing containment under acyclic DTDs for TPs in $\widehat{P}^{\{//,[],*\}}$.

Acknowledgement. This work is supported by Griffith University Research Grant GUNRG36621.

References

1. Amer-Yahia, S., Cho, S., Lakshmanan, L.V.S., Srivastava, D.: Minimization of tree pattern queries. In: SIGMOD, pp. 497–508 (2001)
2. Arion, A., Benzaken, V., Manolescu, I., Papakonstantinou, Y.: Structured materialized views for XML queries. In: VLDB, pp. 87–98 (2007)
3. Halevy, A.Y.: Answering queries using views: A survey. VLDB J. 10(4), 270–294 (2001)
4. Lakshmanan, L.V., Ramesh, G., Hui (Wendy) Wang, Z.J.Z.: On testing satisfiabilty of tree patterns. In: VLDB (2004)
5. Lakshmanan, L.V.S., Wang, H., Zhao, Z.J.: Answering tree pattern queries using views. In: VLDB, pp. 571–582 (2006)
6. Mandhani, B., Suciu, D.: Query caching and view selection for XML databases. In: VLDB, pp. 469–480 (2005)
7. Miklau, G., Suciu, D.: Containment and equivalence for a fragment of XPath. J. ACM 51(1) (2004)
8. Xu, W., Özsoyoglu, Z.M.: Rewriting XPath queries using materialized views. In: VLDB, pp. 121–132 (2005)

Addressing New Concerns in Model-Driven Web Engineering Approaches

Nathalie Moreno[1], Santiago Meliá[2], Nora Koch[3,4], and Antonio Vallecillo[1]

[1] Universidad de Málaga, Spain
[2] Universidad de Alicante, Spain
[3] Luwig-Maximilians-Universität München, Germany
[4] Cirquent GmbH, Germany
vergara@lcc.uma.es, santi@dlsi.ua.es,
kochn@pst.ifi.lmu.de, av@lcc.uma.es

Abstract. In the last few years, almost all model-driven Web Engineering approaches have evolved in response to the new challenges of Web systems design, which are due to new requirements and implementation technologies in the Web domain. The evolution implies the extension and adaptation of current approaches, in terms of new models, transformations and processes in order to incorporate new concerns or aspects. Such changes in a methodology are a risky and error-prone process. In this paper, we analyze different alternatives to address the evolution and in particular, the addition of a new concern in a Model-Driven Web Engineering approach: (a) extending the original method with an additional modeling concern, (b) merging the original proposal with another approach covering the specific concern and, (c) finally, we propose a an interoperable and architectural-centric approach that aims to reduce the impact of adding a new concern. We discuss the main advantages and drawbacks of each alternative.

Keywords: Separation of Concerns, Web Engineering, Model-Driven Software Development, Metamodeling, Model Transformations.

1 Introduction

Model-driven development (MDD) is an approach to software development that uses models, metamodels and model transformation as key elements of the development process [2]. It incorporates a higher level of abstraction in the specification of systems guided by the *separation of concerns principle* and allows the (semi)-automated derivation of the final implementation code. In this sense, most of the existing Web engineering approaches match the MDD philosophy, because they address the development of Web applications using separate models to describe the different concerns that constitute Web systems. Furthermore, they provide model compilers that permit automatic generation of system implementations from high-level models.

Traditionally, the majority of Web engineering proposals have considered the content, navigation and presentation models, as the most relevant concerns in the design of a Web application. However, due to new requirements and implementation

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 426–442, 2008.

technologies in the Web domain, Web engineering approaches have to evolve. Following a model-driven approach, the evolution may imply: (a) the definition of new models and modeling elements that capture additional requirements; (b) the redefinition of the metamodel for handling these additional features; (c) the adaptation of the development process to incorporate the new concern and the information it represents; (d) the adaptation of the modeling process and code generation tools that support the method.

To avoid the need for these changes, some approaches have studied the practical viability of merging their design methods with others [10,11]. Indeed, one of the main advantages of using MDD in the Web application domain is the possibility of establishing and exploiting the synergies existing among the approaches using simple model transformations. In this paper we will discuss three different alternatives for addressing the evolution of Web engineering methods: (1) extending the original method with an additional modeling concern, (2) merging the original proposal with another approach covering the specific concern and, (3) finally, we propose an interoperable and architecture-centric process that aims to reduce the impact of adding a new concern. For the last approach we propose to use the (Web Engineering Interoperability) WEI common metamodel [11,18] which encapsulates the concepts of almost all Web engineering approaches and the architecture models of the (Web Software Architecture) WebSA approach [9]. The three alternatives will be illustrated by introducing a new concern to model the business process operations performed within Web applications.

The remainder of this document is structured as follows. Section 2 classifies the concerns involved in the design of a Web application based on its dependency relationships with other concerns and the impact that addressing these relationships would have on a method. Section 3 analyzes three ways to address the evolution of a Web engineering method. In Section 4 the main advantages and drawbacks of each alternative are discussed. Then, Section 5 relates our work with other similar studies. Finally, Section 6 draws some conclusions and outlines some future research activities.

2 Classification of Concerns

Adding a new concern to a Web engineering method, with a well known and structured separation of concerns, models and code generation process, is not a trivial task. The complexity of this process will depend largely on the nature of the concern considered and the relation it has with the current ones. Based on these considerations, we distinguish the following categories of concerns:

Dependent concern. This concern has one or more dependency relationships with others. Dependency relationships establish an order relation when defining the system models, since they force the designer to define firstly the independent concerns and subsequently the other concerns that depend on these. In addition, this kind of concern involves managing consistency at model level between related concerns. Modifying some modeling elements of a concern may have a knock on effect causing changes in other concerns that depend on it. For example, both in the UML-based Web Engineering (UWE) [6] and the OO-H [7] methods the navigation model is derived in

part from the content or conceptual model respectively (i.e. there is a dependency relationship between elements of both models). Therefore, when a class of the content/conceptual model is deleted, all navigational classes or relations that were defined as a view on elements of that model must also be deleted. Nowadays, the addition of a dependent concern to a methodology is done by hand. This process involves the definition of an extension of the existing metamodel that addresses it and also the specification of possible relations between the new metaclasses and the existing ones.

Replacement concern. This is a concern that replaces another previously defined for the same method but it offers a new viewpoint to address the modeling requirements. When the new concern represents a total change with respect to the previous one, its corresponding metamodel is replaced by the new one. In other cases, the original metamodel is only subject to certain modifications. In fact, a replacement concern may also be a dependent concern that has to maintain consistency with other concerns at the metamodel level, i.e., respecting the relations that the concern being replaced had with the others. Presentation is an illustrative example of replacement concern. The advent of the Web 2.0 has shown that existing Web methods require more expressive models for addressing the user interface of a system due to traditional mechanisms are now inadequate [8].

Orthogonal concern. It represents a new concern that models a feature of a system which is completely independent of all the others. In this case an independent metamodel package and corresponding transformations are defined. They capture the aspects of the new concern without affecting the other packages of the metamodel. In this way, not only the design time but also the development time can be optimized since developers can work simultaneously, each one on a different concern. Software architecture is, for example, an orthogonal concern to navigation and presentation.

3 Addressing a New Concern

Business processes have gained a lot of importance in Web applications. Addition of business processes to modern Web applications entails new challenges to the development of Web applications. Following the previous classification, the business process concern can be considered a dependent concern because it has a strict dependency relationship with the content model. Hence, current Web engineering methods have to propose extensions that include appropriate modeling concepts specifically tailored to cope with this kind of requirements and appropriate horizontal and vertical transformations. Horizontal transformations are those between models at the same level of abstraction; vertical transformations “refine” system models, adding information and therefore making them closer to the technology or implementation platform. In this section we present three different ways to address a Web method extension, illustrated by a running example.

The example models a very simple music portal Web application that allows users to search music albums and songs by name. The search result is presented as a list of matching albums/songs that provides links to a detail page for each album. The album

detail pages show the title of the album, the name of the artist, the list of songs and the album's price. For the sake of reducing complexity, in this example each album has only one artist. The new concern, introducing *modeling of business process operations*, is illustrated by the functional requirement to allow registered users to buy albums, which then can be downloaded as archive files containing MP3s. The following list gives a short informal description of the requirements.

- A user becomes a registered user by logging in. Unregistered users can register with a username and a freely chosen password.
- If the user has already bought the album then a download link is shown. Otherwise, there will be a link for buying the album. Only full albums can be downloaded.
- Each registered user has a credit account that is used to buy albums. The credit account can be recharged by credit card payment.
- The links for logging in or out, for registering and to the user's account page are always shown. This also holds for the album search box.

3.1 Extending UWE with an Additional Modeling Concern

Similar to other Web engineering methods, UWE [6,16] addresses the different concerns of a Web application by the construction of different models for the content, the navigation structure, and the presentation. The distinguishing feature of UWE is the use of standards, in particular the Unified Modeling Language (UML [15]). It is UML2 compliant since modeling with UWE is based on a UML2 profile, which is defined on a lightweight extension of the UML metamodel (see [6] for more details about the UWE method and notation). UWE evolves, in the same way other Web engineering approaches do, integrating modeling facilities for additional concerns. In fact, UWE recently integrated techniques for modeling business processes, which are driven by user actions. In the following we show UWE models by example and explain how the new concern is added to the UWE approach [7].

A content model built with UML classes models the content of Web applications in UWE. In our running example the content is modeled by classes such as Album, Song and Artist (Fig. 1). Content classes are shown in a UML class diagram together with their relationships (associations, composition, inheritance, etc.).

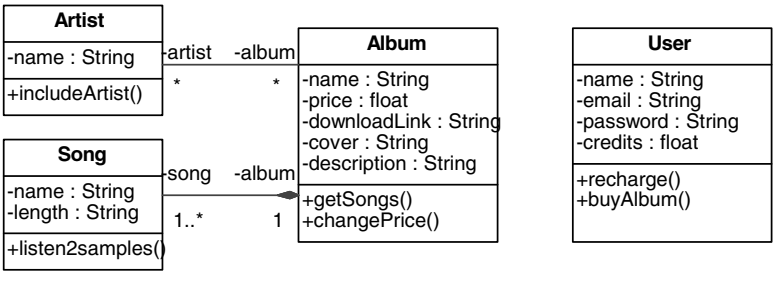


Fig. 1. UWE content mod el for the music portal example

The navigation model is based on the content model and represents the navigation paths of the Web application. A navigation class represents a navigable node in the Web application and is associated to a content class containing the information of the node, e.g. navigation classes Album and Song. Navigation paths representing direct links between two navigation nodes are represented by associations called navigation links (for simplicity the corresponding stereotype is omitted in Fig. 2). Additional navigation nodes are access primitives used to reach multiple navigation nodes (index and guided tour) or a selection of items (query). Menus model alternative navigation paths. Examples of access primitives in the navigation model of the music portal are AlbumQuery and AlbumIndex; example of menu is the MainMenu.

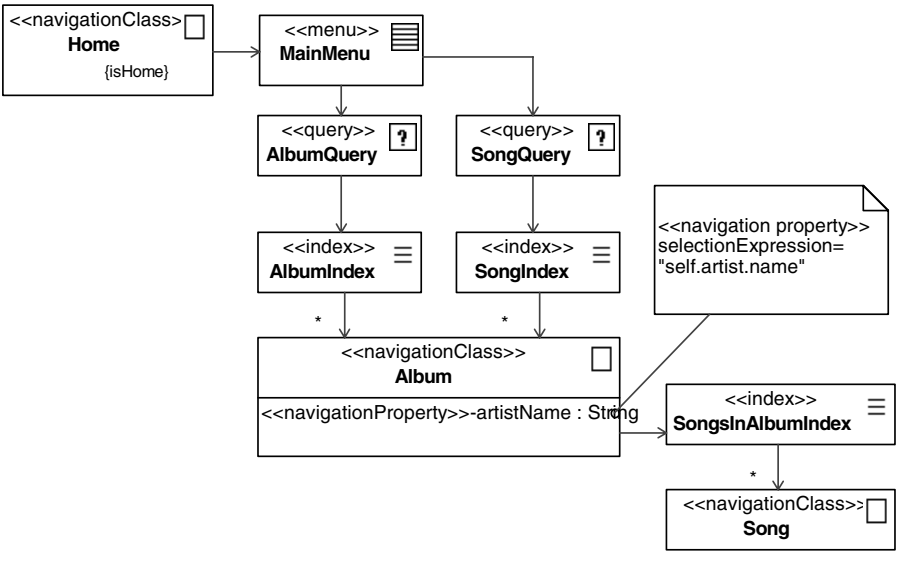


Fig. 2. UWE navigation model for the music portal example

The presentation model is used to sketch the layout of Web pages associated to the navigation nodes. For an example of a presentation model the reader is referred to [6].

Web applications are no longer built for browsing information, they are instead required to offer to the user more and more functionalities, such as search facilities and business process operations. This kind of operations constitutes therefore a new concern that needs to be modeled for certain Web applications. UWE is extended to cover this new concern as follows:

- A process model is added. It includes the process classes which contains the required data for the process, such as Login, BuyAlbum, BuyAlbumConfirmation, and InsufficentCreditMessage (see Fig. 3)[12].
- A navigation model is enriched with process classes indicating entry and exit points of the processes, e.g. process classes Register, Login, BuyAlbum, etc. Process classes and process links are shown in Fig. 4.

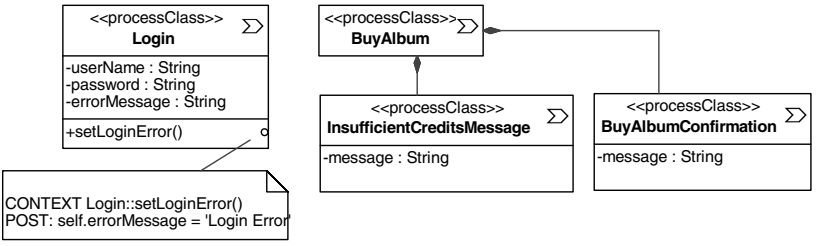


Fig. 3. Excerpt of the UWE process model of the music portal example

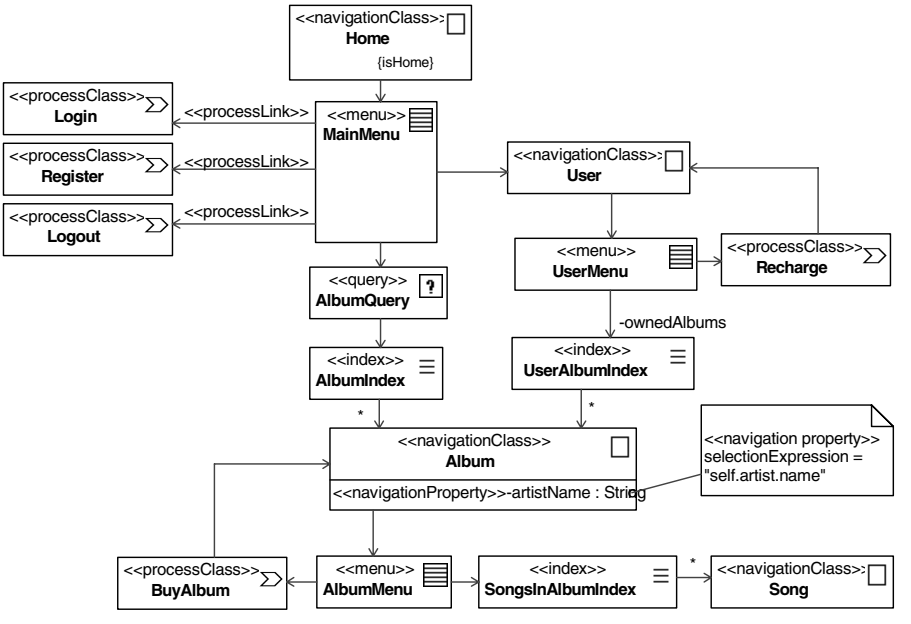


Fig. 4. Extended UWE navigation model of music portal example

- A workflow is specified which models the activities of the business process logic, e.g. describing the Login and Logout processes, the registration or the download of an album (not shown here due to space limitations).

Adding a concern in UWE means that new elements and relationships have to be defined and included in the UWE metamodel. Fig. 5 illustrates how the Navigation package of the UWE metamodel is extended with ProcessClass and ProcessLink.

The UWE approach defines in addition to the UML profile and the UWE metamodel, a model-driven process for the development of Web systems. The process relies on models and model transformations following the MDA principles and using several other standards [13]. A set of design model types is used in UWE to model the different concerns of the Web applications. The transformations for refining the design models comprise mappings from the content to the navigation model, refinements of

the navigation model, and from the navigation into the presentation model. In UWE, an initial navigation model is generated based on classes of a stereotyped content model. This generation step can be rendered as a transformation Content2Navigation. From a single content model different navigation views can be obtained, e.g., for different stakeholders of the Web system.

Starting with this basic navigation model, it can be further refined by a set of vertical transformation rules that can be applied fully automatically. These rules include for example the insertion of indexes and menus. Similarly, presentation elements are generated from navigation elements. For example, for each link in the navigation model an appropriate anchor is required in the presentation model. So far, look and feel aspects have to be added manually. Transformations are defined as OCL constraints (by preconditions and postconditions) in UWE and are implemented either in Java in plug-ins for CASE tools such as MagicDraw and ArgoUML, or in the model transformation language ATL [1].

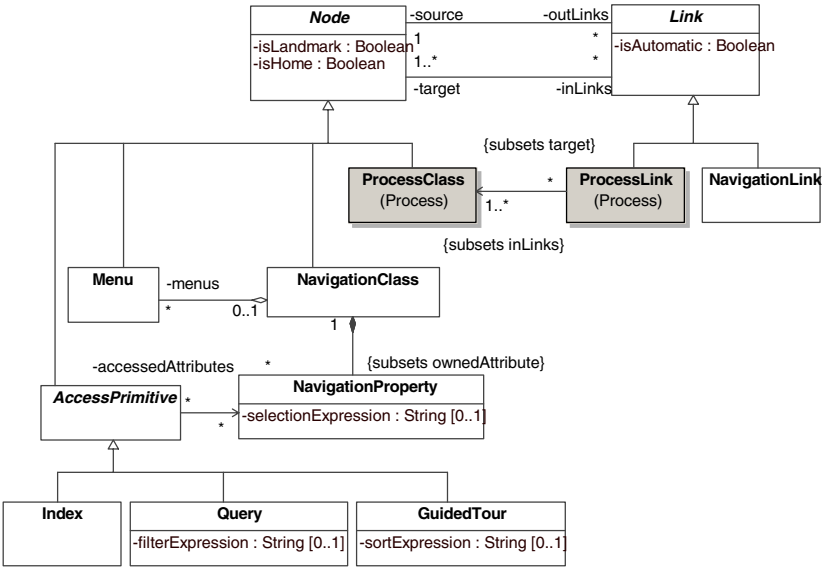


Fig. 5. Excerpt of the UWE metamodel (metaclasses for new concern are colored)

The UWE MDD process comprises also an integration step. The aim is the creation of a single model for validating the correctness of the functional models. This "big picture" model is a UML state machine, representing the content, navigation structure, and the business processes of the Web application as a whole. A model checker like Hugo/RT [6] can check this state machine. In order to transform platform independent to platform specific models additional information of the platform is required. It can be provided as an additional model or it is implicitly contained in the transformations. For mappings from design models to code UWE also uses vertical transformation rules written in ATL that generate Java Beans and JSPs.

The activities required for the extension of the UWE approach with the new concern *business processes* are detailed in the following indicating which is the expertise required for such an extension. This extension process is shown in Fig. 6. The UWE expert has to extend the profile and the metamodel. Model-to-model and model-to-code transformations need to be defined by the transformation expert. Finally, the tool builder has to introduce the corresponding changes in the UWE tool to support modeling and generation of Web applications including business processes. The steps outlined previously would be more complex and costly to implement if the separation of concerns is not previously established as it is in UWE (i.e., where concerns are grouped in a single model). In these cases it may even be necessary to reorganize the models and transformations defined initially for the approach in order to be able to carry out the required extension.

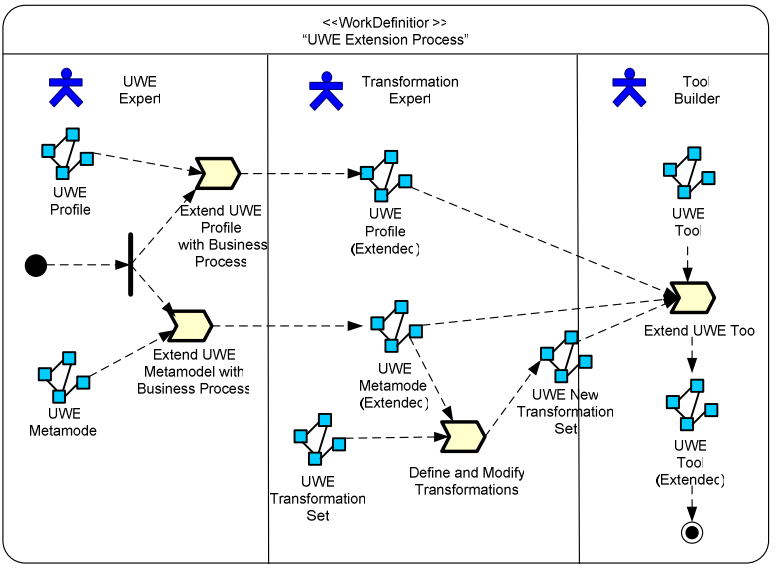


Fig. 6. The UWE extension process for adding the business process concern

3.2 Merging UWE Models with a Model of Another Approach

Instead of extending the UWE approach with new modeling constructs, an alternative is to use modeling features of another approach covering the new concern, i.e. the business process in our example. We choose OOWS [14] to illustrate the merging process. OOWS models business processes in BPMN [3] notation, which is used to generate WS-BPEL code. For OOWS, the business process is a concern depending on their functional and structural models (this is the most complex case of the three types of concerns we looked at in the classification presented in Section 2).

Thus, merging the OOWS business process with UWE requires firstly identifying those UWE models with direct correspondences to OOWS models. Identifying those models is strictly necessary for generating the process model in OOWS and requires the definition of a set of horizontal transformation rules from the UWE content and

requirements models to the OOWS business process model in order to implement the correspondences. Once these transformations are defined, OOWS can generate the code corresponding to the business logic that has been modeled.

However, the objective of merging UWE and OOWS does not end there. It is required in some way to integrate the information of the process model with the other UWE concerns (i.e., to link the business logic with the other views of the UWE system models). To do this, the next step is to determine what relation the new concern has with those already considered by the method. These dependencies must be identified and dealt with at the modeling level. In the case of UWE, it was decided that the process model would provide input for the navigation and presentation models. In order to keep those dependencies, it is necessary to define transformation rules from the OOWS business process model to the UWE navigation and presentation models as well as to establish some links between the code generated using UWE that relates the user interface with the implementation of the business logic generated with OOWS.

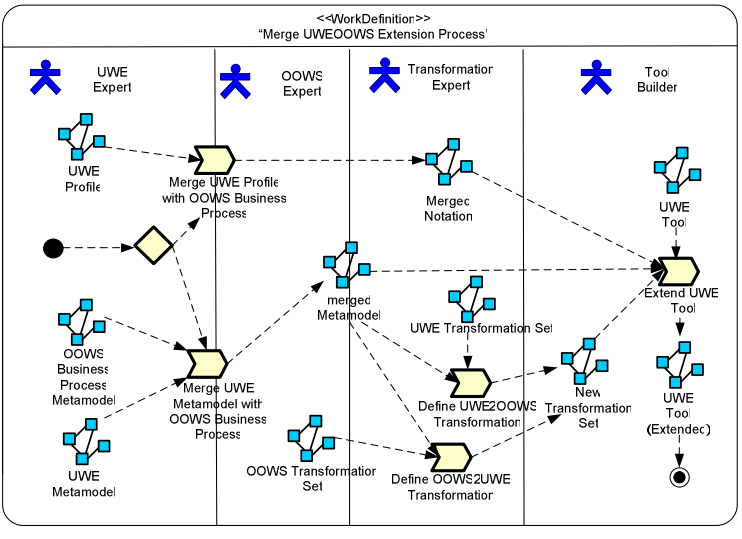


Fig. 7. The Merge UWEOOWS extension process for adding the business process concern

Fig. 7 illustrates these tasks and the experts responsible for them. Essentially, merging the UWE approach with a different one, such as for example OOWS, for modeling processes, means that the UWE expert must combine the UWE metamodel with the subset of the OOWS metamodel that deals with modeling Business Process. The resulting combination requires that a certain graphical notation must be incorporated into the new modeling elements of the UWE metamodel dealing with the processes. If the merge is done at the graphic design level, the transformations previously referred to would still need to be defined so that the UWE modeling elements correspond to the OOWS modeling elements and viceversa: in Fig. 7, the "transformation expert" is in charge of this task. Finally, the tool builder may construct a new code generation tool for UWE and the OOWS process model added,

using the original tool and the newly defined notation and set of transformations as the basis. Obviously, all this process is simplified considerably if instead of addressing a dependent concern we address an independent concern since the first one requires working with at least two tools.

3.3 Adding the Process Concern Using the WEISA Approach

As an alternative to extending UWE and merging UWE and OOWS, this section presents a new model-driven Web approach called WEISA that aims at obtaining interoperability and extensibility through a common metamodel (defined by WEI [18, 11]) based on the consensus of the most important Web methodologies regarding functional concerns; (2) WEISA also proposes a model-driven development process that provides the necessary extensibility able to incorporate a new concern with the lowest possible cost. Moreover, this process introduces an early representation of the software architecture guided by WebSA [9] which permits to reduce the complexity of the Web design with a small set of models and provides a closer match between the system modeled and the final implementation.

Fig. 8 presents the WEISA extension process which permits to add a new concern that comes from any MOF-compliant methodology in two different ways: (1) if WEISA contemplates the mechanisms for modeling this concern, it only requires the definition of horizontal transformations from the models that the initial method does contemplate; (2) If WEISA does not contemplate the requested concern, it is necessary extending their metamodel and studying the extension with a third model proposal that defines this concern. The last step consists on establishing the vertical transformations for introducing the new concern into the different types of WEISA components.

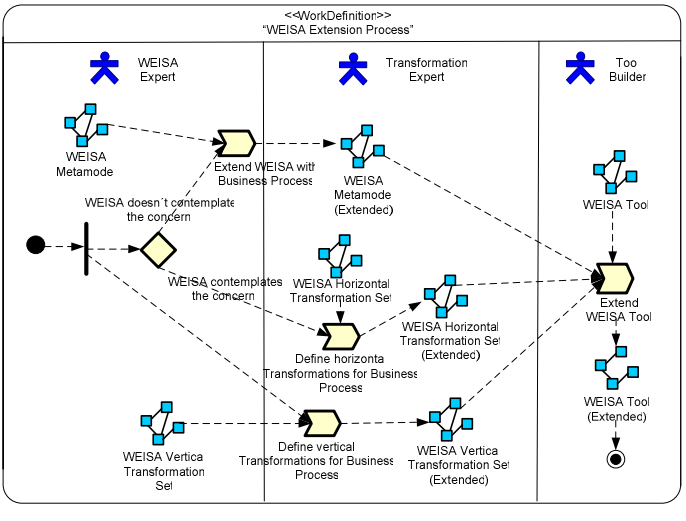


Fig. 8. The WEISA extension process for adding the business process concern

In the case study of this paper, we can apply the first possibility, in which WEISA represents the new process concern with its own process models, and obtains the rest of models (i.e. presentation, navigation and domain models) from UWE through a set of horizontal model transformations called UWE2WEISA. These transformations are defined in ATL. In fact, one of the major advantages of our proposal is its ability to design and implement Web applications reusing existing models from other Web engineering methods.

WEISA represents the process concern using the BusinessLogicStructure Model of WEI shown in Fig. 9. This model is a UML2 [12] stereotyped class diagram that establishes the main classes and operations that implement the business logic from our application. From here, we describe the behavior of each method by means of an stereotyped activity diagram, where the new stereotypes model the operations invoked from the user interfaces and the structure data returned by the business logic and visualized in the interface. Here we describe the structural business aspects, without delving into the behavioral business aspects. However, the interest reader in WEI profiles may refer to [12] for more details.

At the same time, the WEISA designer defines the software architectural model (called Configuration Model, CM) which uses the Web component as architectural unit, and defines around it a set of specific type of components of the Web application family (e.g. Controller, ServerPage, ProcessComponent, etc.). These kinds of components allow structuring the functionality of a Web application according to a given architectural style. Thus, this model provides a representation of the software architecture of the system, orthogonally to its functionality, thereby allowing for its reuse in different Web applications.

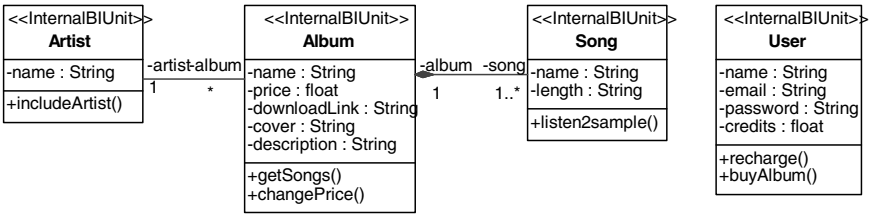


Fig. 9. The Business Logic Structure model of music portal example

Fig. 10 depicts the CM of the case study. The front-end part of the model shows a ServerPage component which receives the user's requests and renders the response in a PC browser. The ServerPage also has a reference to EntityData in order to represent the functionality and is responsible for sending messages to the Controller. At this point, each ProcessComponent PC receives the requests through the BusinessFaçade ServiceInterface from the Controller, and re-sends them to the Entity. Finally, the Entity references to a DataAccessComponent called DAC in order to store and to recover data from a database.

After the models defined by the WEISA designers are completed, these become the entry point of the Merge2Design transformation which converts the functional and

architectural models into a detailed design model represented by the WEISA Integration Model.

This complex and extensible transformation is based on the concatenation of a set of smaller transformations associating each type of component with a concern (e.g. the data model is related to the data access component, the ServerPage to the model presentation, the EntityWeb to the domain model, the ProcessComponent to the process model, etc.). This provides us with the integration model representing the design components that constitute the Web application where we have introduced the functional content from the functional models. Finally, this process establishes a model-to-text transformation called Integration2Platform that allows us to obtain the final implementation. This is a model-to-text transformation that obtains the code from the integration model and the functional models requested by different concerns.

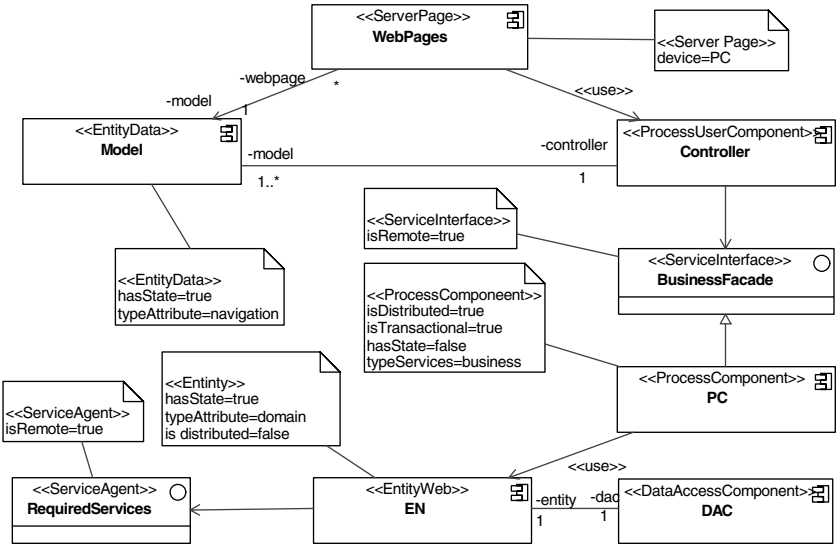


Fig. 10. The Configuration Model of the music portal example

In the case study, we focus on the Merge2Design Tranformation part in charge of introducing the process concern into the design model. More specifically, the ProcessComponent and the EntityWeb are the only ones which obtain the data from the process concern. Fig. 11 shows a fragment of integration model that represents the ProcessComponent and EntityWeb components obtained from the BusinessLogicStructure classes such as Artist, Album and Song.

Finally, this process establishes a model-to-text transformation called Integration2Platform that allows us to obtain the final implementation. This transformation obtains the code from the integration model and the functional models requested by different functional concerns.

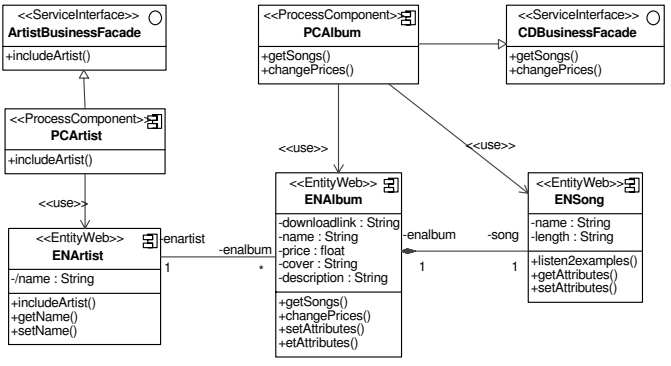


Fig. 11. A fragment of the Integration Model of the music portal example

Table 1. Comparing alternatives for adding a new concern

Criteria	Proprietary Method (UWE)	Merging 2 Methods	WEISA
Abstract syntax	Adding modeling elements to original metamodel.	Putting both metamodels in relation (when it is possible) maybe by means of a third one.	Defining an independent metamodel and putting it in relation with the others.
Notation	Extending UML profile with new artifacts.	Two options: (a) Using, notations of each method for modeling the concerns (b) defining a new notation for modeling element results of the merge.	Extending UML profile with new artifacts.
Transformations	Vertical transformations for the concern added and modifying the others in order to guaranty the consistency between all concerns.	Defining vertical transformations for the concern added; one transformation for each method.	Defining vertical transformations for concern added without modifying previous one. Other methods will benefit of new trans-formation without change.
Tool	Extending graphical interface and transformation engine of CASE tool to include modified vertical transformations.	Importing ad-hoc of all the models in a new environment The total set of transformations rules also have to be included in the merging environment.	Extending graphical interface and adding only one transformation to the transformation engine of the CASE tool.
Interoperability/ Extensibility	Representing different concerns with its own models.	Representing the concerns with both methods.	Representing different concerns with WEISA or with any approach with a MOF metamodel.

4 Analyzing and Comparing the Three Alternatives

Like most design decisions, the choice of any of the previous alternatives may have far-reaching consequences for a Web engineering method. We can indeed find

arguments for and against each way proposed. In this sense, we hope this study will not only make it easier to understand the relative strengths or weakness of each strategy, but will also serve as a basis for analyzing the obstacles to evolve or maintain a method. As a summary, Table 1 identifies the most relevant features of a model-driven Web engineering approach, which can be affected as a consequence of the evolution process when addressing a new concern. Although there are other evaluation criteria, the following may provide a good basis for comparing the three alternatives:

Abstract syntax. The abstract syntax of a modeling language describes the vocabulary of concepts provided by the language, the definition of such concepts and the relationships that exist between them. It also establishes how the concepts may be correctly combined to create models by means of a metamodel definition. Therefore, adding a new concern may involve a review of the current syntax in order to identify if the new concern added requires specific modeling concepts. Furthermore, the method must decide where the new artefacts are going to appear within a system description, (i.e, as a part of an existing concern, in a new model, etc.).

Notation. All methodologies provide a notation that facilitates the presentation and construction of models in their associated languages. There are two main types of concrete syntax or notation: textual and visual. In any case, when the abstract syntax is affected by a change then normally this change will carry out notation changes.

Development process. In model-driven Web development methods, changes to abstract syntax must be also mapped into changes to the development through the definition of transformation rules: (a) on the one hand, it may imply the definition of vertical transformations that convert models from a higher to a lower level of abstraction and (2) on the other hand, it may suppose the definition of horizontal transformations which describe mappings between models of the same level of abstraction.

Tool. New transformation rules must be integrated into the CASE tool that supports the Web engineering method. However, not only the code generation engine requires changes but also the model editor of the same tool. This task may be more complex and time-consuming than we expect, especially when the CASE tool design was not prepared to assume the evolution.

Interoperability/Extensibility. The ability to extend a system and the level of effort required to implement the extension is a measure of its extensibility. The central theme is to provide for current and future changes while minimizing the impact to the existing proposal.

All three alternatives have their own advantages and disadvantages, and therefore it is particularly difficult to offer general guidelines on when a designer should opt for one or for another. At first sight it could be considered, for example, that in those cases in which we deal with an *orthogonal concern* the least expensive decision at the implementation level is the merging or the WEISA approach. On the contrary, in the cases where we deal with the *dependent concern*, it may be more appropriate to directly extend the methodology.

Although the influence of the type of concern is extremely relevant for implementing the method extension, other factors such as the semantic distance between the source and target metamodels may be equally relevant. This semantic distance would determine the complexity of the transformations to be implemented.

5 Related Work

As far as we know, there are not studies in the Web engineering domain that analyze the real impact of extending a design method with an additional concern. However, we have found proposals in other research areas that address relevant and related issues to the proposed research topic of this article.

In the product-line context, released products are built on various versions of core assets and glued together with product specific code. Thus, the domain evolution problem (i.e., the metamodel evolution in our case) arises when existing product-line must be extended and/or refactored to handle unanticipated requirements. Clements and Northrop list in [4] a set of metrics to measure the opportunities for future asset or infrastructure of a certain product-line. These metrics can be adapted and applied on the three alternatives we propose here.

In [5] we find an interesting classification of the changes that may occur in a metamodel. According to the authors, changes can be grouped into three categories: (a) changes that preserve the semantics of the metamodel, (b) changes that introduce new classes and/or properties to the metamodel, (c) changes that remove/destroy elements of the metamodel. Adding a new concern has always effects on a metamodel. In particular, orthogonal and dependent concerns introduce new artefacts and extend the semantic of the original metamodel. On the contrary, replacement concerns may modify it by removing key elements of the metamodel so they require especial attention.

On the problem of synchronizing models with evolving metamodels, [17] introduces and outline an approach to addressing it efficiently. The authors aim to minimize the effort required to perform model migration in face of metamodel changes (some of them are shown in Table 1).

6 Conclusions and Future Work

Model-driven development (MDD) is being adopted due to its advantages of portability and facilities for the integration of models produced by different approaches, which is supported by model transformations. In particular, MDD is applicable in the Web application domain as a very clear separation of concerns is one of the main characteristics of almost all Web engineering methods.

Another advantage of MDD is the flexibility when introducing a new concern that is part of the evolution of Web methodologies. Including a new concern may be more or less difficult depending on the type.

This paper presents a classification of concerns and a discussion on three different alternatives for addressing the evolution of Web engineering methods. The more traditional way is the adaptation of the own method extending it with an additional

modeling concerns. Another alternative is merging the original proposal with another approach covering the specific concern. Finally, we propose a new approach called WEISA based on an interoperable and architecture-centric process that aims to reduce the impact of adding a new concern. A table comparing the three alternatives is presented as well.

We will continue working on the variants detailed, completing the models with dynamic aspects and we will define the complete set of transformations required for the MDD process. In addition, adding a new concern to a Web engineering approach may in general affect three different dimensions: the way of modeling, the way of working (i.e., the associated methodology), and the supporting environment and tools. The approach presented here has focused on the way of modeling first, because the changes to other two dimensions depend on the alternative selected to incorporate the new concern at the modeling level. Once we have identified the alternatives, the next step is to study their potential impact on the other two dimensions.

Acknowledgements. This work has been partially funded by projects Desarrollo de Software para Sistemas Distribuidos Peer-to-Peer (TIN2005-09405-C02-01), MOVIS (P07-TIC-03184), EU project SENSORIA (IST-2005-016004), ESPIA (TIN2007-67078). We would also like to thank the reviewers for their insightful comments and suggestions.

References

1. ATL ATLAS Transformation Language project, `http://www.eclipse.org/m2m/atl/`
2. Bézivin, J.: In Search of a Basic Principle for Model Driven Engineering. UPGRADE V(2), Novática (2004)
3. Business Process Modeling Notation (BPMN) Version 1.0 - OMG Final Adopted Specification (February 6, 2006)
4. Clements, P., Northrop, L.: Software Product Lines: Practices and Patterns. Addison-Wesley, Reading (2001)
5. Gruschko, B., Kolovos, D., Paige, R.F.: Towards Synchronizing Models with Evolving Metamodels. In: Proc. of 11th Workshop on Model-Driven Software Evolution (MODSE 2007) (2007)
6. Koch, N., Knapp, A., Zhang, G., Baumeister, H.: UML-based Web Engineering: An Approach based on Standards. In: Rossi, G., Pastor, O., Schwabe, D., Olsina, L. (eds.) Web Engineering: Modelling and Implementing Web Applications. Springer, Heidelberg (2007)
7. Koch, N., Kraus, A., Cachero, C., Meliá, S.: Integration of Business Processes in Web Applications Models. Journal of Web Engineering (JWE) 3(1), 22–49 (2004)
8. Linaje, M., Preciado, J.C., Sánchez-Figueroa, F.: Engineering Rich Internet Application User Interfaces over Legacy Web Models. IEEE Internet Computing 11(6), 53–59 (2007)
9. Meliá, S., Gomez, J.: The WebSA Approach: Applying Model Driven Engineering to Web Applications. Journal of Web Engineering 5(2), 121–149 (2006)
10. Meliá, S., Kraus, A., Koch, N.: MDA Transformations Applied to Web Application Development. In: Lowe, D.G., Gaedke, M. (eds.) ICWE 2005. LNCS, vol. 3579, pp. 465–471. Springer, Heidelberg (2005)

11. Moreno, N., Vallecillo, A.: Towards Interoperable Web Engineering Methods. Journal of the American Society for Information Science and Technology (JASIST) 59(7), 1073–1092 (2008)
12. Moreno, N., Vallecillo, A.: Modeling Interactions between Web Applications and Third Party Systems. In: Proc. of the 5th International Workshop on Web Oriented Software Technologies (IWWOST 2005) (2005)
13. Object Management Group (OMG), http://www.omg.org
14. Torres, V., Giner, P., Pelechano, V.: Web Application Development Focused on BP Specifications I Taller sobre Procesos de Negocio e Ingeniería del Software (PNIS) (2007)
15. Unified Modeling Language (UML). Superstructure, version 2.1.2. Specification, OMG, http://www.omg.org/cgi-bin/doc?formal/07-11-01
16. UML-based Web Engineering (UWE), http://www.pst.informatik.uni-muenchen.de/projekte/uwe/
17. Wachsmuth, G.: Metamodel Adaptation and Model Co-adaptation. In: Ernst, E. (ed.) ECOOP 2007. LNCS, vol. 4609, pp. 600–624. Springer, Heidelberg (2007)
18. Web Engineering Interoperability (WEI), http://www.lcc.uma.es/~nathalie/WEI/

A Web-Based Automated System for Industry and Occupation Coding

Yuchul Jung[1], Jihee Yoo[1], Sung-Hyon Myaeng[1], and Dong-Cheol Han[2]

[1] School of Engineering, Information and Communications University,
119, Munjiro, Yuseong-gu, Daejeon, 305-732, Korea
{enthusia77,zzihee5,myaeng}@icu.ac.kr
[2] Information System Development Division, Korean National Statistics Office,
139, Seonsaro, Seo-gu, Daejeon, 302-701, Korea
dchandc@hanmail.net

Abstract. This paper describes our newly developed Automated Industry and Occupation Coding System (AIOCS). The main function of the system is to classify natural language responses of survey questionnaires into equivalent numeric codes according to the standard code book from the Korean National Statistics Office (KNSO). We implemented the system using a range of automated classification techniques, including hand-crafted rules, a maximum entropy model, and information retrieval techniques, to enhance the performance of automated industry/occupation coding task. The result is a Web-based AIOCS available for public services via the Web site of KNSO. Compared with the previous system developed in 2005, the new Web-based system decreases coding cost with a higher speed and shows significant performance enhancement in production rate and accuracy. Furthermore, it facilitates practical uses through an easy Web user interface.

Keywords: Automated Industry and Occupation Coding System, Web-based System, Classification, Hand-crafted Rule, Maximum Entropy.

1 Introduction

Industry and occupation information is important for national statistics in order to grasp life trends or distributions of people. In census surveys, data samples on industry and occupation are collected with specially designed multiple questionnaires. Based on the responses including “company name”, “business category”, “department”, “position”, and “job description”, each respondent is classified into one of the 442 industry categories and 450 occupation categories in South Korea. However, respondents are not supposed to choose an industry (or occupation) code directly, because they do not have enough background knowledge about the code hierarchy or the categories. As a result, the classification task had been done manually by experts before the development of automated coding systems. However, manual industry and occupation coding has two major problems. First, the task is very time-consuming and complicated when the coding data size is large. Second, the results fail in maintaining

J. Bailey et al. (Eds.): WISE 2008, LNCS 5175, pp. 443–457, 2008.

consistency when not well-trained coders or a number of coders who have different criteria are employed.

To solve these problems, there have been several approaches to computerized, automated coding systems since the early 1980s in U.S., France, Canada, Japan, and Korea [1], [2], [3], [4]. Although the implementation of the Automated Industry and Occupation Coding System (AIOCS) of each country follows its own code definition and hierarchy, the techniques used fall into hand-crafted rules, machine learning approaches, information retrieval methods, and their combinations.

To analyze public Census data efficiently, the Korea National Statistics Office (KNSO) has had grave concerns about the Automated Industry and Occupation Coding System for the Koreans (AIOCSK) for several years, and has built a hybrid classification system that combines a rule-based coding technique and an information retrieval technique [4]. However, it suffered from relatively low accuracy and production rate as a fully automated coding system and did not support public access because it ran on a UNIX shell environment that permits only registered internal users.

KNSO launched a new project to establish a highly stabilized, Web-based AIOCSK (i.e., AIOCSK 2008) that ensures high performance in terms of production rate and accuracy and low maintenance cost. For robust classifications over Industry and Occupation coding, it combines three different automated coding methods (hand-crafted rules, a maximum entropy model, and a set of information retrieval techniques) selectively. The system achieves almost 98% accuracy with 83% production rate in industry coding and 73% in occupation coding. In addition, AIOCSK 2008 supports basic management functionalities that can assist system administrators not only in handling several administrative tasks, but also in performance related tasks (e.g. updating hand-crafted rules and a domain-specific thesaurus, re-learning a maximum entropy model with the new training data, and re-building an index DB with refined training data).

With the newly developed Web-based AIOCSK, industry and occupation coding and an easy access control by KNSO Web site visitors will be available. Anyone can try industry and occupation coding with his or her own records, such as a company name or job descriptions. The system supports three different types of access rights: as an anonymous user, an authorized user, or an administrator. A one-stop and on-the-spot coding service is supported through the Web-based user interface. The coding status can be checked with a graphical progress bar. This will greatly improve the level of convenience for users, compared with a simple text display.

This paper is composed of related works (Section 2), a description of the industry and occupation code for the Koreans (Section 3), the system architecture (Section 4), our experiments (Section 5), our application (Section 6), the key payoffs (Section 7), and the conclusions (Section 8).

2 Related Works

Many government agencies around the world use rule engines, case-based reasoning, incremental decision trees, and others to assist their decision making ([5], [6]). Artificial intelligence (AI) techniques have been used to provide highly intelligent and accurate assessment capabilities and to assist a good number of automated services.

Compared with the rule-based method, machine learning methods do not require manual labors to create and/or maintain rules. There exist various kinds of machine learning techniques such as Decision Tree, Hidden Markov Model (HMM), Maximum Entropy Model (MEM), and Supporting Vector Machine (SVM).

Applying or combining those machine learning methods should consider the characteristics of the target data and the system resources available. Although SVM is superior to the other methods in accuracy in many tasks including document classification [7], we rejected it because of the problems of large memory and CPU time requirements when a computer is trained on large samples with more than 400 classes in our case. Among several machine learning techniques we considered in order to assign a code to a census record, we chose to use a maximum entropy model (MEM) based classifier and linguistically-motivated features to model the MEM based classifier because MEM offers a clear way to combine diverse pieces of contextual evidence (or clues) to estimate the probability of a certain linguistic class occurring in the context ([8], [9]) requiring reasonable memory consumption and processing time.

3 Overview of Code Hierarchy

The standard code book ([10], [11]) made by KNSO defines each classification code with a code name, a short description of the code, several examples, and exceptional cases. The descriptions mostly consist of "company name", "business category", "department", "position", and "job description". The classification codes are organized hierarchically into four levels as shown in Table 1, and there are 442 codes in the industry code book and 450 codes in the occupation code book.

Table 1. Structure of Industry (Occupation) Classification Codes (Excerpted from [4])

Code	Sections (level1)	Divisions (level2)	Groups (level3)	Classes (level4)
Digits	1	2	3	4
Type	alphabetic	numeric	numeric	Numeric
Example	A	01	012	0121
# of Industry (occupation) codes	17 (11)	63 (46)	194 (162)	442 (450)

4 System Architecture

The newly developed system is a part of an on-going project by the Information System Division at KNSO. It aims at offering common architecture for various kinds of automated coding systems, equipped with Artificial Intelligence (AI) techniques, which use prior domain knowledge, machine learning, and keyword-based search techniques, to respond efficiently to continuously required analysis of census data.

The system supports two key roles. One is Automated Industry and Occupation Coding (AIOC), performed three different automated coding modules (i.e., *Handcrafted rule based coding*, *Maximum entropy model based coding*, and *Information retrieval technique based coding*). Each coding module has its own customized

cut-off value (or threshold) [1] to achieve high performance in terms of production rate and error rate. We combined these different modules sequentially with applying the most trustable module first. More detailed coding steps are shown in Figure 1.

With the newly developed Web-based automated coding system, users can submit their data in a file or by manually entering the data. After submission, stepwise pre-processing is done to quickly prepare acceptable data formats needed by the three different automated coding modules. The system then starts automated industry (or occupation) coding adhering to the already scheduled procedure if the user selects HCR+MEM+IRT as a coding option (See Fig. 1). Finally, the user can check the coding results in the same page.

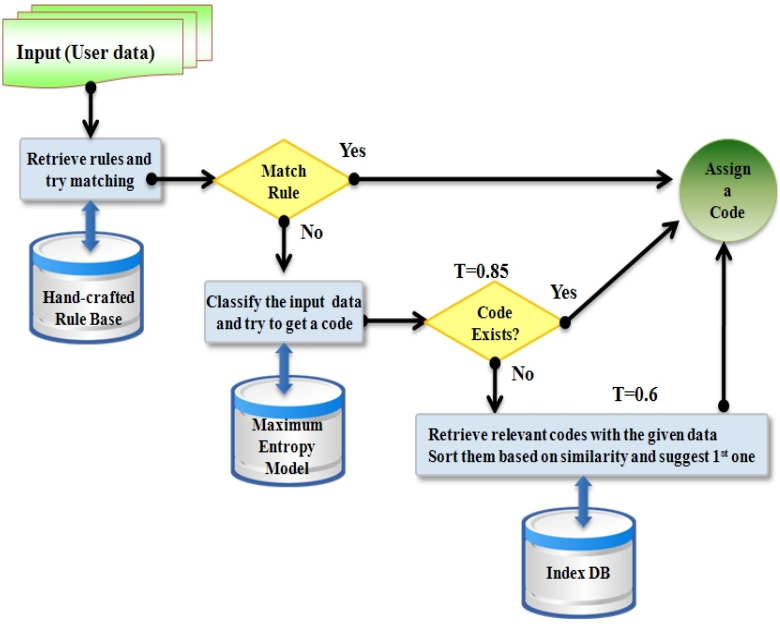


Fig. 1. General coding procedure

The other is the resource management functionality that includes insertion and deletion and modification of language resources, which are composed by a domain specific thesaurus and hand-crafted rules and the index data, related to industry and occupation codes as in Figure 2.

4.1 Hand-Crafted Rules (HCR)

The first rule-based coding was developed using a part of industry and occupation data of Census 2002 in 2005 and was manually improved several times until 2007. Rules for a code are prioritized, and every code has more than one rule in its rule If the highest priority rule does not satisfy the given input record, next priority rules are called sequentially (1^{st} priority rule→2^{nd} priority rule→ …). As in the following

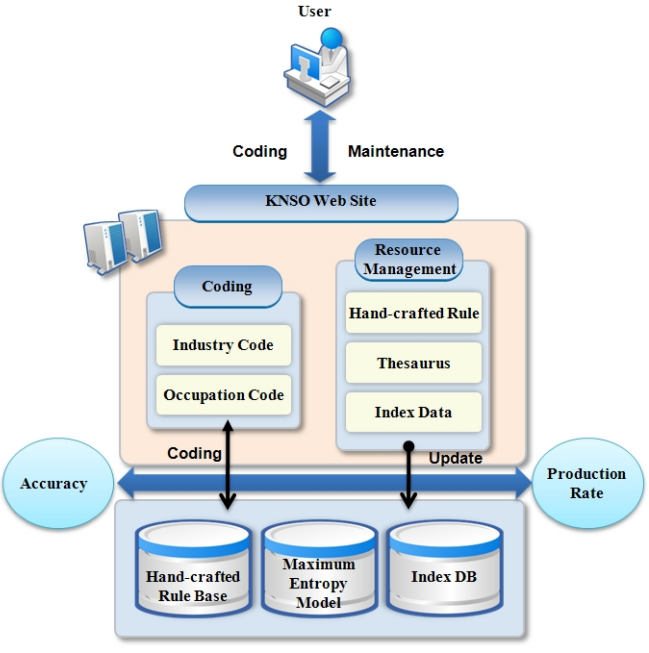


Fig. 2. Overall system framework

set.example, every record will be checked whether the discriminative words appeared in the given rule exist or not. If an appropriate rule is found, the industry (or occupation) code is assigned to the record. If no rule matches the input until the final rule of the rule set, "undetermined" is assigned.

Structure:
<company_name ^ business_category ^ department ^ position ^ job_description ^ exclusion_words> ➔ industry code, priority
Example:
<business_category = "벼농사(rice farming);벼재배(rice growing);쌀농사(rice farming);쌀재배(rice growing)", job_description = "농사(farming)", exclusion_words = "제조(making);제작(manufacture);조립(assembling);가공(processing);음식(food);식당(restaurant);..."> ➔ industry code=0111, priority=1 <company_name = "대학(college);대학(university);대학원(graduate school);전문대학(2-year college); 사관학교(military academy)", position="대학(college);교수(professor);시간강사(part-time instructor);조교(teaching assistant), exclusion_words="우체국(post office);병원(hospital);유치원(kindergarten);어린이집(nursery);초등학교(elementary school);중학교(middle school);고등학교(high school);..."> ➔industry code=8030, priority=0

Currently, there exist 5,386 rules for the industry coding and 3,800 rules for the occupation coding. Because those rules have been evaluated several times by domain experts of KNSO with previous Census data, we can trust coding quality by these rules. The high (approximately, 98~99%) accuracy are verified using Census 2005 data and newly collected human resource data.

However, in the rule-based approach, it is important to enrich both the rule set and its subordinate keyword set to keep good performance with future Census data. It is

necessary to make a constant effort to maintain the rules, because new words or expressions appear as new jobs are frequently created.

4.2 Maximum Entropy Model (MEM)

MEM utilizes contextual information (or evidence) in census data and can handle undiscovered or missed patterns automatically. Because domain experts can miss some patterns about all records, HCR may show low production rate when making hand-crafted rules. In order to overcome the pitfalls of HCR, MEM is selected, especially, as a classifier that guarantees stable production rate when accuracy is fixed at 98 or 99%.

The main idea of MEM is that the most uniform distribution among the probability distributions that satisfies the given constraints is ideal. The object of this modeling is to find *y* that maximizes the conditional probability *p(y|x),* which is expressed as *p*. In our model of MEM-based AIOCSK, *x* is the context of a sentence and *y* is the value of pre-defined slot (industry code or job code) in the sentence. Given *k*-feature constraints, the conditional probability is

$$\mathrm{p(y|x)} = \frac{1}{\mathrm{Z}} \exp\left(\sum_{\mathrm{k=1}}^{\mathrm{K}} \lambda_{\mathrm{k}} \mathrm{f_k(y,x)}\right)$$

where k is the number of features, f_k denotes the features, λ_{k} means the weighted parameters for features in the ME model and *Z* is a normalization factor to ensure that $\sum p(y|x) = 1$.

MEM has been applied to various NLP tasks such as sentence boundary detection, POS tagging, and parsing. Also, the ME classifier is good for integrating information from many heterogeneous information sources. For automated industry and occupation coding tasks that employ the MEM, we use company name, business category, department, position, and job description as the linguistic and contextual features to assign industry and occupation codes.

On the other hand, the variability of the terms and expressions among many respondents is also a serious problem: the same occupation can be described in many ways with many different terms, but the standard code description book ([10], [11]) contains a very small number of fixed terms. Experts obviously use their prior knowledge and experience to classify the descriptions into codes, in addition to the help of the coding guidelines provided. This fact also imposes a burden on MEM by requiring too big a feature space. It can result in memory overflow or system failure.

To alleviate the problem caused by the limited information available in the standard code book and to overcome the memory space problem, we decided to build a domain specific thesaurus in Korean, which had not been available previously. This method is to build a large-scale value (expression) mapping table by adding synonymous nouns and phrases that collected from the Census 2005 data. It includes a specially designed stepwise expression normalization procedure based on textual similarity computed by edit-distance [12] among expressions used in the census data.

The following example shows how synonymous expressions (or noisy expressions) frequently appearing in raw census data can be mapped with their normalized expressions.

Example:	
Normalized Expression [i.e., entry]	**Synonymous expressions (plus some broken expressions) [i.e., element]**
여행서비스 [Tour service]	여행서비스업무[Tour service affairs] 여행서비스제공[Supporting tour service affairs] 내국인관광알선여행서비스업[Tour service for domestic people] 운송업여행서비스 [Transport and tour services] 여행서비스업여행상품개발및여행객모집[Tour service including developing trip package and recruiting tourists] 관광여행서비스[Sightseeing & Tour services] 여행서비스업[Tour services] 여행서비스업항공권여권비자등[Tour services including plane reservation and visa] …
회계과 [Accounting department]	세무회계과[Tax & Accounting Department] 회계과경리계[Accounting Department] 경리계회계과[Accounting Department] 경영회계과[Management & Accounting Department] 기획행정국회계과[Accounting Department of Planning & Administrative Bureau] 재무회계과[Financial & Accounting Department] 금융회계과[Money & Accounting Department] ㅎ회계과[Accounting Department] PX회계과[Post Exchange Accounting Department] 회계과ㅣ[Accounting Department] 강남회계과[Accounting Department of Gangnam] 재경회계과[Accounting Department of Seoul] …

Table 2. Domain-specific Thesaurus

Field Name	Industry (entry/element)	Occupation (entry/element)
Company Name	9393 / 338446	6154 / 361102
Business Category	7412 / 679798	5805 / 748758
Department	5714 / 279720	5322 / 374907
Position	2090 / 108909	2336 / 171444
Job Description	6911 / 539309	7759 / 616939

Table 2 shows the statistics of the domain specific thesaurus. An entry slot and an element slot mean the number of distinct normalized expressions and the total sum of each entry's synonymous expressions. With the help of the thesaurus, we could make a manageable size of ME training model (about 30MB) by reducing the number of distinct expressions in each field. ME training model also contributed to enhancing the classification performance by identifying normalized representation of each feature and avoiding noisy expression (mostly broken or ill-formed words caused by spacing and spelling problems).

4.3 Information Retrieval Techniques (IRT)

Before the employment of MEM, IRT was used with HCR [4]. Since Korean is an agglutinative language and has quite unique linguistic characteristics, there were

many difficulties in adopting other countries' cases. IRT contributed to increasing production rate but produced unsatisfactory results in accuracy.

Our experiments show that the combination of HCR and MEM performs very well for the data including already known rules and patterns. However, there are some unknown cases where users' data are non-dominant and somewhat ambiguous. To handle these remaining cases, we added IRT as an automated coding option. It helped obtaining high production rate by sacrificing accuracy.

IRT utilizes the vector space model (VSM) based on term frequency (TF) and inverse document frequency (IDF) information [14]. Through indexing and weighting terms, each industry (or occupation) code is represented as a vector in the n-dimensional term space. When a record is coded (i.e. classified), it is also represented as a vector. The distance between the two vectors is calculated to compute similarity. A set of candidate codes are ranked by the cosine similarity in descending order, and the highest ranked code is assigned as a result of IRT. We used the Apache Lucene [15] as the core engine of IRT rather than re-implementing the information retrieval model to ensure cost-effectiveness in terms of speed and reliability.

5 Experiments

We evaluated our system with the Korean Census 2005 data (which is composed by about 2 million records), which have five fields: "company name", "business category", "department", "position", and "job description". The data were manually annotated with their industry/occupation codes by KNSO experts. All experiments were implemented in Java and tested on Microsoft Windows XP with Core2Duo 2.66GHz processor and 2.0GB of main memory. The cut-off value (or threshold) of MEM is the confidence score of Java MaxEnt [16], and that of IRT is the relevance score of the Apache Lucene. In our experiments, production rate $\left(\frac{\#\text{ of records that can generate a code}}{\#\text{ of inputted records}}\right)$, accuracy $\left(\left(\frac{\#\text{ of correctly assigned cases}}{\#\text{ of records that can generate a code}}\right)\right.$, and processing time were used as measurement factors.

5.1 Single Mode Experiment

Hand-crafted Rule (HCR) based Coding: Hand-crafted rule based coding shows persistently high accuracy, but the production rate is not satisfactory. Among 2 million records of Census 2005, HCR module generated industry codes about 1.18 million records. However, the accuracy of the generated codes amount to almost 98.8%. In terms of processing time, it takes about three hours on average because it recursively checks and evaluates several thousands of hand-crafted rules for each input record.

Maximum Entropy Model (MEM) based Coding: The performance of the MEM-based coding is close to 99% accuracy even though the production rate of industry and occupation coding was improved by 15% compared with that of HCR when we set the threshold at 0.85. The MEM part of Table 3 gives the result verified by 10-fold cross-evaluation. In terms of processing time, it takes only approximately 10 minutes when expression normalization time is ignored.

Information Retrieval Technique (IRT) based Coding: The IRT-based coding module using 5-field information has insufficient discrimination power to be a robust classifier, mainly due to potential ambiguities among frequently occurring keywords in the index. Thus, the production rate and the accuracy of IRT are not as high as those of HCR and MEM. In case of IRT, the accuracy of occupation coding hardly reaches 85% even if we adjust the cut-off values. Now, the production rate is lower than 30%.

Table 3. Experiments on Census 2005: HCR, MEM, and IRT

Type	HCR (No threshold)		MEM (Threshold=0.85)		IRT (Threshold=0.6)	
Category	Industry	Occupation	Industry	Occupation	Industry	Occupation
# of Records (A)	2031072	2031072	2020699	2023732	2020699	2023732
# of Production (B)	1181907	812118	1503499	1341851	565995	587193
# of Correct (C)	1168120	802108	1491851	1335575	485763	475140
Production Rate (B/A)	58.5%	40.1%	74.40%	66.31%	28.01%	29.02%
Accuracy (C/B)	98.83%	98.77%	99.23%	99.53%	85.82%	80.92%
Processing Time	188 minutes	188 minutes	10 minutes	11 minutes	60 minutes	63 minutes

5.2 Hybrid Mode Experiments

The purpose of hybrid-mode experiments is almost the same as that of single-mode experiments, but we tried several promising combinations using the three modules. We are going to skip a detailed analysis of HCR+IRT because the experiment on this combination did not show meaningful improvement in the production rate and the accuracy.

HCR+MEM based Coding: The following table and figure show that one of the best experiments, the combination of HCR and MEM, outperforms the MEM based coding module in both production rate and accuracy [Table 4]. It proves our assumption that MEM could enhance the production rate when we fixed the accuracy at 98%.

Table 4. Experiments on Census 2005: HCR + MEM

Type	HCR+MEM (Threshold=0.85)	
Category	Industry	Occupation
# of Records (A)	2031072	2031072
# of Production (B)	1699980	1480668
# of Correct (C)	1678693	1457117
Production Rate (B/A)	83.70%	72.90%
Accuracy (C/B)	98.75%	98.41%
Processing Time	193 minutes	194 minutes

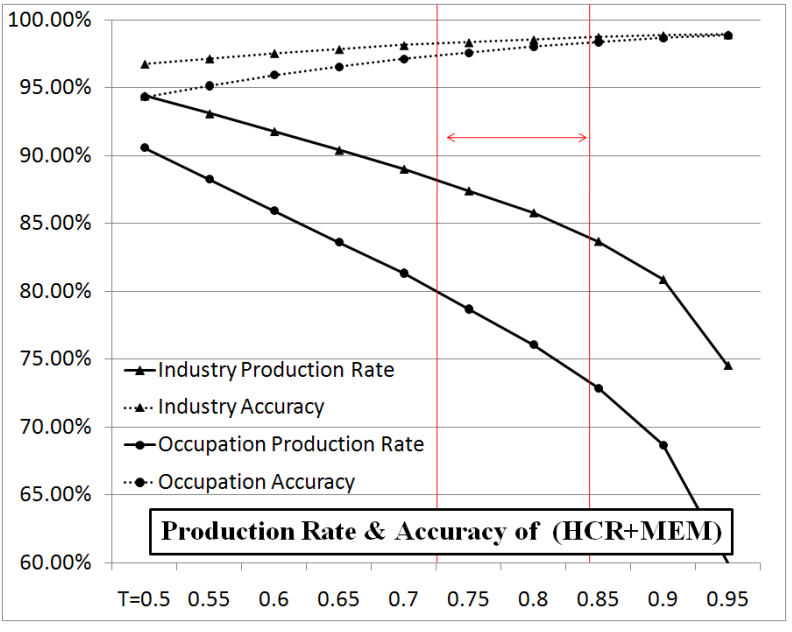


Fig. 3. Distribution of production rate and accuaracy for (HCR+MEM) when thersholds are varied

When we adjusted the threshold of MEM, the production rate and the accuracy changed as in Figure 3. Based on our experiments, the optimal thresholds for the combination of HCR+MEM were identified from 0.75 to 0.85. If we consider that the Census 2005 data themselves may have more than 10% potential errors arisen from the manual annotation process, the performance of HCR+MEM is quite impressive.

HCR+MEM+IRT based Coding: The combination of the three different automated coding modules did not outperform the performance of HCR+MEM [Table 5, 6]. Patterns discovered in the hand-crafted rules and the classification model driven by MEM handled approximately 80% of industry coding and 70% of occupation coding

Table 5. Step-wise analysis of Performance: Industry

Type	HCR (No threshold)	MEM (Threshold=0.85)	IRT (Threshold=0.6)	Accumulated Total
# of Records (A)	2031072	832212	288203	2031072
# of Production (B)	1181907	544009	56098	1798967
# of Correct (C)	1168120	533056	28278	1746167
Production Rate (B/A)	58.5%	65.38%	19.46%	88.57%
Accuracy (C/B)	98.83%	97.99%	50.41%	97.06%
Processing Time	188 min	5 min	5 min	198 min

Table 6. Step-wise analysis of Performance: Occupation

Type	HCR (No threshold)	MEM (Threshold=0.85)	IRT (Threshold=0.6)	Accumulated Total
# of Records (A)	2031072	1213274	485698	2031072
# of Production (B)	812118	727576	86691	1632065
# of Correct (C)	802108	707614	42828	1558099
Production Rate (B/A)	40.1%	59.97%	17.85%	80.35%
Accuracy (C/B)	98.77%	97.26%	50.56%	95.53%
Processing Time	188 min	6 min	9 min	203 min

with 98% accuracy. However, when we applied IRT after HCR+MEM, we observed that IRT did not discriminate effectively the remaining records in terms of production rate and accuracy.

Comparison among Different Combinations: The effectiveness levels that our hybrid techniques (especially, HCR+MEM) have achieved in the above experiments are quite stable and satisfactory. Although AIOCSK 2005 was successfully used as a

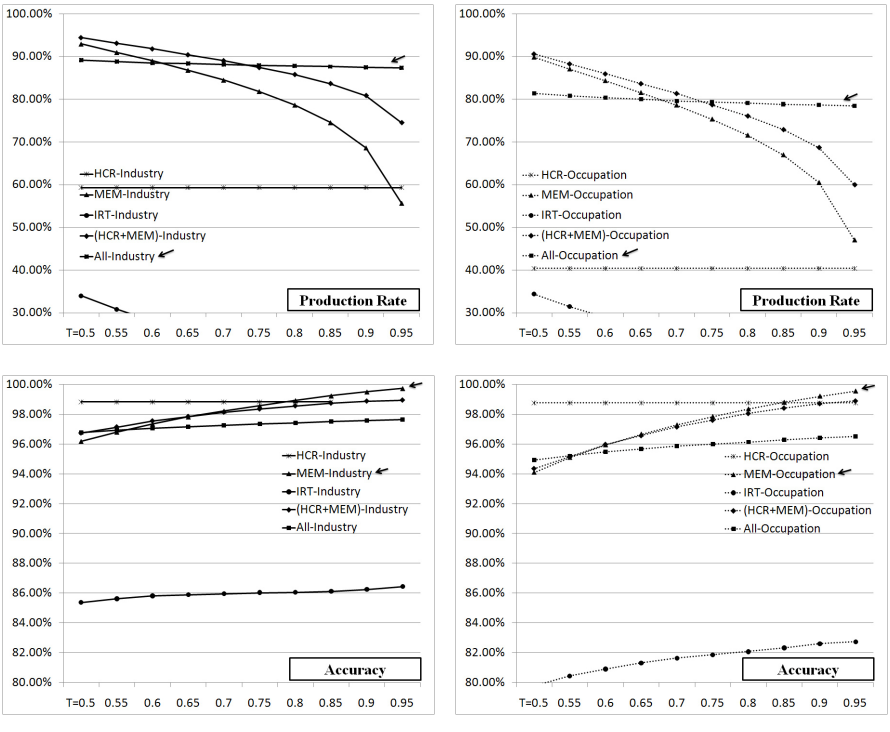


Fig. 4. Production rate vs. Accuracy

computer-clerical system that generates several candidate codes in a fixed size of candidate set [4], AIOCSK 2008 shows more satisfactory performance as a fully automated system that assigns the best code without any residual code. Actually, AIOCSK 2005 (which combines HCR and IRT) performed slightly better than HCR in every case, that is, it helped increase the production rate. It suggested the second best code and third best code according to term similarity, but the accuracy was maintained at lower than 98%. As shown in Figure 4, there is a trade-off relationship between production rate and accuracy. Thus, maintaining a certain level of performance in industry and occupation coding is very important when we combine different types of classification modules. If the user wants to adhere to high accuracy, MEM is preferred (See the lower part of Fig. 4). However, when the goal is highest production rate, HCR+MEM+IRT will be a good choice (See the upper part of Fig. 4). As mentioned before, our experiments conclude that HCR+MEM is the best combination in maintaining the optimal level of production rate and accuracy in industry and occupation coding.

6 Application

The first version of AIOCSK system applying hand-crafted rule and information retrieval technique was released in mid-January 2005 just for internal usage (i.e., AIOCSK 2005). This was followed by the development of new AIOCSK in mid-Dec. 2007. Employment of maximum entropy model based automated coding module and a domain-specific thesaurus are distinguishing characteristics compared with the AIOCSK 2005.

User testing began in Oct. 2007 with the first rollout to the official launch which is scheduled as June 2008. Before the public release, the system has been undergoing extensive testing until now. In parallel, the each automated coding modules will be optimized with fine-tuning features and more stable performances.

The Web platform for the AIOCSK 2008 is Java Server Page (JSP). Most core modules are therefore also Java-based and packaged and deployed as Java jar files. The front-end to AIOCSK 2008 is a Web-based thin client operated by KNSO. The layout and design of the Web client is typical form-based systems. For each coding option (HCR only, MEM only, IRT only, HCR+MEM, and HCR+MEM+IRT) provided, there are several radio buttons that allow a user to select a coding option that he or she wants (See Fig. 5). Basically, everything related to automated coding (including required linguistic and computational resources) is all consolidated in back of an integrated UI for the users. Figure 5 shows a screen shot of the Web-based AIOCSK when accessed as an anonymous user.

Although users can access AIOCSK 2008 as anonymous users, more detailed functions such as large-scale coding and system resource management require higher level of access rights. A series of coding procedure operates in near real-time within a few seconds to assign a code. Other maintenance-related modules that are not performance-critical, such as updating index DB or domain-specific thesaurus, are done behind the scene as asynchronous processes.

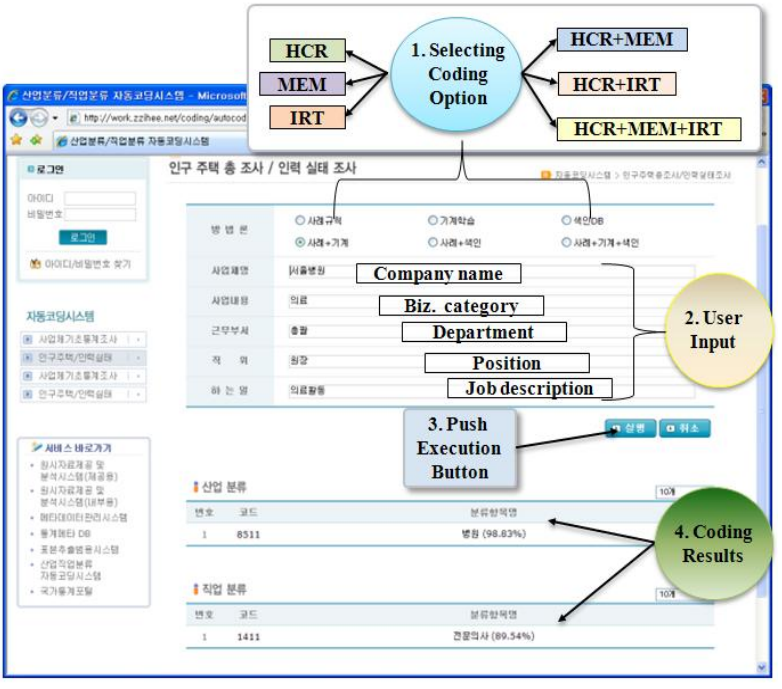


Fig. 5. Web-based automated coding screen shot

7 Key Payoffs

Quality of Services: The hybrid approach enhances the quality of automated coding. Employing MEM achieves more discriminative, powerful automated coding with a higher production ratio and a lower error rate than AIOCSK 2005. With the Web-based AIOCSK, one-stop service is available, regardless of the access location as long as Internet connection is available. It provides immediate coding using basic functions as shown Figure 5. In addition, the graphical progress bar that represents a progress status in terms of the percent of the given coding task allows users to recognize the completion ratio.

Increased Productivity: With the newly updated AIOCSK 2008, the most relevant code can be assigned without further effort by a KNSO expert or additional guidelines. After users' uploading the target file or entering the data manually, just their clicking on the "Execute Coding" button will show the results of automated coding. Table 4 and Figure 3 show the enhanced productivity of AIOCSK 2008 when compared with other coding modules and their combinations. It shows 83% in industry production rate and 73% in occupation production rate when we fix their accuracy at 98%.

Improved Agility: The new system's tuning is parameter-driven: for example, changing in the cut-off values (i.e., thresholds) or the number of recommendation codes, and any other urgent changes in policies can be made promptly without any changes in the underlying software modules. With the help of the fully automated architecture that permits user's further modifications, the system adapts itself, reflecting recent changes in rules, thesaurus, and index data from domain experts and administrators.

Maintenance: Just like any other mission critical software, there will inevitable be changes and upgrades to modules in AIOCS 2008 after deployment to reflect legislative or functional changes followed by KNSO. The design of the self-evolving framework is such that these types of change are very easy to deal with.

First, all linguistic resource related changes can be done without any java coding and simply by updating thesauri and clicking "update" button. Then, embedded modules trigger required procedures correspondingly.

Second, in case of parameter tuning, needed behaviors of user is selecting a threshold value as an automated coding constraints (e.g. T=0.85). Packaging the automated coding modules as a decoupled component from the other parts of the system helps reducing cost of supplementary maintenance and integration.

Third, by adhering object oriented program (OOP) paradigm, all source codes written in Java class can be easily adapted and modified on demand.

For support, the prime development team of KNSO provides front-line technical and end-user support while we provide additional assistance on the automated coding technologies and algorithm changes when needed.

8 Conclusion

This paper describes a successful realistic case of developing a high-performance Automated Industry and Occupation Coding System for the Koreans (AIOCSK) that combines different types of automated classification techniques (hand-crafted rules, maximum entropy model, and information retrieval technique). This is most likely the first serious attempt in the world that utilizes the chain of artificial intelligence methodologies in automated industry and occupation coding and starts public services on a web environment.

The hybrid approach of Web-based AIOCSK results in delivering higher quality and faster services to users, including both novice users and experts of KNSO. Our experiments verified the fact that the combination of hand-crafted rules and maximum entropy model achieves 83% in industry production rate and 73% in occupation production rate when we fix their accuracy at 98%.

Although the results are promising, we think that further research is necessary in a number of directions for improvements.

- **Self-evolving framework:** It is clear to upgrade the existing system without a drastic change in the near future.
- **Weighted voting based on content-analysis:** Currently, different types of classification methods are combined almost sequentially, but we could expect a more synergistic combination of different classifiers if the system could identify a content-preferred classifier according to inputted records.

- **Semantic value mapping to domain-specific thesaurus:** Generating a domain-specific thesaurus based on textual similarity was quite useful for dealing with synonyms having syntactic similarity and broken words, but we need to develop a more advanced value mapping algorithm to cover semanticsimilarity.

Acknowledgment

The Census2005 data used in this work were collected by the Korea National Statistics Office (KNSO). We are grateful to Information System Development Division of KNSO for providing the corpus and for assisting us in their interpretation.

References

1. Chen, B., Creecy, R.H., et al.: On Error Control of Automated Industry and Occupation Coding. Journal of Official Statistics 9(5), 729–745 (1993)
2. Takahashi, K.: A Supporting System for Coding of the Answers from an Open-ended Question: An Automatic Coding System for SSM Occupation Data by Case Frame. Sociological Theory and Methods 15(1), 149–164 (2000)
3. Takahashi, K., Takamura, H., Okumura, M.: Automatic Occupation Coding with Combination of Machine Learning and Hand-Crafted Rules. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 269–279. Springer, Heidelberg (2005)
4. Lim, H.S., Lee, W.K.H., et al.: An Automatic Code Classification System by Using Memory-Based Learning and Information Retrieval Technique. In: Lee, G.G., Yamada, A., Meng, H., Myaeng, S.-H. (eds.) AIRS 2005. LNCS, vol. 3689, pp. 577–582. Springer, Heidelberg (2005)
5. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann, San Mateo (1993)
6. Mitchell, T.: Decision Tree Learning. In: Mitchell, T. (ed.) Machine Learning, pp. 52–78. McGraw-Hill, New York (1997)
7. Vapnik, V.: Statistical Learning Theory. John Wiley, New York (1998)
8. Ratnaparkhi: A Maximum Entropy Model for Part-of-speech Tagging. In: Proc. of the Empirical Methods in Natural Language Processing, pp. 133–142 (1996)
9. Ratnaparkhi: A Simple Introduction to Maximum Entropy Models for Natural Language Processing, Technical Report 97-08, Institute for Research in Cognitive Science, Univ. of Pennsylvania (1997)
10. Korean Standard Industry Classification, Korea National Statistics Office (2000)
11. Korean Standard Occupation Classification, Korea National Statistics Office (2000)
12. Vilares, M., Ribadas, F.J., Vilares, J.: Phrase Similarity through the Edit Distance. In: Proc. of Database and Expert Systems Applications 2004. LNCS, vol. 31080, pp. 306–317. Springer, Heidelberg (2004)
13. Melz, R., Ryu, P.-M., Choi, K.-S.: Compiling large language resources using lexical similarity metrics for domain taxonomy learning. In: 5th Int. Conf. on Language Resources and Evaluation (2006)
14. Baeza-Yates, R., Ribeiro, B.: Modern Information Retrieval. Addison-Wesley, Reading (1998)
15. An Indexing Engine, Apache Lucene, `http://lucene.apache.org/`
16. Java package for training and using maximum entropy models, OpenNLP MaxEnt, `http://maxent.sourceforge.net/`

Author Index