SIGMOD Record – June 2005 – Volume 34 Issue 2 – www.sigmod.org/record

SIGMOD OFFICERS, COMMITTEES AND AWARDS 1
EDITOR'S NOTES 2
CHAIR'S MESSAGE 3

INVITED ARTICLE
A Tribute to Professor Hongjun Lu 5
M. J. Carey and J. Han

RESEARCH ARTICLES AND SURVEYS
Peer-to-Peer Management of XML Data: Issues and Research Challenges 6
G. Koloniari and E. Pitoura
Mining Data Streams: A Review 18
M. M. Gaber, A. Zaslavsky and S. Krishnaswamy
Analytical Processing of XML Documents: Opportunities and Challenges 27
R. R. Bordawekar and C. A. Lang
On Six Degrees of Separation in DBLP-DB and More 33
E. Elmacioglu and D. Lee
Semantic Characterizations of Navigational XPath 41
M. Marx and M. de Rijke
Nested Intervals Tree Encoding in SQL 47
V. Tropashko

RESEARCH CENTERS
The Indiana Center for Database Systems at Purdue University 53
M. Ouzzani et al.

EVENT REPORTS
Report on the Ninth Conference on Software Engineering and Databases (JISBD 2004) 59
J. Hernández, E. Pimentel and A. Toval
Report on the 1st International Symposium on the Applications of Constraint Databases (CDB'04) 62
B. Kuijpers and Peter Revesz
Report on the International Workshop on Pattern Representation and Management (PaRMa'04) 65
Y. Theodoridis and P. Vassiliadis

REMINISCENCES ON INFLUENTIAL PAPERS
K. A. Ross, editor 68

DISTINGUISHED DATABASE PROFILES
Bruce Lindsay Speaks Out 71
M. Winslett

INDUSTRY PERSPECTIVES
Building Data Mining Solutions with OLE DB for DM and XML for Analysis 80
Z. Tang, J. Maclennan and P. P. Kim

DATABASE PRINCIPLES
Tools for Composite Web Services: A Short Overview 86
R. Hull and J. Su

[**Editor's note**: With the exception of the last pages –which would be the back cover of the printed issue– that are not included in this file, it has the same contents as the printed edition. All the articles are also available individually online and have been put together here for convenience only.]

SIGMOD Record

SIGMOD Record is a quarterly publication of the Special Interest Group on Management of Data (SIGMOD) of the Association for Computing Machinery (ACM). SIGMOD is dedicated to the study, development, and application of database and information technology. *SIGMOD Record Web Edition* is also freely available online at `http://www.acm.org/sigmod/record`.

SIGMOD Record solicits contributions of articles, technical notes, reports, and proposals for special sections. Conference announcements and calls for papers are published if relevant to the interests of the group and, in most cases, are limited to one page. All contributions should be sent to the editor for consideration. Submitted technical papers are reviewed for importance and correctness. Priority is given to papers that deal with current issues of interest to a broad audience. Papers should be submitted electronically in POSTSCRIPT, PDF or Microsoft Word format to `record@sigmod.acm.org`, and they should follow a format similar to that of the SIGMOD conference proceedings (but with a larger font): 10 point font, single-space, 2-column, 8.5" by 11" page size with 1" margins all around and no page numbers. Submitted articles are limited to 6 pages unless prior agreement of the editor is obtained.

Mario A. Nascimento, *SIGMOD Record* Editor.
`record@sigmod.acm.org`
Dept. of Computing Science, University of Alberta
Edmonton, AB, Canada

SIGMOD Record (ISSN 0163-5808) is published quarterly by the Association for Computing Machinery, Inc., 1515 Broadway, New York, NY 10036. Periodicals postage paid at New York, NY 10001, and at additional mailing offices. POSTMASTER: Send address changes to *SIGMOD Record*, ACM, 1515 Broadway, New York, NY 10036.

Visit the SIGMOD Online website at `http://www.acm.org/sigmod`

SIGMOD Officers, Committees, and Awardees

Chair
Tamer Ozsu
University of Waterloo
Waterloo, Ontario
Canada N2L 3G1
(519) 888-4567
tozsu@db.uwaterloo.ca

Vice-Chair
Marianne Winslett
University of Illinois
1304 West Springfield Avenue
Urbana, IL 61801 USA
(217) 333-3536
winslett@cs.uiuc.edu

Secretary/Treasurer
Joachim Hammer
University of Florida
Gainesville
FL 32611-6125, USA
(352) 392-2687
jhammer@cise.ufl.edu

Information Director: Alexandros Labrinidis, University of Pittsburgh, labrinid@cs.pitt.edu.

Associate Information Directors: Manfred Jeusfeld, Dongwon Lee, Michael Ley, Alberto Mendelzon, Frank Neven, Altigran Soares da Silva, Jun Yang.

Advisory Board: Richard Snodgrass (Chair), University of Arizona, rts@cs.arizona.edu, H.V. Jagadish, John Mylopoulos, David DeWitt, Jim Gray, Hank Korth, Mike Franklin, Patrick Valduriez, Timos Sellis, S. Sudarshan.

SIGMOD Conference Coordinator: Jianwen Su, UC Santa Barbara, su@cs.ucsb.edu

SIGMOD Workshops Coordinator: Laurent Amsaleg, IRISIA Lab, Laurent.Amsaleg@irisa.fr

Industrial Advisory Board: Daniel Barbará (Chair), George Mason Univ., dbarbara@isse.gmu.edu, José Blakeley, Paul G. Brown, Umeshwar Dayal, Mark Graves, Ashish Gupta, Hank Korth, Nelson M. Mattos, Marie-Anne Neimat, Douglas Voss.

SIGMOD Record Editorial Board: Mario A. Nascimento (Editor), University of Alberta, mn@cs.ualberta.ca, José Blakeley, Ugur Cetintemel, Brian Cooper, Andrew Eisenberg, Leonid Libkin, Alexandros Labrinidis, Jim Melton, Len Seligman, Jignesh Patel, Ken Ross, Marianne Winslett.

SIGMOD Anthology Editorial Board: Curtis Dyreson (Editor), Washington State, cdyreson@eecs.wsu.edu, Nick Kline, Joseph Albert, Stefano Ceri, David Lomet.

SIGMOD DiSC Editorial Board: Shahram Ghandeharizadeh (Editor), USC, shahram@pollux.usc.edu, A. Ailamaki, W. Aref, V. Atluri, R. Barga, K. Boehm, K.S. Candan, Z. Chen, B. Cooper, J. Eder, V. Ganti, J. Goldstein, G. Golovchinsky, Z. Ives, H-A. Jacobsen, V. Kalogeraki, S.H. Kim, L.V.S. Lakshmanan, D. Lopresti, M. Mattoso, S. Mehrotra, R. Miller, B. Moon, V. Oria, G. Ozsoyoglu, J. Pei, A. Picariello, F. Sadri, J. Shanmugasundaram, J. Srivastava, K. Tanaka, W. Tavanapong, V. Tsotras, M. Zaki, R. Zimmermann.

SIGMOD Digital Review Editorial Board: H. V. Jagadish (Editor), Univ. of Michigan, jag@eecs.umich.edu, Alon Halevy, Michael Ley, Yannis Papakonstantinou, Nandit Soparkar.

Sister Society Liaisons: Stefano Ceri (VLDB Foundation and EDBT Endowment), Hongjun Lu (SIGKDD and CCFDBS), Z. Meral Özsoyoğlu (IEEE TCDE), Serge Abiteboul (PODS and ICDT Council).

Latin American Liaison Commitee: Claudia M. Bauzer Medeiros (Chair), University of Campinas, cmbm@ic.unicamp.br Alfonso Aguirre, Leopoldo Bertossi, Alberto Laender, Sergio Lifschitz, Marta Mattoso, Gustavo Rossi.

Awards Committee: Moshe Y. Vardi (Chair), Rice University, vardi@cs.rice.edu. Rudolf Bayer, Masaru Kitsuregawa, Z. Meral Ozsoyoglu, Pat Selinger, Michael Stonebraker.

Award Recipients:

Innovation Award: Michael Stonebraker, Jim Gray, Philip Bernstein, David DeWitt, C. Mohan, David Maier, Serge Abiteboul, Hector Garcia-Molina, Rakesh Agrawal, Rudolf Bayer, Patricia Selinger, Don Chamberlin, Ronald Fagin.

Contributions Award: Maria Zemankova, Gio Wiederhold, Yahiko Kambayashi, Jeffrey Ullman, Avi Silberschatz, Won Kim, Raghu Ramakrishnan, Laura Haas, Michael Carey, Daniel Rosenkrantz, Richard Snodgrass, Michael Ley, Surajit Chaudhuri.

Editor's Notes

Dear Colleagues,

I have recently received a few inquiries regarding publication/turnaround time. I would like to have around 6 weeks as the turnaround time, i.e., time between original submission, review, decision notification and receiving the final camera-ready version. Sometimes this is feasible but some, perhaps I should say most, times it is not. One non-trivial issue that hinders this is the so-called "reviewer fatigue." This is not a new issue, and unfortunately I see little that can be done to alleviate this in the short term. As a consequence sometimes I have a hard time finding a good reviewer for submitted papers, which obviously adds to the turnaround time. As well, since the *Record*'s Associate Editors have their own material to handle it is left to me to find suitable reviewers, which sometimes requires some research from my side. I am considering adopting some policy that would require authors to suggest 2-4 names of suitable (and perhaps also of undesired) reviewers (subject to usual conflict of policy rules, of course). Let me also remind you that I must work with a page budget that also constrains when accepted papers will be actually published. In fact, in order to save some space I will break from the tradition of writing a sentence or two about the articles featured in the issues, a page here and there may eventually allow for an extra paper.

Along these lines, it may be also time for the *Record* to have a larger and broader set of policies in place. For instance, some of the columns should have some kind of policy, set in particular the columns on Event Reports and on Research Centers. Even whether some columns should be discontinued and/or others be introduced is a question not easy to answer. I am hoping to soon have some sort of online mechanism where you can provide some feedback regarding these topics. In the meantime I will maintain the current status-quo. Changes can be good but have to be well thought out before implemented. The *Record* is considered among the ACM SIGs to be a good quality, current and regularly published newsletter and it is very important to us all, the SIGMOD community, that we keep this reputation.

In summary, I am considering and discussing ideas which aim at improving and maintaining the quality of the papers published in the *Record*, and therefore of the *Record* itself, as well as assure timely dissemination. I hope to have a set of policies (or guidelines) published in the next edition. In the meantime, ideas from you, the readers and potential contributors, in the two main issues above will be most welcome.

Recently we have had a change in the list of Associate Editors. Amit Sheth, who served for quite some time as the Associate Editor for Research Centers has retired from that position. On behalf of SIGMOD's Executive Committee and past *Record* editors I want to extend our sincere thanks to Amit for his continuous help. To replace him I have invited Ugur Cetintemel from Brown University. I am confident he will continue Amit's good work. Welcome aboard Ugur!

When the printed issue reaches you we will probably have a new Chair, Vice-chair and Treasurer elected. I just want to take this chance to, on behalf of the Associate Editors and myself, thank the former officers, Tamer, Marianne, and Joachim for the opportunity they have given us, as well as for their guidance and help. Likewise, to the new officers, we offer our best wishes and our continuous help with the *Record*.

Mario Nascimento, Editor.
May 2005

Chair's Message

This is my last Chair's Message; as you read this, my term as SIGMOD Chair, along with the terms of Marianne Winslett (Vice-Chair) and Joachim Hammer (Secretary/Treasurer), is coming to a close. In a few weeks, we will know the results of elections and who will lead our organization for the next four years. The candidate slate is strong, and I am convinced that the organization will be very well served whoever are the new slate of officers.

Since this is the last time I communicate with you in this capacity, I thought I would provide a short report on what we accomplished in the past four years and where we failed. The following are some of the issues that we worked on:

- One of the major initiatives we worked on was the "internationalization" of SIGMOD and its activities. This was on my and Marianne's candidacy platforms, and I had raised it as a major issue in my first Chair's message. We have made significant progress on this front with the location of our flagship conference in locales outside of North America for the first time in its 30 year history. We had SIGMOD/PODS 2004 in Paris, France and we will have SIGMOD/PODS 2007 in Beijing, China. We hope to continue this trend in future years. Along these lines, we made progress in ensuring that there was better geographic representation on various SIGMOD bodies. In particular, both candidates for the Vice-Chair position this year are from outside North America. We need to continually work on expanding our membership base in all countries where there are data management research and development activities and ensure that these colleagues contribute to the running of our organization.

- SIGMOD is not a very large organization in terms of its membership size, but it is a very active one with many activities. To better cope with this level of activity and to lay the framework for future growth, we spent considerable time and energy to put in place an administrative structure that is more distributed and more "cabinet-like". Consequently, we created some new positions (e.g., Conference Coordinator) and we enlarged the Executive Committee to include all of the major volunteers. The Executive Committee, with the advice of the Advisory Board, now makes all the major decisions regarding SIGMOD. In addition to our face-to-face meetings, we hold monthly conference calls that allow us to address issues in a timely fashion. We updated the Bylaws to reflect these changes and these changes are now in the ratification process by membership.

- One of the joys of my position is to run the awards ceremony at our annual SIGMOD/PODS conference; this is a highlight event where we recognize those among us who have excelled in various activities. They also bring recognition to SIGMOD within the ACM community and beyond. We have renamed our Innovations Award in honor of Ted Codd who passed away in 2003. The new name "SIGMOD Edgar F. Codd Innovations Award" was approved by ACM Council and the first renamed award was given last year.

  This year we are starting a new award, SIGMOD Doctoral Dissertation Award, which, as its name implies, will be given to one dissertation each year, starting in 2006. This award is supported by Microsoft Research (thanks to Jim Gray) and we are in the process of forming the selection committee.

- We started to collaborate with other database organizations. Two activities are particularly important to note. The first is the SIGMOD-VLDB Library Program where we collaborate

with VLDB Endowment in distributing 1000 SIGMOD Anthology Silver Edition DVDs to institutions in developing countries. To date, we have distributed about 400 copies. The second is our recent collaboration with VLDB Endowment in handling papers submitted to the annual SIGMOD conference and the VLDB conference. As I reported in earlier columns, the authors of some of these papers will be given the option of revising their papers and submitting the revised version for re-consideration by the original reviewers, but for presentation at the next conference. The hope is that this will reduce the load on each conference PC and will reduce, to a certain extent, the chance-effect of conference reviews. We will include ICDE in the mix as soon as the process gets smoothed out.

- Working on and stabilizing SIGMOD budget was a major activity over my tenure. SIGMOD's many activities had drained our reserves and the membership fees had not been adjusted in over ten years, causing significant pressure on our budget. ACM's changing (read increasing) charges and fund balance requirements only added to this pressure. We addressed the issue in a number of ways. We went over each of our activities and decided to scale back or cut those which are not within our core services; we instituted a "pay-as-you-go" system whereby any new activity had to be accompanied by a budget analysis and reasonable assurance that the required funds were available; and we conducted a study to determine our direct membership costs and adjusted the membership fee to cover our direct costs. Right now the budget is stable, thanks in part to the ACM Digital Library income (which attests to the wisdom of digitizing most of SIGMOD material by Rick Snodgrass's administration), but expenses will need to be carefully watched.

- Membership is one area that we were not able to address adequately. Following the adjustment of the membership fee, we are now in a good position to increase membership, but we were not able to launch a major membership drive. This is one task that we will have to hand over to the next administration. Our current membership is stable around 2,800, but I believe there is room for growth, in particular outside of North America and among graduate students.

I'd like to end with some personal thanks. I have been a member of SIGMOD since 1975 or 76 (I don't have the exact date and ACM's records are not helpful) when I was only a master's student and this has always been my professional home. It was a great honor to be asked to run for Chair four years ago and it was a privilege to serve; I have greatly enjoyed the experience (well, most of it!) and appreciate having had the chance to put some of my ideas into action. SIGMOD's work during these past years were shouldered by a great many colleagues who volunteered their time, sometimes more than they were led to believe. I will not name all of them, their names can be found on the inside covers of current and past SIGMOD Record issues. I thank them all for their work. I want to thank Marianne and Joachim for being tremendous partners, and Rick for his continuous help. Most importantly, I thank you for the opportunity to serve in this capacity and for your support; I am sure you will extend the same support to our next set of officers.

M. Tamer Özsu
May, 2005

A Tribute to Professor Hongjun Lu

Michael J. Carey Jiawei Han

Dr. Hongjun Lu, Professor of Computer Science, Hong Kong University of Science and Technology, lost his brave fight against cancer and left us in the evening of March 3, 2005. The world lost a dedicated and brilliant computer scientist. The database research community lost a respected and prolific researcher, an effortless organizer and promoter of database research in the world, a friend cherished by many colleagues and researchers, and a wonderful teacher who deeply affected and was revered by his students.

In response to this sad news, messages of condolence flooded into the Hong Kong University of Science and Technology from all over the world. Hundreds of memorial messages are posted on the HKUST website paying tribute to him, and hundreds of wreaths and flowers lined his memorial service. Reading those touching words from all over the world, from Asia, Australia, North America, and Europe, one can clearly feel the impact of Hongjun's departure. His dedication and his contributions to our community will live forever in our hearts.

Hongjun received his B.Sc. from Tsinghua University in China, obtaining his M.Sc. and Ph.D. degrees from the University of Wisconsin-Madison. After a short span of successful research at Honeywell, Hongjun moved into academia, taking a faculty position at the National University of Singapore (NUS). His twelve years of dedication at NUS led to a "sea change" for database research in Singapore, as NUS today is a major stronghold in database research. In 1998, Hongjun joined the faculty at the Hong Kong University of Science and Technology. There he set out to organize a strong database research thrust in Hong Kong and China, and he contributed greatly to raising the research level in the region and attracting major international database conferences there.

Hongjun was an internationally recognized database researcher. He leaves a tremendous legacy with his seminal work on data and knowledge base management systems, query processing and optimization, physical database design, database performance, data warehousing, and data mining. His query processing and physical design research examined join processing, indexing and query evaluation for object-oriented and XML databases, parallel query processing, transitive closure, grid query processing, and multi-query optimization. His research on knowledge base management and data mining included work on classification using neural networks, association rule mining, data cube computation methods, text classification, frequent pattern mining, and learning-based web query processing. Most recently, he had started to work in the emerging area of stream data processing. His prolific career spanned 20 years—his work as a graduate student appeared in ICDCS'85, VLDB'85, ICDE'86 (which won the best student paper award), and SIGMOD'86, and this past year, his technical productivity was still as high as ever, leading to an ICDE'04 paper, a SIGMOD'04 paper, three VLDB'04 papers, and two ICDE'05 papers. In all, Hongjun co-authored over 200 publications in scientific journals, conferences, and workshops during his 20 years in the field. In addition, he gave many tutorials, invited talks, and keynote talks in international and regional database-related conferences. His level of contribution is remarkable, even more so given that he spent the bulk of his career working in the Far East, away from the usual geographic centers of database research.

Besides his research, Hongjun devoted a great deal of time and energy to selflessly serving the database community. He served as a member of the ACM SIGMOD Advisory Board from 1998 to 2002. He became a Trustee of the VLDB Endowment in 2000 and was its secretary at the time of his death. He was an Editor for IEEE Transactions on Knowledge and Data Engineering, Knowledge and Information Systems: An international Journal, and a Springer-Verlag book series on Database Management and Information Retrieval. He served on program committees and organizing committees for all of the major international database conferences and workshops, including the SIGMOD, VLDB, ICDE, SIGKDD, and ER conferences. In addition, he was the founder and the Steering Committee chair for two regional conferences in the Far East, the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD, which began in 1997) and the International Conference on Web-Age Information Management (WAIM, which started in 2000).

Hongjun worked hard to raise the level and visibility of academic database research in the Asia-Pacific region. This began when he was a young faculty member at NUS, where he helped to build an internationally known database research group. He also worked to promote quality database research in mainland China, helping to cultivate internationally active database groups there and to coach young Chinese faculty and students to conduct quality research and to publish at quality database conferences. Hongjun was a bridge between the international and mainland Chinese database communities. He served as a liaison for ACM SIGMOD to the Database Society of the China Computer Federation, recruiting ACM SIGMOD members from China, and he was an organizer of the VLDB Database School (China), starting in 2002, which was set up to train young faculty members and senior research students in the field.

With his devotion to family, friends, research, and teaching, Hongjun leaves behind a vast number of grieving people on four continents whose lives were touched and improved by his presence. He was a role model for many of us over the years, and he will live on in the memories of those who knew him. We will always remember Hongjun's intelligence, his enthusiasm, his smile, his kind-heartedness, and his unselfish dedication to the advancement of our field. He will truly be missed around the globe.

Peer-to-Peer Management of XML Data: Issues and Research Challenges

Georgia Koloniari and Evaggelia Pitoura
Computer Science Department, University of Ioannina, Greece
{kgeorgia, pitoura}@cs.uoi.gr

ABSTRACT

Peer-to-peer (p2p) systems are attracting increasing attention as an efficient means of sharing data among large, diverse and dynamic sets of users. The widespread use of XML as a standard for representing and exchanging data in the Internet suggests using XML for describing data shared in a p2p system. However, sharing XML data imposes new challenges in p2p systems related to supporting advanced querying beyond simple keyword-based retrieval. In this paper, we focus on data management issues for processing XML data in a p2p setting, namely indexing, replication, clustering and query routing and processing. For each of these topics, we present the issues that arise, survey related research and highlight open research problems.

1. INTRODUCTION

The popularity of file sharing systems (such as Napster [33] and Gnutella [17]) has resulted in attracting much current research in peer-to-peer (p2p) architectures as an efficient means of sharing data. Peer-to-peer computing [32] refers to a form of distributed computing that involves a large number of autonomous computing nodes (the peers) that cooperate to share resources and services. The peers form logical overlay networks by establishing links to some other peers they know or discover. A user in a p2p system issues queries that describe data of interest. The queries are propagated through the overlay network to locate peers that provide data relevant to the query and any matching results are returned to the user.

Although the best-known application of p2p systems is file sharing (for example, music files in Napster), p2p system applications go beyond data sharing. Peer-to-peer computing is also a way of implementing systems based on the notion of increased decentralization and self-organization of systems, applications, or simply algorithms. By leveraging vast amounts of computing power, storage, and connectivity from personal computers distributed around the world, p2p systems provide a substrate for a variety of applications such as network monitoring and routing, web search and large scale event/notification systems.

XML [54] has evolved as the new standard for the representation and exchange of semistructured data on the Internet. Several application domains for XML already show that XML is inherently distributed on the Web, for example, Web services that use XML-based descriptions in WSDL and exchange XML messages with SOAP, e-commerce and e-business, collaborative authoring of large electronic documents and management of large-scale network directories. All these applications demonstrate that much of the traffic and data available in the Internet are already represented in XML format. Thus, it is natural to assume that much of the data in a p2p system is already represented in XML format. Furthermore, the deployment of XML as the underlying data model for p2p systems can provide a solution to two important issues in current p2p systems: the limited expressiveness of the available query languages and the heterogeneity of data.

In most p2p systems, different users and applications employ various formats and schemas to describe their data. A user is usually unaware of the schemas remote peers use. Moreover, some application domains, such as health-related applications, use sensitive data that are required not to be exposed to all users for privacy reasons. Therefore, there is a need for a query language that can work with incomplete or no-schema knowledge but also capture whatever semantic knowledge is available. The flexibility of XML in representing heterogeneous data that follow different schemas makes it suitable for distributed applications where the data are either native XML documents or XML descriptions of data or services that are represented in various formats in the underlying sources (i.e. in relational databases).

With regards to the query language, in most p2p systems, users specify the data they are interested in through simple keyword-based queries. These keywords are matched against the names of the shared files and any results are returned to the user. Often, most results returned are not relevant to what the user is interested in. Thus, new more expressive languages are needed to describe and query the shared data. XML seems to be a promising candidate in this direction, since it enables more precise search through provision of structural and self-describing metadata that allow for context and category-based search.

Traditionally, research work in XML querying has been following one of the two paths: the structured query approach (XQuery [7]) and the keyword-based approach (XKeyword [21], XSEarch [9]). While structured queries work effectively with the inherent structure of XML data and can convey complex semantic meaning, they require from the user to know the schema (or part of the schema) of the XML data to write the right query. The problem becomes even more difficult when the user has to deal with data from different schemas where the query requires rewriting before it can be evaluated. On the other hand, keyword-based searches do not require any knowledge about the underlying data schema but they also do not allow the users to convey semantic knowledge in their queries.

Motivated by the important role of XML in p2p systems, in this paper, we survey recent work on distributed processing of XML data in p2p systems. We focus on data management issues, such as indexing (Section 3), clustering (Section

4), replication (Section 5) and query processing and routing (Section 6). Although, schema integration of heterogeneous data is one of major issues in p2p processing [20], [22], it is beyond the scope of this survey.

2. P2P DATA MANAGEMENT

In this section, we provide a classification of p2p systems and describe some of their distinctive characteristics.

2.1 P2p characteristics

Scalability: While even in large traditional distributed systems, the number of participating nodes is in the order of hundreds; p2p systems must achieve scalability at the Internet-level.

Decentralization: Peer-to-peer computing is an alternative to the centralized and client-server models of computing, where there is typically a single or small cluster of servers and many clients. In its purest form, the peer-to-peer model has no concept of a server; rather all participants are equal, with each node given both server and client capabilities. Between the centralized and the pure peer-to-peer approach, there are hybrid p2p systems, in which some of the participants, called superpeers, have extended responsibilities and control over the others. The superpeers are often peers that have increased capabilities (storage and processing) and good stability properties.

Autonomy: We distinguish four kinds of autonomy, (i) storage, (ii) execution, (iii) lifetime and (iv) connection autonomy. *Storage autonomy* refers to the freedom of what a node in the system stores. In traditional distributed systems with central administration, the system enforces to the nodes which data items or indexes to store. P2p systems are self-configured: each peer stores its own data according to its interests and needs. Storage autonomy has another dimension related to *ownership* of data. This kind of autonomy allows a peer to determine which other peers in the system can store its own data or index information about its data.

Execution autonomy refers to the ability of a node to answer queries and change its own data. *Lifetime autonomy* refers to the freedom of each node to join and leave the system arbitrarily. In contrast to traditional distributed systems, p2p systems support this kind of dynamism and many issues concerning fault-tolerance, self-maintenance, ad-hoc connectivity and data availability arise from this requirement. Since the peers leave the system very frequently, a p2p system has to provide mechanisms to cope with these disconnections without causing significant problems in its operation. Furthermore, self-maintenance techniques should be used to deal with the frequent changes in the network. Finally, *connection autonomy* refers to the form of the overlay network in a p2p system. It enables a peer to select with how many and which peers it will connect to, based for example on trust or friendship with other users.

Due to the various forms of autonomy, many peers may exhibit selfish behavior. Selfish peers may refuse to evaluate or propagate queries or store index information or data copies. Such selfish peers try to exploit the resources and services provided by the system, without being willing to offer anything back to the peer community. To prevent this kind of behavior, p2p systems must provide incentives to peers for sharing their data and participating in query processing.

Other challenges that p2p systems need to cope with include anonymity, security and administration transparency.

A nice introduction to p2p systems and their basic goals and characteristics can be found in [32]. Some research challenges with emphasis on search are presented in [40], search and security issues are discussed in [13] and initial research ideas on data management in [19].

2.2 Types of p2p data management systems

We classify peer-to-peer data management systems based on (i) the degree of decentralization, (ii) the topology of the overlay network, (iii) the way information is distributed among the nodes and (iv) the type of data they store. Regarding the degree of decentralization, this varies from *pure* p2p systems, where all peers have equal roles, to *hybrid* architectures, where specific peers (the superpeers) are assigned different roles. Topology refers to the way the nodes in the p2p system are interconnected. Example topologies include the star, ring and the grid topology. In hybrid architectures, the topology of the superpeers may differ from that of the other peers. For instance, the superpeers may be fully inter-connected, while each simple peer is only connected with a single superpeer.

With regards to the distribution of information among the peers, we distinguish between structured and unstructured p2p systems. In *unstructured* p2p systems, there is no assumption about the distribution of data to the peers. Unstructured p2p systems can be further distinguished between systems that use indexes and those that are based on flooding and its variations. Topologies in unstructured p2p systems are usually not restricted to some regular structure, however, they may be regulated, for instance, by setting limits on the number of neighbors each peer can have.

In *structured* p2p systems, data items (or indexes of data items) are placed at specific nodes. Usually the distribution of data items to the peers is based on distributed hashing (DHTs) (such as in CAN [38] and Chord [45]). In DHTs, each item is associated with a key and each peer is assigned a range of keys and thus items. We make an additional distinction regarding structured p2p systems based on whether they assign to peers actual data items or indexes of items. Most DHT-based structured p2p systems follow a strict topology (such as a ring or torus) in which each peer has a specific number of neighbors. For example, in Chord, a hash function creates an m-bit identifier space. Identifiers are ordered on an identifier circle modulo 2^m, that forms the Chord virtual ring. As new peers join the system, their identifier, produced by hashing their IP address and port, is used for mapping them to the virtual ring. Data keys are also hashed and distributed to the peers according to their hash value, so that each key is assigned to its successor peer, which is the peer with the nearest hash-value traveling the ring clockwise. When a peer n joins the system, certain keys previously assigned to n's successor now become assigned to n. When peer n leaves the network, all of its assigned keys are reassigned to n's successor. Each peer maintains a finger table with its successors. Query routing proceeds by consulting the finger tables to locate the peer with the identifier closer to the search key. Other DHT-based systems exploit a less strict topology. For instance, P-Grid [1] builds a virtual distributed search tree which may be unbalanced.

DHT-based p2p systems support efficient key lookup (e.g., of order O(logn) in Chord). However, in most structured p2p systems, both storage and connection autonomy is compromised. The peers are forced to store information for data

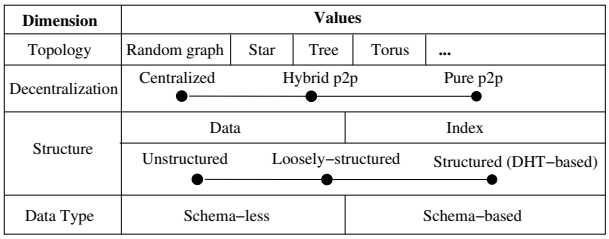


Figure 1: Classification of p2p systems

items assigned to them by distributed hashing and also follow a regulated topology. In addition, structured p2p systems require sophisticated load balancing procedures to cope with the increased demand for popular items and the dynamic behavior of the peers. There are also systems that are not based on distributed hashing, but do impose some structure. Such systems organize the overlay network into groups of peers with similar properties. We call such systems *loosely-structured* or *clustered* p2p systems.

Note that structured, unstructured and loosely structured p2p systems can use either pure or hybrid p2p architectures. For example in the case of structured DHT-based p2p systems, the DHT may be built only upon the superpeers, which have to follow the strict topology imposed by the system, while simple peers may connect to one or more superpeers without following any particular structure.

Finally, we distinguish between schema-less and schema-based p2p systems. In schema-based p2p systems, the peers use explicit schemas to describe their content. Schemas can be heterogeneous. A potential candidate for describing resources in p2p systems is the Resource Description Framework (RDF). RDF [39] is used to annotate resources on the Web, thus providing the means by which computer systems can exchange and comprehend data. RDF schemas are flexible and can evolve over time by allowing the easy extension of schemas with additional properties, thus being suitable for dynamic p2p systems. RDF usually uses an XML-based syntax to represent the metadata of the described resources. Apart from representing RDF descriptions in XML, RDF schemas can be used to provide semantic meaning for XML documents by using ontologies that are also described in RDF [23]. Figure 1 summarizes our taxonomy.

3. DISTRIBUTED INDEXES

In a p2p system, queries are initiated at various peers. These queries may require data that are located at a large number of peers distributed over the system. Traditionally, distributed systems use centralized or distributed indexes (catalogs) to store information about the location of data. For query processing, the indexes are consulted and the queries are sent to the appropriate nodes and evaluated there. Maintaining indexes in p2p systems poses additional requirements. In particular, indexes in p2p systems must support frequent updates, as peers join and leave the system constantly. Furthermore, the indexes need to be highly scalable, since the number of peers reaches Internet-scale, while in traditional distributed systems, the number of participating nodes is much smaller and controlled.

3.1 Types of p2p indexes

There are three basic approaches regarding indexes: maintaining (i) no index, (ii) a centralized index and (iii) a distributed index. When there is *no index* (such as in Gnutella [17]), some form of flooding is used for routing: the peer where the query is initiated contacts its neighbors in the overlay network, which in turn contact their own neighbors until the requested items are located or some system-defined bound is reached. Flooding does not compromise storage autonomy but incurs large network overheads. In the case of a *centralized index* (such as in Napster [33]), information about the contents of all peers in the system is maintained at a single peer. Peers that enter the system publish information about their data in this central index that is consulted when a query is submitted. The drawback of this approach is that the central index server becomes a bottleneck and a single point of failure. Maintaining replicas of the centralized index may increase reliability and scalability but still fails to handle efficiently the huge number of updates.

The distribution of the index depends on the overlay topology and on whether the system is structured or unstructured. In *structured p2p systems*, each peer stores index information for the data assigned to it by the hash function. For the evaluation of a query, its hash value is computed and the query is routed through the peer overlay network towards the peer that is responsible for storing the corresponding value.

In *unstructured p2p systems*, a popular form of distributed indexing is routing indexes [10]. The *routing indexes* of a peer summarize information about the contents of other reachable peers; they are used during routing to direct the queries towards the peers that are expected to hold relevant data. Since knowledge about all peers in the network is infeasible for scalability reasons, *horizons* are used to limit the number of other peers for which each peer stores information. Each index of a peer summarizes information about the data of all other peers that can be reached at a maximum distance h, where h is called the *radius* of the horizon. A peer may have just one such index or one for each of its links, summarizing the information of all peers on the path starting from this link and at a maximum distance h. Another distribution strategy is a hierarchical topology. In this case, the peers form one or multiple hierarchies. Each peer in the hierarchy stores information about the peers in its subtree, and the roots of the hierarchies are interconnected. The peers in the upper layers of the hierarchy assume most of the load and use their indexes to forward the queries to parts of the hierarchy where relevant data may be found.

Distributed indexes can also vary according to the degree of decentralization of the p2p system. In hybrid p2p systems, each superpeer is responsible for a number of other peers for which it stores index information. The superpeers are interconnected with each other following either a structured or an unstructured architecture. Each query is forwarded to a superpeer that is responsible for propagating it to the relevant peers or superpeers using its indexes. Query routing follows different protocols among the superpeers and the peers of the system and different types of indexes are used between superpeers and between superpeers and peers.

When using XML as the underlying data format, additional requirements arise for p2p indexes. In partcular, for XML documents, we need both *value* and *path indexes* for addressing the content as well as the structure of documents. Commonly used path indexes are indexes that assume an unordered tree structured data model and consist of a tree structure that summarizes path information [18]. Evaluating a query consists of traversing the path tree and match-

N1	D1: device/printer/postscript, D2: device/camera
N2	D3: device/printer/postscript, device/local/printer
N3	D4: device/camera
N4	D5: device/local/printer, D6: device/network/printer
N5	D7: printer/laser/color, device/camera
N6	D8: network/printer/laser
N7	D9: book/databases/greek
N8	D10: book/databases/english

Figure 2: Example of XML data distributed at peers

ing the path expression against the tree nodes. Other approaches assume that the data follow an arbitrary graph model. These indexes construct reduced graphs [36] that summarize all paths in the original data graph, by collapsing nodes that are equivalent (two nodes are equivalent if the paths from the root to them are the same).

Next, we describe in detail how indexes are adopted in some structured and unstructured p2p systems to support XML. We shall use the simple example depicted in Fig. 2 of 8 peers and their data, where for example, peer $N1$ stores 2 documents $D1$ and $D2$ containing paths "device/printer/postscript" and "device/camera" respectively.

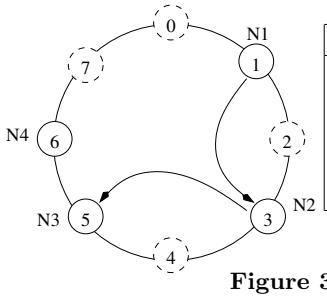
3.2 XML indexes in structured p2p systems

The use of XML as the format for data representation introduces additional problems in structured p2p systems. Most current structured p2p systems use document names as the data keys that are mapped to the underlying virtual network. Recent research [44], [49] has extended structured p2p systems by exploiting the content of documents for determining the keys. In particular, a vector describing each document is extracted and used as the key to map the documents to the virtual multi-dimensional space of the network. These vectors, used in information retrieval applications, typically consist of the document's keywords weighted by the frequencies of their appearance. The extracted vector is of much higher dimension than the dimension of the virtual space of the network. Thus, dimensionality reduction is required. This reduction should cause a minimal distortion, that is, the distance of the initial vectors should be approximated by the distance of the new vectors with the reduced dimensions. Vectors consisting of keywords and their corresponding weights are not suitable for representing XML data, since they do not capture the relationships between the elements (hierarchical structure). Thus, the challenge in this context is mapping the appropriate index keys for XML (both value and path indexes) to the multi-dimensional space created by distributed hashing.

In [15], a distributed catalog framework based on a structured p2p system is proposed. The system uses Chord [45] as the overlay network. The distributed catalog stores sets of key-summaries information for all the peers. The *keys* for XML data are either element or attribute names. The summaries that correspond to each key are either structural or value summaries. The *structural summaries* of a key are all possible paths leading to that key. The type of the *value summary* depends on the domain of the key, i.e. histograms are used for arithmetic keys. For each new peer that enters the system, each key-summary pair is inserted in the system by the Chord protocol according to the hash values of the keys. The class of supported XPath queries are of the form: $p = a_1[b_1]/a_2[b_2]/.../a_n[b_n]$ *op value*, where each a_i is a key and each b_i a path. The structural part of the query is handled using the structural catalog information, while the value predicates use the value summaries. For query routing, first all the simple paths ($sp_i = /a_{i1}/a_{i2}/.../a_{i,mi}$ *op value*) are extracted from the query. The peer responsible for the next $a_{i,mi}$ is found and the set of candidate peers for sp_i are retrieved using the catalog of that peer. The intersection of all the candidate peers that are produced after all the simple paths are processed is the set of peers that should receive the query.

Figure 3 shows an example of routing in Chord and its extension in [15]. The circles correspond to peers, while dotted circles correspond to logical positions (nodes) in the Chord ring that are not currently occupied by actual peers. The numbers within the circles correspond to the identifier of each logical node in the virtual space, while the labels next to them show which peers occupy the positions. The system has four peers $N1$ to $N4$ whose data are presented in Fig. 2. The table presents the index information that each peer holds for both Chord and its extension. The first column presents the finger tables of each peer, i.e., $N1$ knows that the successor of identifier 2 is peer 3, of identifier 3 again peer 3 and of identifier 5 peer 5. The finger tables are the same for both systems. For Chord, the index keys are the names of the files shared by the peers. Therefore, the second column of the table shows which files are assigned to each peer after the hash function is applied to their names. In the extended Chord (third column of the table), the indexed keys are the elements found in the documents along with their structural summaries denoted by S_i. Each S_i contains all the possible paths that lead to the corresponding indexed key in the contents of peer Ni. Let us assume that $N1$ issues the query: "local/printer". This query is not supported in Chord, therefore to retrieve this information the user must know the names of the files that contain it and pose a query using these names. On the other hand, the extended Chord is designed to support exactly this kind of queries. The arrows show the route of this query in the extended Chord.

XP2P [4] also extends Chord to support XML data. The system assumes that each peer stores a set of XML fragments (subtrees of XML data). In addition, each peer stores the local content of the user's fragments and their related path expressions that are the lists of each fragment's child fragments (path expressions stored as PCDATA within subtags in the fragment) and their super fragment (a path expression of the fragment which is the ancestor of the current fragment). These expressions are hashed into the Chord virtual space. The hashing technique used is different from that used in Chord. In particular, a fingerprinting technique is proposed based on [37]. The produced fingerprints are shorter than the hash keys used in Chord and support a concatenation property that allows the computation of the tokens associated with path expressions to proceed incrementally. Partial and full match lookups are supported, where in the first case, a match to a fragment is returned without unfolding its child fragments, while in the latter case, all the sub tags of the fragment are unfolded and the corresponding child fragments are retrieved. The queries are fingerpinted as well and when the fingerprint of a query (either in full or partial lookup) matches the fingerpint of a data fragment, the results are located by the lookup functionality of Chord. If the system cannot find a match, for instance if some peers are temporarily unavailable, additional techniques based on gradually pruning the query path are deployed to provide the user with at least a partial match.



Peer	Finger Table	Keys (Chord)	Keys–Summaries (extended Chord)
N1	2:3, 3:3, 5:5	D1	camera–S1{device}, S3{device}
N2	4:5, 5:5, 7:1	D2, D3	printer–S1{device}, S2{device, device/local} S4{device/network, device/local}
N3	6:6, 5:5, 7:1	D4, D5	–
N4	7:1, 0:1, 1:1	D6	postscript–S1{device/printer}

Figure 3: Indexes in structured p2p systems

RDFPeers [6] are based on MAAN (Multi-Attribute Addressable Network) which extends Chord to answer multi-attribute and range queries. Each RDF is viewed as a (subject, predicate, object) triple. Each triple is hashed and stored for each of its values in the corresponding positions in the Chord ring. For arithmetic attributes, MAAN uses order preserving hash functions so as to place close values to neighboring peers in the ring for the evaluation of range queries. Each query is transformed into triples and each value is searched as in Chord.

A DHT-based approach based on CAN is presented in [53]. The system presumes that the schema of the data is known by all peers. An overlay network similar to CAN is built where each dimension in the virtual multi-dimensional space corresponds to either a path level (a level of the path expression corresponding to some element name) or a unique attribute name on a specific path level. The dimensionality of the virtual space depends on the maximum depth of the path expressions and the number of distinct attributes each path level has. The virtual space is viewed as a hyper-rectangle and each distinct path corresponds to a logical node in the overlay network. The overall hyper-rectangles are disjointly partitioned among sub hyper-rectangles with exactly one logical node corresponding to each one of them. Each piece of XML data is mapped to a logical node according to its coordinates that are derived by hashing each element name and attribute that corresponds to each of the dimensions. Each peer keeps catalog information about all the paths that are mapped to it along with its own coordinates and its corresponding hyper-rectangle. Additionally, each peer keeps a routing table where it stores tuples of the form (coordinate, hyper-rectangle, address) for its neighbors in the virtual space. When a query is issued, it is also hashed to provide the query coordinates. When the queries consist of absolute location paths with only parent-child axis, the routing can be easily done by using the CAN routing mechanism which is enhanced in order to find the closest neighbor for propagating the query (finding the closest hyper-rectangle). If the query is more complex, it is transformed into one or more absolute location paths and the same mechanism is deployed.

In [34], a hybrid structured (non-DHT) p2p architecture is presented. The superpeers are organized into a hypercube topology that supports efficient broadcasting, while (non-super) peers connect to superpeers in a star-like fashion where each peer connects to only one superpeer. RDF metadata are used to describe the content of peers and to build routing indexes. Queries and answers to queries are also represented using RDF metadata. Two kinds of indexes are maintained at the superpeers: superpeer/peer indexes (SP/P) and superpeer/superpeer (SP/SP) indexes. SP/P indexes at a superpeer store information about metadata usage at each peer connected to it. This includes schema information such as schemas or attributes used, as well as possibly conventional indexes on attribute values. SP/SP indexes contain the same kind of information as SP/Ps, but refer to the direct superpeer neighbors of a superpeer. Queries are forwarded to superpeer neighbors according to the SP/SP indexes and then sent to the connected peers based on the SP/P indexes.

3.3 XML indexes in unstructured p2p systems

In unstructured p2p systems, research efforts focus on building space efficient routing indexes for XML documents. Most approaches build path indexes with the use of aggregation and suitable encoding schemes for the paths.

Figure 4 shows an example of query routing when using simple routing indexes such as in [10] and when XML-based routing indexes are used. The peers $N1$ to $N4$ hold the same data as in Fig. 2. Let us assume that the horizon is of radius 2. The gray circles represent the peers that are within peers $N1$ horizon. The table shows the routing indexes of peers $N1$ and $N3$ and their edges, for both simple routing indexes and path-based ones. Assume that $N1$ issues the same query as in the example of Fig. 3, that is "local/printer". If path indexes are not supported, the user must know the names of the files that contain the requested path expression. The arrows show the route the query follows when path-based routing indexes are used.

Two architectures for distributing the routing indexes, namely, the open and the agreement model, that differ in the degrees of shared knowledge among the peers, are proposed in [27]. In the open model, each peer is allowed to know about and potentially communicate with every other peer, while in the agreement model, each peer enters into bilateral agreements with some other peers called its neighbors. Path indexes are used as the internal organization of the routing indexes which maintain pointers from each path to the corresponding peers that contain it. Since the information included in a path index can grow excessively, aggregating paths with common prefixes is proposed to accommodate the routing index in a given space overhead.

Processing of containment queries in p2p systems is presented in [16]. Containment queries exploit the structure of XML data (i.e. book contains author contains name = "John Smith"). XML elements and text words are treated uniformly as index keys. Local indexes at each peer consist of inverted lists, which map keywords to XML documents stored at the peer. In addition to its local inverted lists, each peer also maintains routing indexes, called peer inverted indexes, that map keywords to the identifiers of remote peers. A query is forwarded to remote peers by using the peer inverted index and set operations are used to minimize the number of relevant destinations. Indexes are built when a

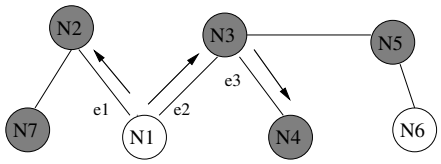
peer joins the system by exchanging information with other peers. These indexes are smaller than local indexes, since a peer only exchanges a small subset of its keywords, such as words that are often found in queries or that are representative of its local data. The result is a p2p system in which each peer has a summary of important data of all other peers. Horizons are used to limit the number of peers for which a peer has summarized information. A peer maps keywords outside of its horizon to peers on the boundary of the horizon that are closer to them.

In [26], each peer maintains a local index, summarizing its local content and one or more merged indexes summarizing the contents of its neighbors. The peers form hierarchies in which each peer stores summarized data for the peers belonging to its subtree. The roots are interconnected and store additional summaries for all other roots. Each peer that receives a query first checks its local index for any matches. Then, if it is an internal peer, it checks its merged index and if there is a match it forwards the query to its subtree. Furthermore, it sends the query to its parent or if it is a root peer to the other matching roots. The indexes used are based on Bloom filters that are compact data structures for the representation of a set of elements. To support the evaluation of regular XPath expressions, multi-level Bloom filters are introduced that preserve hierarchical relationships between the inserted elements. These relationships are preserved by inserting the elements of the XML tree to a different level of the filter according to their depth in the tree (Breadth Bloom filters), or by using paths of different lengths as keys and inserting them to the corresponding level of the filter according to their length (Depth Bloom filters).

A secure service discovery protocol for p2p systems is described in [12]. Service providers use the Service Discovery Service (SDS) to advertise descriptions and metadata of services expressed in XML. Clients use the SDS to locate the services they are interested in. The SDS servers are organized into hierarchies, which can be modified according to each server's workload. Each server is responsible for a particular domain; it receives advertisements and queries from a specific part of the network. When a server becomes overloaded, one or more child servers are spawned and assigned part of their parent domain. Each internal peer of the hierarchies stores summaries of the descriptions of its children, which are used for query routing. Summaries consist of a single Bloom filter. To insert a description in the filter, the description is divided to all possible subsets of the elements and attributes up to a certain threshold. Each query is split to all possible subsets and each one is checked in the index. The system emphasizes on security and access control issues and uses cryptography and authentication to ensure them.

4. CLUSTERING

Data clustering refers to grouping data items together to form clusters (groups) of items with common attributes or properties. In centralized systems, query performance may be improved by an appropriate placement of clustered data or indexes in main memory or in disk so that the I/O cost during query evaluation is minimized. In a distributed setting, clustering may improve query performance by reducing the communication cost through placing similar data at neighboring nodes. In a p2p setting, we further distinguish between (i) clustering similar data items (or indexes of similar data items) so that similar data (or indexes of similar data) are placed in neighboring peers and (ii) clustering peers with similar data items, so that their distance in the overlay network is small. By grouping similar peers, a query that reaches a peer in the cluster finds all other peers with relevant data nearby. Each form of clustering provides a different degree of storage autonomy. Data (or index) clustering violates storage autonomy, since it enforces peers to store specific items.



Peer	Edge	Keys	Keys (XML−based)
N1	e1	D3, ...	device/local/printer, device/camera, ...
	e2	D4, D5, D6, ...	device/camera, device/local/printer, device/network/printer ...
N3	e3	D5, D6, ...	device/local/printer, device/netwrok/printer, ...

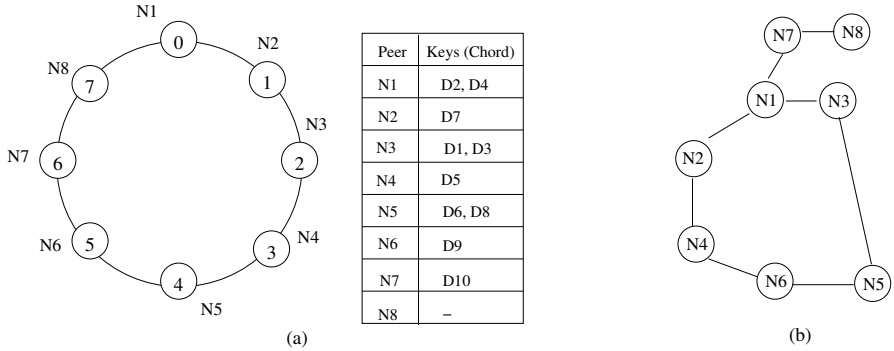
Figure 4: Indexes in unstructured p2p systems

In structured p2p systems, if the hash function is order-preserving, similar documents are stored at the same or neighboring peers. Order preserving hash functions are those hash functions that for similar inputs produce outputs close in the identifier space. Then, content-based clustering can be achieved by using as input to the hash function not just the name of the document but a semantic vector describing its content and structure.

Clustering the peers affects the topology of the p2p overlay. Issues of interest are how the peers within a cluster are interconnected (*intra-cluster organization*) and how the clusters are connected to each other (*inter-cluster organization*). Usually, query routing proceeds in two steps: first the appropriate cluster is identified and then the query is routed inside the cluster. Different mechanisms for intra-cluster and inter-cluster routing are often required.

In hybrid clustered p2p systems, superpeers act as cluster representatives. Each cluster contains at least one superpeer, which is in charge of the management of the cluster: query processing and peer information management. The superpeers collect their peers' schemas and communicate with each other for query evaluation.

Figure 5 illustrates the two techniques for clustering in p2p systems, data or index clustering (Fig. 5(a)) and peer clustering (Fig. 5(b)). The documents stored at each peer are described in Fig. 2. In Fig. 5(a), the peers form a structured p2p system that follows a ring topology. We assume that the vectors extracted from each of the documents are given as input to an order preserving hash function. The output of the hash function maps each document to the virtual ring. The table shows which documents are mapped to each peer. Documents with the same or similar content are mapped to the same or neighboring peers. For example, documents $D2$ and $D4$ that have the same content are both mapped to logical node 0, while documents $D3$ and $D5$ that share some content (i.e., the path expression "device/local/printer") are mapped to neighboring peers. On the other hand, in Fig. 5(b) where peers clustering is performed what is affected is the overlay network. In particular, two peers that have similar content are connected in the overlay network. For example, peer $N3$ is connected to both



Peer	Keys (Chord)
N1	D2, D4
N2	D7
N3	D1, D3
N4	D5
N5	D6, D8
N6	D9
N7	D10
N8	–

Figure 5: (a) Data (index) clustering and (b) peer clustering

$N1$ and $N5$ with which it has similar content (document $D2$ of $N1$ has the same content with $D4$ of $N3$, while document $D7$ of $N5$ shares a common path expression with $D4$).

When compared to distributed clustering, the autonomy of the peers makes the application of distributed clustering in a p2p context challenging. Peers may join and leave the system or change their content very often. The clusters in the system must adapt to reflect the new setting without requiring the application of the clustering procedure from scratch. Thus, the allocation of similar data to neighboring peers must be dynamic and incremental. Furthermore, clustering algorithms for centralized applications rely on having global knowledge of the data or the schema that the data follow and use this information to define the clusters or the categories to group or classify the data. However, the changing environment and the lack of global knowledge in p2p systems prohibits the use of predefined clusters or categories and clustering must be applied as an adaptive distributed procedure among peers using only partial knowledge about the data and the system's topology. Moreover, using a fixed number of clusters creates unbalanced clusters that cannot cope with changes in the system workload. Thus, load-balancing should also be taken into account in the clustering procedure, since some clusters may become over-populated and assume most of the query workload, while others may be almost empty.

In most current research, with regards to peer clustering, the number or the description of the clusters is predefined and fixed while global knowledge of this information is required. With Semantic Overlay Networks (SONs), peers with semantically similar content are logically linked to form overlay networks based on a classification hierarchy of their documents, which is defined a priori [11]. Queries are processed by identifying which SONs are better suited to answer them. In [52], clusters of peers are again formed based on the semantic categories of their documents. Sophisticated procedures are proposed for both inter-cluster and intra-cluster load balancing. Similarly in [3], peers are partitioned into topic segments based on their documents. A fixed set of C clusters is assumed, each one corresponding to a topic segment. Knowledge of the C centroids is global.

To perform clustering in a p2p system that uses XML as the underlying model, we need to identify which peers have similar content or which XML documents are similar and therefore should be grouped together. Thus, we need to define appropriate similarity measures for XML documents. Clustering for XML documents in centralized applications mostly relies on structural information. For the classification of schema-less data, the authors of [51] combine text terms, structural information in the form of twigs and paths and also ontological knowledge (WordNet [14]) to construct more expressive feature spaces that are then used for the classification. XRules [55] assigns the documents to categories through a rule based classification approach that relates the presence of a particular structural pattern in an XML document to its likelihood of belonging to a particular category. S-GRACE [29] is a hierarchical algorithm for clustering XML documents with a distance metric based on the notion of a structure graph, which is a minimal summary of edge containment in the data. Finally, XClust [28] addresses clustering when schema information in the form of DTDs is available in contrast with the previous methods that can be applied to schema-less data. XClust clusters DTDs based on the semantics, immediate descendants and leaf-context similarity of DTD elements. These centralized methods for XML clustering cannot be directly applied to p2p systems since they require global knowledge of the data or of their schema. They need to be adapted to work with incomplete local knowledge acquired by the cooperation of the peers.

In [26], a form of peer clustering is applied to an unstructured p2p system of XML peers. The peers are organized into hierarchies according to their content similarity. Content similarity is derived from the similarity of their routing indexes. Similarity takes into account both the structure and the content of data and is efficiently calculated from the routing indexes without requiring any knowledge of the schemas or the data of the other peers. Upon entering the system, each peer sends its index to the roots of the hierarchies that compare it with their own indexes. The peer attaches to the most similar hierarchy, so that peers with similar content are organized into the same hierarchy. The number of the hierarchies, i.e. clusters, is not fixed and changes according to the content of the new peers that enter the system. An adaptation of this clustering procedure for non-hierachical architectures is presented in [25].

The systems we describe next, assume that schema information is available. In structured p2p systems this information can be used to provide for an appropriate mapping of data items or peers to the virtual space. If the schema is known a priori, the virtual address space can be split to sub-spaces each one corresponding to a different part of the global schema. Then, upon entering the system, each peer (or data item) can be mapped to the sub-space of the virtual space that corresponds to its schema, thus creating clusters of peers that follow the same schema.

Along this line, in [43], the system assumes that all peers share a common topic ontology, and organizes them into concept clusters that are described by a logical combination

of ontology concepts. These concept clusters are organized into a hypercube topology. A hypercube topology is followed within the concept clusters as well. In particular, a peer in an ontology-based hypercube carries an address, which concatenates a set of concept coordinates (outer hypercube) and a set of storage coordinates (inner hypercube). In [35], the assumption is that the data of the peers belong to some categorization hierarchies relevant to a domain, called a multihierarchic namespace. Each hierarchy is called a dimension. The coordinates of a data item in the system are expressed as n-tuples. Groups of peers choose what data to host based on their own interests, thus defining their interest areas. Interest areas are defined as subsets of the cross product of some dimensions and are divided into interest cells. The interest areas describe index coverage of other groups' data and are encoded into URNs. The associated index servers to each interest area are contacted to find relevant data.

The next three systems are hybrid clustered p2p systems that use global schema information to organize peers into clusters. In SQPeer [24], peers are grouped based on their RDF-schema similarity. The peers that hold RDF descriptions conforming to the same RDF schema are clustered together. The system uses active schemas that are fine-grained schema advertisements that contain only information about what is actually stored in a peer. In XPeer [42], peers are logically organized into clusters that are also formed on a schema-similarity basis, whenever this is possible. Superpeers are organized to form a tree, where each peer hosts schema information about its children; superpeers having the same parent form a group. In both SQPeer and XPeer, the procedure that creates the clusters is not described and the authors focus on the exploitation of the clusters when such clusters exist. The system in [30], is based on RDF schemas that are assumed to be globally known. Rule-based clustering is used. Peers are registered and grouped in clusters based on cluster specific rules that describe the properties that each peer in the cluster should possess. These rules are provided by the cluster's administrator.

5. REPLICATION

The goals of replication in a p2p system do not differ much from those in a non p2p distributed system. Replication is basically used to improve system performance and to increase data availability in case of peer failures. Performance can be improved with replication either through balancing the load among the peers or by increasing locality, i.e. placing copies of data closer to their requestors. Replication refers either to the data items or their indexes. Issues of interest in both p2p and non p2p distributed systems include determining which items to replicate, where to place the replicas and how to keep them consistent. However, replication in p2p systems creates new requirements. The lifetime autonomy of the peers makes replication essential for keeping data available even when some of the peers storing them go offline. On the other hand, this makes enforcing consistency very difficult in p2p systems. In traditional distributed systems, we have either *eager* or *lazy replication*. Eager replication keeps all replicas exactly synchronized at all nodes. Lazy replication propagates updates asynchronously. Most applications of p2p systems do not require strict consistency, thus only lazy replication is usually applied, because of the high cost of eager replication. Furthermore, the lack of any central administration and of global knowledge of both the data and the query workload make the decision of which items to replicate and where much more complicated. Handling these issues depends both on the overlay topology and the routing mechanism in use.

With regards to the number of replicas, there are various approaches. At one extreme, *uniform replication* creates the same number of replicas for all data items irrespectively of their popularity. At the other extreme, *proportional replication* creates for each data item a number of C replicas, where C is proportional to its popularity, i.e. the number of queries that concern the given item, assuming that the popularity, i.e. the query workload, is globally known. While in proportional replication, popular queries are satisfied very efficiently since a large number of copies for the requested data is available across the system, unpopular data require many search steps thus compromising the overall system performance. On the other hand, with uniform replication, a large portion of the system resources is wasted in replicating data that appear in queries very rarely. Between these two extremes, *square-root replication* [8] creates copies so that for any two data items the ratio of replication is the square root of the ratio of their query rates. Square-root replication provides better results by leveraging the efforts for finding popular and unpopular data items.

In unstructured p2p systems, two different strategies are mainly used for placing replicas: path and owner replication. With *owner replication*, whenever a peer issues a successful query about a specific data item, a replica of the item is created at that peer. With *path replication*, copies of the requested data are stored at all peers along the path in the overlay network from the requestor peer to the provider peer [31]. Although path replication outperforms owner replication, it tends to replicate data items to peers that are topologically along the same path, which somewhat hurts the system performance. For updating the copies, the authors in [41] propose an investment policy where each individual peer propagates an update only if it estimates a benefit from doing so, i.e., an investment return.

For structured p2p systems, the issue that is addressed by replication is mostly load balancing and data availability. Bringing the data items closer to the requestors is not an issue since these systems already require only a small number of search steps to locate each item. Distributed hashing that assigns data items to peers does not take into consideration the popularity of each item, thus some peers may be assigned with many popular items. Such peers are burdened with satisfying most of the query workload and become hot spots. CAN [38] tries to solve this problem by using multiple hash functions to assign each item to more than one peers or by creating multiple coordinate spaces. As far as data availability is concerned, if a peer in a structured p2p system fails, all the items assigned to it become inaccessible even if they are stored at other peers, since the routing protocol cannot operate correctly. For this reason, in Chord [45], each peer stores a list of k of its successor peers, so if its successor fails, it can contact the first available successor in its list. Finally, an additional requirement for structured p2p systems that map indexes of data items to peers and replicate items instead of indexes, is that the index entries must be extended to maintain the identifiers of all the peers that hold copies of the indexed item.

The problem of replicating XML data or indexes in p2p systems has not received much attention yet. An important

issue that arises is the granularity of replication and distribution for XML. The issue is discussed in [5] but for a non p2p distributed XML repository. There, a global conceptual schema is used by a simplified structure called RepositoryGuide, which is a tree-structured index that resembles a DataGuide [18]. The system supports XML-path and tree pattern queries. The fragmentation scheme decomposes the RepositoryGuide into a disjoint and complete set of tree-structured fragments that preserve data semantics. A sublanguage of XPath is used for data fragmentation that supports vertical fragmentation, which is solely based on the selection of node types through path properties. The language can also be extended to support horizontal fragmentation, which includes conditions and branching, although some consistency issues arise. The allocation phase consists of three steps: determining which fragments to allocate at which system nodes, placing schema structures at local nodes and suitable instances of fragments at each node. For the first step, existing methods from distributed databases are used. For the second step, the RepositoryGuide is fully distributed among the nodes. Finally, for the third step, the global context of each fragment is kept by storing the data path from the global root node to the root of the local fragment. To this end, three indexes are used: a path index that encodes the global context of local fragments, a term index that allows processing of queries that include conditions on terms and an address index that stores the physical addresses of the fragments. Space efficient path indexes are constructed with the use of a path identification scheme. Because of their small size, path indexes are replicated at all system nodes, while term and address indexes are distributed among them.

The replication of XML indexes in structured p2p systems for load balance is described in [15]. Since some objects may be more popular, thus creating a considerable load to the peers that store them, two methods are proposed for splitting this load with other peers, namely the split-replicate and the split-toss methods that split catalog information among peers. A peer increases the level of a popular index key, where level is the number of XML-tree path steps contained in the key (initially set to 1), and either replicates the new keys at the corresponding peers (according to the hash function) in the Chord ring (split-replicate), or only sends them there and then discards them (split-toss). The peer also creates a mapping in its catalog for the new locations of the key, which it hands over to peers that request it. Once they obtain these mappings, they query one of the new locations in a round-robin fashion.

In [2], in contrast with [15], the actual XML documents, called dynamic XML documents, and not their indexes are replicated or distributed in an unstructured p2p system. Dynamic XML documents are XML documents that contain materialized XML data that are part of the document and intentional data that can be produced by service calls. Since dynamic documents may contain calls to services on other peers, some form of distribution is inherently part of the model. External edges are added to the XML document to point to peers that store other parts of the documents to allow for a higher form of distribution. A cost model is introduced and used for query processing and for deciding whether to replicate some parts of other peers' documents to increase the efficiency of query execution.

6. QUERY PROCESSING

Query processing in traditional distributed systems can be divided into four phases: query decomposition, data localization, global and local query optimization. Firstly, a relational query is decomposed into an algebraic query on global relations using techniques from centralized databases. Secondly, data distribution information is exploited to localize the algebraic query. The global query optimization phase receives an algebraic query on data fragments and finds an optimal strategy for its execution taking into account selectivity estimations and communication costs. At the last phase, each node receiving a query subplan tries to optimize it locally using centralized algorithms. The first three phases are performed centrally with global knowledge, while the fourth one is executed on each participating node with the knowledge available locally. For scalability reasons and to preserve autonomy, query processing in p2p systems must be coordination free. Furthermore, the lifetime autonomy of peers suggests that the execution plan must be constructed dynamically, since the number of participating peers and the available data changes with time.

We consider two generic approaches to this issue. The first approach is based on evaluating the query *incrementally*. A peer initiates the execution of a query, and the query is routed among the peers using the indexes. The execution plan evolves by accumulating partial results evaluated at each peer and locating the next peer with relevant data that needs to process the query. The other approach resembles the traditional approach with the known four phases but localization of data is achieved by using the indexes placed at each peer. In particular, the initiator of a query transforms the query into a path list and sends this list to remote peers where with the use of index-joins (similar to semi-joins) between the list and the peers' path indexes, the peers that store data relevant to the query are retrieved. In a way, this method relies on *index shipping* rather than query or data shipping that is common in traditional distributed systems. If the system is a hybrid p2p system, the queries may be sent to the superpeers that are responsible for constructing the execution plan and coordinating query evaluation.

Figure 6 illustrates the different scenarios for the distributed processing of the query illustrated in Fig. 6(a) issued by peer $N1$. The query asks for all the printers in a building that have printing quality larger than 600x450 dpi. Let us assume that the information about the available printers in the building is stored in peer $N4$ and the information about various printers characteristics in peer $N2$. The query can then be decomposed in Q_1 that retrieves all the printers with the desired quality from $N2$ and Q_2 that retrieves all the available printers in the building from $N4$. The lines between nodes correspond to overlay network connections, while the arrows show the route the query takes. In Fig. 6(b), we illustrate the traditional client-server scenario where the query is forwarded to a central server that has the available information for decomposing, localizing and optimizing globally the query. The server constructs the distributed query plan and forwards the two sub-queries to the corresponding peers for local optimization and evaluation. The results are returned to $N1$ that performs the final join. In Fig. 6(c), the superpeer based scenario is illustrated. Peer $N1$ sends the query to the superpeer it is connected to, namely SP_1. SP_1 interacts with the other superpeers in the network (dotted arrows) and gathers the re-

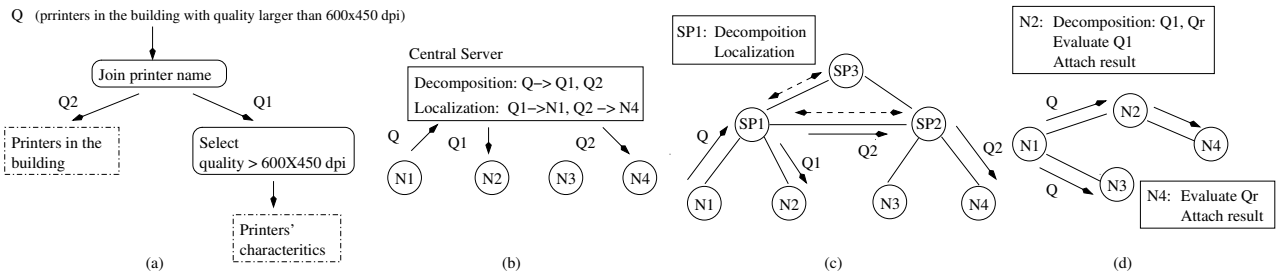


Figure 6: Distributed query processing scenarios

quired information to construct the distributed query plan. It then forwards the sub-query Q_1 to $N2$ that is attached to it, and Q_2 to SP_2 that in turn forwards the query to $N4$. The results are returned to SP_1 that performs the final join. Finally, in Fig. 6(d) we demonstrate the fully-distributed incremental query evaluation strategy. $N1$ issues the query and forwards it to its neighbors, following a flooding-based approach. When $N2$ receives the query, it realizes that it can evaluate a part of it (Q_1). It evaluates this part and attaches to the query plan the computed result. It then forwards the query further to its own neighbor $N4$. $N4$ evaluates the remaining part of the query and it either performs the final join itself or just attaches its own partial result and forward sit to $N1$ for the evaluation of the final join.

In the non p2p distributed XML repositories of [47], distribution is implemented by having links from the local XML data to XML objects at remote nodes. The data is represented by a rooted labeled graph. The model distinguishes between local links that point to local objects and cross-links that point to remote objects. Every node determines which of its data have incoming edges from other nodes (input data nodes) and which have outgoing edges to remote objects (output data nodes). Copies of the external data nodes are added to the node's graph. Given a query, an automaton is computed and sent to every node. Each node traverses only its local graph starting at every input data node and with all the states in the automaton. When the traversal reaches an output data node, it constructs a new output data node with the given state. Similarly, new input data nodes are also constructed. Once the result fragments, which consist of an accessibility graph that has the input and output data nodes and edges between them if and only if they are connected in the local fragment, are computed they are sent to the origin of the query. The client at the origin of the query assembles these fragments by adding missing cross-links, and computes all the data nodes accessible from the root. The algorithm requires only four communication steps and the size of the data exchanged depends only on the number of cross links and the size of the query answer.

Optimizing the cost of communication in answering XPath queries over distributed data based on the client-server model is considered in [48]. Minimal views that describe a query's results are used to avoid the redundancy met in such results where the same data may appear many times. The system leaves part of the evaluation of the query to the client that may have to extract all the answers from the minimal view to obtain the results to the initial queries.

Query processing in [5] is based on shipping index entries among nodes and efficiently evaluating chains of local joins of indexes. The node that issues a query determines the partitions of the query into path indexes lists that need to be sent to other nodes and instructs these nodes to compute the intermediate results. Then, the index results produced at the first node are sent to the second one to compute the join, and the resulting index is send to the third and so on. The final result is sent to the initial node where with the help of the RepositoryGuide the nodes that store the XML fragments defined by the resulting path index are retrieved. Similarly in the distributed RDF repository of [46], the main issue is to find the optimal ordering for a chain of joins of simple path expressions that are extracted from each query.

Mutant Query Plans (MQPs) [35] extend the weak query capabilities and limitations in index scalability and result quality of current p2p systems. A mutant query plan is an algebraic query plan graph, encoded in XML that may also include verbatim XML-encoded data, references to resource locations (URLs), and references to abstract resource names (URNs). Each MQP is tagged with a target, the peer that needs the results. An MQP starts as a regular query operator tree at the client peer and is then passed around among peers accumulating partial results, until it is fully evaluated into a constant piece of XML data. A peer can choose to mutate an MQP either by resolving a URN to one or more URLs, or a URL to its corresponding data. The peer can also reduce the MQP by evaluating a subgraph of the plan that contains only data at the leaves, and substituting the results in place of the subgraph. Resolving URLs is done either by connecting to the specified peer or by sending the MQP to it. For the URN the system's catalog is used. The cost model introduced in [2] for dynamic XML documents, is also used by a peer to decide on a query execution plan that minimizes its own observed cost. Each query is split to subqueries. A peer tries to satisfy locally as much of the query as it can and then forwards the remaining subqueries to the associated peers that follow the same procedure.

In [50], data is represented in XML and peers schemas in XML Schema. Query evaluation is incremental with an additional logical-level search where data are located based on schema-to-schema mappings. Instead of a local index, each peer maintains mappings between its own schema and the schemas of its immediate neighbors. Mappings are described as query expressions using a subset of XQuery. Peers are considered as connected through semantic paths of such mappings. Peers may store mappings, data or both. Query processing starts at the issuing peer and is reformulated over its immediate neighbors, which, in turn, reformulate the query over their immediate neighbors and so on. Whenever the reformulation reaches a peer that stores data, the appropriate query is posed on that peer, and additional results may be appended to the query result. Various optimizations are considered regarding the query reformulation process such as pruning semantic paths based on XML query containment, minimizing reformulations and precomputing some of the semantic paths.

Table 1: Issues and challenges in p2p management of XML

<table>
<tr><th></th><th></th><th>Structured P2P</th><th>Unstructured P2P</th><th>Non P2P</th></tr>
<tr><td rowspan="2">Indexing</td><td>Pure P2P</td><td>Mapping of paths onto the multi-dimensional virtual network [15], [6], [44], [49], [4], [53]</td><td>Use of space efficient routing path indexes through aggregation and encoding [26], [16], [27]</td><td rowspan="2">Less strict autonomy requirements, smaller scale</td></tr>
<tr><td>Hybrid P2P (superpeers)</td><td colspan="2">Superpeers responsible for routing hold all index information and propagate the queries to their peers
Different routing protocols and index structures between superpeers and between superpeers and peers [34], [12]</td></tr>
<tr><td rowspan="2">Clustering</td><td>Pure P2P</td><td>Clustered index
DHT-based clustering of peers based on their content by adopting the input of the hash function to split the multi-dimensional space into regions of peers with similar content [35], [43]
Non-DHT clustering based on schema similarity</td><td>Formation of peer clusters based on schema or content similarity, with the use of routing indexes
Different mechanism used for inter-cluster and intra-cluster routing [26]</td><td rowspan="2">Centralized techniques assume global knowledge</td></tr>
<tr><td>Hybrid P2P (superpeers)</td><td>Clustered index built only upon superpeers</td><td>Superpeers used as cluster representatives
Inter-cluster communication between superpeers [42], [24], [30]</td></tr>
<tr><td rowspan="2">Replication</td><td>Pure P2P</td><td rowspan="2">Data allocation with the use of distributed hashing
Replication requires additional mappings among peers responsible for the data (or indexes) and the peers that hold the replicas [15]</td><td rowspan="2">Replication increases data availability and performance [2]</td><td rowspan="2">Fragmentation-allocation of XML [5]</td></tr>
<tr><td>Hybrid P2P (superpeers)</td></tr>
<tr><td rowspan="2">Query Processing</td><td>Pure P2P</td><td colspan="2">Coordination-free based on the type of index
No clear distinction among query processing phases
Dynamic construction of the execution plan
Query passed around the peers accumulating partial results [35], [2], [50]</td><td rowspan="2">Distinct phases: query decomposition, localization, global and local optimization [47], [48]
Index shipping [5]
Ordering of joins [46]</td></tr>
<tr><td>Hybrid P2P (superpeers)</td><td colspan="2">Superpeers responsible for the construction and the coordination of the execution plan [42], [24]</td></tr>
</table>

In XPeer [42], peers export a tree-shaped DataGuide description of their data, which is automatically inferred by a tree search algorithm. The query language supported is the FLWR subset of XQuery without universally quantified predicates and sorting operations. Query compilation is performed in two phases by the superpeers. First, the peer that issues the query translates it into a location-free algebraic expression. Then, the query is sent to the superpeer network for the compilation of a location assignment. After the location assignment is completed, the query is passed back to the peer that issued it for execution to minimize the load of the superpeer network. The peer applies common algebraic rewriting and then starts the query execution: the query is split into single-location subqueries that are sent to the corresponding peers. Subqueries are locally optimized and the results are returned to the initial peer which executes operations such as joins involving multiple sources.

In SQPeer [24], queries are posed in RQL, according to the RDF schemas that are known to each peer. When a peer receives a query, it parses it and by obtaining the involved path creates the corresponding query pattern graph, which describes the schema information employed by the query. For each path pattern, the peers that contain the required data to answer it are discovered. If the schema of the peer is subsumed by the selected query path, then this query path is annotated with the name of the given peer. This is done for all known schemas that belong to remote peers. The result is an annotated query pattern graph with information for all peers that need to be contacted to answer the query. The query processing algorithm receives as input the annotated query graph and outputs the execution plan according to the underlying data distribution. Channels are used by the peers to exchange query plans and results.

7. CONCLUSIONS

In this paper, we present data management issues that arise in p2p systems that use XML as the underlying data model. Table 1 summarizes related issues and challenges in handling XML data. Regarding indexing, the main challenge is handling both value and path indexes. In structured p2p systems, the central problem is deriving appropriate mappings of documents to peers, while in unstructured p2p systems, the main issue is constructing space and update efficient routing indexes. Clustering in p2p refers to both (i) clustering index and data items and (ii) clustering peers with similar data so that their distance in the overlay network is small. Since most centralized XML clustering approaches assume global knowledge and relatively static environments, clustering in p2p systems poses new challenges. For replicating XML data in p2p systems, research is needed to cross-fertilize recent research results in (non XML) p2p replication and (non p2p) methods for XML fragmentation. Finally, processing XML queries in p2p systems is still an open issue. Techniques that handle the autonomy, decentralization, scale and dynamicity of p2p systems are required.

8. REFERENCES

[1] K. Aberer, P. Cudre-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-Grid: a Self-Organizing Structured P2P System. *SIGMOD Record*, 32(3), 2003.

[2] S. Abiteboul, A. Bonifati, G. Cobena, I. Manolescu, and T. Milo. Dynamic XML Documents with Distribution and Replication. In *SIGMOD*, 2003.

[3] M. Bawa, G. S. Manku, and P. Raghavan. SETS: Search Enhanced by Topic Segmentation. In *SIGIR*, 2003.

[4] A. Bonifati, U. Matrangolo, A. Cuzzocrea, and M. Jain. XPath Lookup Queries in P2P Networks. In *WIDM*, 2004.

[5] J. M. Bremer and M. Gertz. On Distributing XML Repositories. In *WebDB*, 2003.
[6] M. Cai and M. Frank. RDFPeers: A Scalable Distributed Repository based on a Structured Peer-to-Peer Network. In *WWW*, 2004.
[7] D. Chamberlin. XQuery: An XML query language. *IBM System Journal*, 41, 2003.
[8] E. Cohen and S. Shenker. Replication Strategies in Unstructured Peer-to-Peer Networks. In *SIGCOMM*, 2002.
[9] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv. XSEarch: A semantic Search Engine for XML. In *VLDB*, 2003.
[10] A. Crespo and H. Garcia-Molina. Routing Indices for Peer-to-Peer Systems. In *ICDCS*, 2002.
[11] A. Crespo and H. Garcia-Molina. Semantic Overlay Networks for P2P Systems. Technical report, Computer Science Department, Stanford University, 2002.
[12] S. E. Czerwinski, B. Y. Zhao, T. D. Hodes, A. D. Joseph, and R. H. Katz. An Architecture for a Secure Service Discovery Service. In *Mobicom*, 1999.
[13] N. Daswani, H. Garcia-Molina, and B. Yang. Open Problems in Data-Sharing Peer-to-Peer Systems. In *ICDT*, 2003.
[14] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
[15] L. Galanis, Y. Wang, S. Jeffery, and D. DeWitt. Locating Data Sources in Large Distributed Systems. In *VLDB*, 2003.
[16] L. Galanis, Y. Wang, S. Jeffery, and D. DeWitt. Processing Queries in a Large Peer-to-Peer System. In *CAiSE*, 2003.
[17] Knowbuddy's gnutella faq. http://www.rixsoft.com/Knowbuddy/gnutellafaq.html.
[18] R. Goldman and J. Widom. DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases. In *VLDB*, 1997.
[19] S. Gribble, A. Halevy, Z. Ives, M. Rodrig, and D. Suciu. What Can Databases Do for P2P? In *WebDB*, 2001.
[20] A. Y. Halevy, Z. G. Ives, and D. Suciu. Schema Mediation in Peer Data Management Systems. In *ICDE*, 2003.
[21] V. Hristidis, Y. Papakonstantinou, and A. Balmin. Key-word Proximity Search on XML Graphs. In *ICDE*, 2003.
[22] A. Kementsietsidis, M. Arenas, and R. Miller. Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues. In *SIGMOD*, 2003.
[23] M. Klein. *Interpreting XML via an RDF Schema. Chapter in Knowledge Annotation for the Semantic Web*. IOS Press, Amsterdam, 2003.
[24] G. Kokkinidis. and V. Christophides. Semantic Query Routing and Processing in P2P Database Systems: ICS-FORTH SQPeer Middleware. In *EDBT Workshop on P2P and DB*, 2004.
[25] G. Koloniari, Y. Petrakis, and E. Pitoura. Content-Based Overlay Networks for XML Peers Based on Multi-level Bloom Filters. In *DBISP2P*, 2003.
[26] G. Koloniari and E. Pitoura. Content-Based Routing of Path Queries in Peer-to-Peer Systems. In *EDBT*, 2004.
[27] N. Koudas, M. Rabinovich, D. Srivastava, and T. Yu. Routing XML Queries. In *ICDE*, 2004.
[28] M. L. Lee, L. Yang, W. Hsu, and X. Yang. XClust: Clustering XML Schemas for Effective Integration. In *CIKM*, 2002.
[29] W. Lian, D. Cheung, N. Mamoulis, and S. Yiu. An Efficient and Scalable Algorithm for Clustering XML Documents by Structure. *IEEE TKDE*, 16(1), 2004.
[30] A. Loser, W. Siberski, M. Wolpers, and W. Nejdl. Information Integration in Schema-Based Peer-To-Peer Networks. In *CAiSE*, 2003.
[31] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In *ICS*, 2002.
[32] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-Peer Computing. Technical report, HP Laboratories Palo Alto, TR HPL-2002-57, 2002.
[33] Napster. http://www.napster.com/.
[34] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, I. Brunkhorst, and A. Loser. Super-Peer-Based Routing and Clustering Strategies for RDF-Based Peer-to-Peer Networks. In *WWW*, 2003.
[35] V. Papadimos, D. Maier, and K. Tufte. Distributed Query Processing and Catalogs for Peer-to-Peer Systems. In *CIDR*, 2003.
[36] N. Polyzotis and M. Garofalakis. Structure and Value Synopses for XML Data Graphs. In *VLDB*, 2002.
[37] M. Rabin. Fingerprinting by Random Polynomials. Technical report, CRCT TR-15-81, Harvard University, 1981.
[38] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *SIGCOMM*, 2001.
[39] World-Wide Web Consortium: Resource Description Framework. http://www.w3.org/RDF.
[40] J. Risson and T. Moors. Survey of Research Towards Robust Peer-to-Peer Networks: Search Methods. Technical report, University of New South Wales, UNSW-EE-P2P-1-1, September 2004.
[41] M. Roussopoulos and M. Baker. CUP: Controlled Update Propagation in Peer-to-Peer Networks. In *USENIX*, 2003.
[42] C. Sartiani, P. Manghi, G. Ghelli, and G. Conforti. XPeer: A Self-organizing XML P2P Database System. In *EDBT Workshop on P2P and DB*, 2004.
[43] M. Schlosser, M. Sintek, S. Decker, and W. Nejdl. A Scalable and Ontology-Based P2P Infrastructure for Semantic Web Services. In *P2P*, 2002.
[44] C. Schmidt and M. Parashar. Flexible Information Discovery in Decentralized Distributed Systems. In *HPDC*, 2003.
[45] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *SIGCOMM*, 2001.
[46] H. Stuckenschmidt, R. Vdovjak, G. Houben, and J. Broekstra. Index Structures and Algorithms for Querying Distributed RDF Repositories. In *WWW*, 2004.
[47] D. Suciu. Distributed Query Evaluation on Semistructured Data. *TODS*, 27(1), March 2002.
[48] K. Tajima and Y. Fukui. Answering XPath Queries over Networks by Sending Minimal Views. In *VLDB*, 2004.
[49] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks. In *SIGCOMM*, 2003.
[50] I. Tatarinov and A. Halevy. Efficient Query Reformulation in Peer Data Management Systems. In *SIGMOD*, 2004.
[51] M. Theobald, R. Schenkel, and G. Weikum. Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data. In *WebDB*, 2003.
[52] P. Triantafillou, C. Xiruhaki, M. Koubarakis, and N. Ntarmos. Towards High Performance Peer-to-Peer Content and Resource Sharing Systems. In *CIDR*, 2003.
[53] Q. Wang and M. Oszu. A Data Locating Mechanism for Distributed XML Data over P2P Networks. Technical report, CS-2004-45, University of Waterloo, School of Computer Science, Waterloo, Canada, October 2004.
[54] World Wide Web Consortium. Extensible Markup Language (XML) 1.0 (Second Edition). http://www.w3.org/TR/REC-xml.
[55] M. J. Zaki and C. Aggarwal. XRules: An Effective Structural Classifier for XML Data. In *SIGKDD*, 2003.

Mining Data Streams: A Review

Mohamed Medhat Gaber, Arkady Zaslavsky and Shonali Krishnaswamy
Centre for Distributed Systems and Software Engineering, Monash University
900 Dandenong Rd, Caulfield East, VIC3145, Australia
{Mohamed.Medhat.Gaber, Arkady.Zaslavsky, Shonali.Krishnaswamy} @infotech.monash.edu.au

Abstract

The recent advances in hardware and software have enabled the capture of different measurements of data in a wide range of fields. These measurements are generated continuously and in a very high fluctuating data rates. Examples include sensor networks, web logs, and computer network traffic. The storage, querying and mining of such data sets are highly computationally challenging tasks. Mining data streams is concerned with extracting knowledge structures represented in models and patterns in non stopping streams of information. The research in data stream mining has gained a high attraction due to the importance of its applications and the increasing generation of streaming information. Applications of data stream analysis can vary from critical scientific and astronomical applications to important business and financial ones. Algorithms, systems and frameworks that address streaming challenges have been developed over the past three years. In this review paper, we present the state-of-the-art in this growing vital field.

1- Introduction

The intelligent data analysis has passed through a number of stages. Each stage addresses novel research issues that have arisen. Statistical exploratory data analysis represents the first stage. The goal was to explore the available data in order to test a specific hypothesis. With the advances in computing power, machine learning field has arisen. The objective was to find computationally efficient solutions to data analysis problems. Along with the progress in machine learning research, new data analysis problems have been addressed. Due to the increase in database sizes, new algorithms have been proposed to deal with the scalability issue. Moreover machine learning and statistical analysis techniques have been adopted and modified in order to address the problem of very large databases. Data mining is that interdisciplinary field of study that can extract models and patterns from large amounts of information stored in data repositories [30, 31, 34].

Advances in networking and parallel computation have lead to the introduction of distributed and parallel data mining. The goal was how to extract knowledge from different subsets of a dataset and integrate these generated knowledge structures in order to gain a global model of the whole dataset. Client/server, mobile agent based and hybrid models have been proposed to address the communication overhead issue. Different variations of algorithms have been developed in order to increase the accuracy of the generated global model. More details about distributed data mining could be found in [47].

Recently, the data generation rates in some data sources become faster than ever before. This rapid generation of continuous streams of information has challenged our storage, computation and communication capabilities in computing systems. Systems, models and techniques have been proposed and developed over the past few years to address these challenges [5, 44].

In this paper, we review the theoretical foundations of data stream analysis. Mining data stream systems, techniques are critically reviewed. Finally, we outline and discuss research problems in streaming mining field of study. These research issues should be addressed in order to realize robust systems that are capable of fulfilling the needs of data stream mining applications.

The paper is organized as follows. Section 2 presents the theoretical background of data stream analysis. Mining data stream techniques and systems are reviewed in sections 3 and 4 respectively. Open and addressed research issues in this growing field are discussed in section 5. Finally section 6 summarizes this review paper.

2- Theoretical Foundations

Research problems and challenges that have been arisen in mining data streams have its solutions using well-established statistical and computational approaches. We can categorize these solutions to data-based and task-based ones. In data-based solutions, the idea is to examine only a subset of the whole dataset or to transform the data vertically or horizontally to an approximate smaller size data representation. At the other hand, in task-based solutions, techniques from computational theory have been adopted to achieve time

and space efficient solutions. In this section we review these theoretical foundations.

2.1 Data-based Techniques

Data-based techniques refer to summarizing the whole dataset or choosing a subset of the incoming stream to be analyzed. Sampling, load shedding and sketching techniques represent the former one. Synopsis data structures and aggregation represent the later one. Here is an outline of the basics of these techniques with pointers to its applications in the context of data stream analysis.

2.1.1 Sampling

Sampling refers to the process of probabilistic choice of a data item to be processed or not. Sampling is an old statistical technique that has been used for a long time. Boundaries of the error rate of the computation are given as a function of the sampling rate. Very Fast Machine Learning techniques [16] have used Hoeffding bound to measure the sample size according to some derived loss functions.

The problem with using sampling in the context of data stream analysis is the unknown dataset size. Thus the treatment of data stream should follow a special analysis to find the error bounds. Another problem with sampling is that it would be important to check for anomalies for surveillance analysis as an application in mining data streams. Sampling may not be the right choice for such an application. Sampling also does not address the problem of fluctuating data rates. It would be worth investigating the relationship among the three parameters: data rate, sampling rate and error bounds.

2.1.2 Load Shedding

Load shedding refers [6, 52] to the process of dropping a sequence of data streams. Load shedding has been used successfully in querying data streams. It has the same problems of sampling. Load shedding is difficult to be used with mining algorithms because it drops chunks of data streams that could be used in the structuring of the generated models or it might represent a pattern of interest in time series analysis.

2.1.3 Sketching

Sketching [5, 44] is the process of randomly project a subset of the features. It is the process of vertically sample the incoming stream. Sketching has been applied in comparing different data streams and in aggregate queries. The major drawback of sketching is that of accuracy. It is hard to use it in the context of data stream mining. Principal Component Analysis (PCA) would be a better solution that has been applied in streaming applications [38].

2.1.4 Synopsis Data Structures

Creating synopsis of data refers to the process of applying summarization techniques that are capable of summarizing the incoming stream for further analysis. Wavelet analysis [25], histograms, quantiles and frequency moments [5] have been proposed as synopsis data structures. Since synopsis of data does not represent all the characteristics of the dataset, approximate answers are produced when using such data structures.

2.1.5 Aggregation

Aggregation is the process of computing statistical measures such as means and variance that summarize the incoming stream. Using this aggregated data could be used by the mining algorithm. The problem with aggregation is that it does not perform well with highly fluctuating data distributions. Merging online aggregation with offline mining has been studies in [1, 2, 3].

2.2 Task-based Techniques

Task-based techniques are those methods that modify existing techniques or invent new ones in order to address the computational challenges of data stream processing. Approximation algorithms, sliding window and algorithm output granularity represent this category. In the following subsections, we examine each of these techniques and its application in the context of data stream analysis.

2.2.1 Approximation algorithms

Approximation algorithms [44] have their roots in algorithm design. It is concerned with design algorithms for computationally hard problems. These algorithms can result in an approximate solution with error bounds. The idea is that mining algorithms are considered hard computational problems given its features of continuality and speed and the generating environment that is featured by being resource constrained. Approximation algorithms have attracted researchers as a direct solution to data stream mining problems. However, the problem of data rates with regard with the available resources could not be solved using approximation algorithms. Other tools should be used along with these algorithms in order to adapt to the

available resources. Approximation algorithms have been used in [13]

2.2.2 Sliding Window

The inspiration behind sliding window is that the user is more concerned with the analysis of most recent data streams. Thus the detailed analysis is done over the most recent data items and summarized versions of the old ones. This idea has been adopted in many techniques in the undergoing comprehensive data stream mining system *MAIDS* [17].

2.2.3 Algorithm Output Granularity

The algorithm output granularity (AOG) [21, 22, 23] introduces the first resource-aware data analysis approach that can cope with fluctuating very high data rates according to the available memory and the processing speed represented in time constraints. The AOG performs the local data analysis on a resource constrained device that generates or receive streams of information. AOG has three main stages. Mining followed by adaptation to resources and data stream rates represent the first two stages. Merging the generated knowledge structures when running out of memory represents the last stage. AOG has been used in clustering, classification and frequency counting [21].

Having discussed the different theoretical approaches to data stream analysis problems, the following section is devoted to stream mining techniques that use the above theoretical approaches in different ways.

3- Mining Techniques

Mining data streams has attracted the attention of data mining community for the last three years. A number of algorithms have been proposed for extracting knowledge from streaming information. In this section, we review clustering, classification, frequency counting and time series analysis techniques.

3.1 Clustering

Guha et al. [27, 28] have studied analytically clustering data streams using K-median technique. The proposed algorithm makes a single pass over the data stream and uses small space. It requires O(nk) time and O(nε) space where "k" is the number of centers, "n" is the number of points and $\varepsilon<1$. They have proved that any k-median algorithm that achieves a constant factor approximation can not achieve a better run time than O(nk). The algorithm starts by clustering a calculated size sample according to the available memory into 2k, and then at a second level, the algorithm clusters the above points for a number of samples into 2k and this process is repeated to a number of levels, and finally it clusters the 2k clusters into k clusters.

Babcock et al. [7] have used exponential histogram (EH) data structure to improve Guha et al. algorithm [27]. They use the same method described above, however they address the problem of merging clusters when the two sets of cluster centers to be merged are far apart by maintaining the EH data structure. They have studied their proposed algorithm analytically.

Charikar et al [11] have proposed another k-median algorithm that overcomes the problem of increasing approximation factors in the Guha et al [27] algorithm with the increase in the number of levels used to result in the final solution of the divide and conquer algorithm. The algorithm has also been studied analytically

Domingos et al. [15, 16, 35] have proposed a general method for scaling up machine learning algorithms. They have termed this approach Very Fast Machine Learning *VFML*. This method depends on determining an upper bound for the learner's loss as a function in number of data items to be examined in each step of the algorithm. They have applied this method to K-means clustering *VFKM* and decision tree classification *VFDT* techniques. These algorithms have been implemented and evaluated using synthetic data sets as well as real web data streams. VFKM uses the Hoeffding bound to determine the number of examples needed in each step of K-means algorithm. The VFKM runs as a sequence of K-means executions with each run uses more examples than the previous one until a calculated statistical bound (Hoeffding bound) is satisfied.

Ordonez [46] has proposed several improvements to k-means algorithm to cluster binary data streams. He has developed an incremental k-means algorithm. The experiments were conducted on real data sets as well as synthetic ones. He has demonstrated experimentally that the proposed algorithm outperforms the scalable k-means in the majority of cases. The proposed algorithm is a one pass algorithm in O(Tkn) complexity, where T is the average transaction size, n is number of transactions and k is number of centers. The use of binary data simplifies the manipulation of categorical data and eliminates the need for data normalization. The main idea behind the proposed algorithm is that it updates the cluster centers and weights after examining a batch of transactions which equalizes square root of the number of transactions rather than updating them one by one.

O'Challaghan et al. [45] have proposed STREAM and LOCALSEARCH algorithms for high quality data stream clustering. The STREAM algorithm

starts by determining the size of the sample and then applies the LOCALSEARCH algorithm if the sample size is larger than a pre-specified equation result. This process is repeated for each data chunk. Finally, the LOCALSEARCH algorithm is applied to the cluster centers generated in the previous iterations.

Aggarwal et al. [1] have proposed a framework for clustering data steams called CluStream algorithm. The proposed technique divides the clustering process into two components. The online component stores summarized statistics about the data streams and the offline one performs clustering on the summarized data according to a number of user preferences such as the time frame and the number of clusters. A number of experiments on real datasets have been conducted to prove the accuracy and efficiency of the proposed algorithm. They [2] have recently proposed HPStream; a projected clustering for high dimensional data streams. HPStream has outperformed CluStream in recent results.

Keogh et al [39] have proved empirically that most highly cited clustering of time series data streams algorithms proposed so far in the literature come out with meaningless results in subsequence clustering. They have proposed a solution approach using k-motif to choose the subsequences that the algorithm can work on to produce meaningful results.

Gaber et al. [21] have developed Lightweight Clustering *LWC*. It is an AOG-based algorithm. AOG has been discussed in section 2. The algorithm adjusts a threshold that represents the minimum distance measure between data items in different clusters. This adjustment is done regularly according to a pre-specified time frame. It is done according to the available resources by monitoring the input-output rate. This process is followed by merging clusters when the memory is full.

3.2 Classification

Wang et al. [53] have proposed a general framework for mining concept drifting data streams. They have observed that data stream mining algorithms proposed so far have not addressed the concept of drifting in the evolving data. The proposed technique uses weighted classifier ensembles to mine data streams. The expiration of old data in their model depends on the data distribution. They use synthetic and real life data streams to test their algorithm and compare between the single classifier and classifier ensembles. The proposed algorithm combines multiple classifiers weighted by their expected prediction accuracy. Also the selection of number of classifiers instead of using all is an option in the proposed framework without loosing accuracy in the classification process.

Ganti et al. [18] have developed analytically an algorithm for model maintenance under insertion and deletion of blocks of data records. This algorithm can be applied to any incremental data mining model. They have also described a generic framework for change detection between two data sets in terms of the data mining results they induce. They formalize the above two techniques into two general algorithms: GEMM and FOCUS. The algorithms have been applied to decision tree models and the frequent itemset model. GEMM algorithm accepts a class of models and an incremental model maintenance algorithm for the unrestricted window option, and outputs a model maintenance algorithm for both window-independent and window-dependent block selection sequence. FOCUS framework uses the difference between data mining models as the deviation in data sets.

Domingos et al. [15] have developed VFDT. It is a decision tree learning systems based on Hoeffding trees. It splits the tree using the current best attribute taking into consideration that the number of examined data items used satisfies a statistical measure which is Hoeffding bound. The algorithm also deactivates the least promising leaves and drops the non-potential attributes.

Papadimitriou et al. [48] have proposed AWSOM (Arbitrary Window Stream mOdeling Method) for interesting pattern discovery from sensors. They developed a one-pass algorithm to incrementally update the patterns. Their method requires only O(log N) memory where N is the length of the sequence. They conducted experiments with real and synthetic data sets. They use wavelet coefficients as compact information representation and correlation structure detection, and then apply a linear regression model in the wavelet domain.

Aggarwal et al. have adopted the idea of micro-clusters introduced in CluStream in On-Demand classification [3] and it shows a high accuracy. The technique uses clustering results to classify data using statistics of class distribution in each cluster.

Last [41] has proposed an online classification system that can adapt to concept drift. The system re-builds the classification model with the most recent examples. Using the error rate as a guide to concept drift, the frequency of model building and the window size are adjusted. The system uses info-fuzzy techniques for model building and information theory to calculate the window size.

Ding et al. [14] have developed a decision tree based on Peano count tree data structure. It has been shown experimentally that it is a fast building algorithm that is suitable for streaming applications.

Gaber et al. [21] have developed Lightweight Classification *LWClass*. It is a variation of *LWC*. It is also an AOG-based technique. The idea is to use K-nearest neighbors with updating the frequency of class occurrence given the data stream features. In case of contradiction between the incoming stream and the

stored summary of the cases, the frequency is reduced. In case of the frequency is equalized to zero, all the cases represented by this class is released from the memory.

3.3 Frequency Counting

Giannella et al. [20] have developed a frequent itemsets mining algorithm over data stream. They have proposed the use of tilted windows to calculate the frequent patterns for the most recent transactions based on the fact that users are more interested in the most recent transactions. They use an incremental algorithm to maintain the FP-stream which is a tree data structure to represent the frequent itemsets. They conducted a number of experiments to prove the algorithm efficiency.

Manku and Motwani [43] have proposed and implemented an approximate frequency counts in data streams. The implemented algorithm uses all the previous historical data to calculate the frequent patterns incrementally.

Cormode and Muthukrishnan [13] have developed an algorithm for counting frequent items. The algorithm uses group testing to find the hottest k items. The algorithm is used with the turnstile data stream model which allows addition as well as deletion of data items. An approximation randomized algorithm has been used to approximately count the most frequent items. It is worth mentioning that this data stream model is the hardest to analyze. Time series and cash register models are computationally easier. The former does not allow increments and decrements and the later one allows only increments.

Gaber et al. [21] have developed one more AOG-based algorithm: Lightweight frequency counting *LWF*. It has the ability to find an approximate solution to the most frequent items in the incoming stream using adaptation and releasing the least frequent items regularly in order to count the more frequent ones.

3.4 Time Series Analysis

Indyk et al. [36] have proposed approximate solutions with probabilistic error bounding to two problems in time series analysis: relaxed periods and average trends. The algorithms use dimensionality reduction sketching techniques. The process starts with computing the sketches over an arbitrarily chosen time window and creating what so called sketch pool. Using this pool of sketches, relaxed periods and average trends are computed. The algorithms have shown experimentally efficiency in running time and accuracy.

Perlman and Java [49] have proposed a two phase approach to mine astronomical time series streams. The first phase clusters sliding window patterns of each time series. Using the created clusters, an association rule discovery technique is used to create affinity analysis results among the created clusters of time series.

Zhu and Shasha [54] have proposed techniques to compute some statistical measures over time series data streams. The proposed techniques use discrete Fourier transform. The system is called StatStream and is able to compute approximate error bounded correlations and inner products. The system works over an arbitrarily chosen sliding window.

Lin et al. [42] have proposed the use of symbolic representation of time series data streams. This representation allows dimensionality/numerosity reduction. They have demonstrated the applicability of the proposed representation by applying it to clustering, classification, indexing and anomaly detection. The approach has two main stages. The first one is the transformation of time series data to Piecewise Aggregate Approximation followed by transforming the output to discrete string symbols in the second stage.

Chen et al. [12] have proposed the application of what so called regression cubes for data streams. Due to the success of OLAP technology in the application of static stored data, it has been proposed to use multidimensional regression analysis to create a compact cube that could be used for answering aggregate queries over the incoming streams. This research has been extended to be adopted in an undergoing project Mining Alarming Incidents in Data Streams *MAIDS*.

Himberg et al. [33] have presented and analyzed randomized variations of segmenting time series data streams generated onboard mobile phone sensors. One of the applications of clustering time series discussed: Changing the user interface of mobile phone screen according to the user context. It has been proven in this study that Global Iterative Replacement provides approximately an optimal solution with high efficiency in running time.

Guralnik and Srivastava [29] have developed a generic event detection approach of time series streams. They have developed techniques for batch and online incremental processing of time series data. The techniques have proven efficiency with real and synthetic data sets.

4- Systems

Several applications have stimulated the development of robust streaming analysis systems. The following represents a list of such applications.

- Burl et al. [9] have developed *Diamond Eye* for NASA and JPL. The aim of this project to enable remote computing systems as well as observing

scientists to extract patterns from spatial objects in real time image streams. The success of this project will enable "a new era of exploration using highly autonomous spacecraft, rovers, and sensors" [9]. This project represents an early development in streaming analysis applications.

- Kargupta et al. [37] have developed the first ubiquitous data stream mining system: *MobiMine*. It is a client/server PDA-based distributed data stream mining application for stock market data. It should be pointed out that the mining component is located at the server side rather than the PDA. There are different interactions between the server and PDA till the results finally displayed on the PDA screen. The tendency to perform data mining at the server side has been changed with the increase of the computational power of small devices.
- Kargupta et al. [38] have developed Vehicle Data Stream Mining System (*VEDAS*). It is a ubiquitous data mining system that allows continuous monitoring and pattern extraction from data streams generated on-board a moving vehicle. The mining component is located at the PDA on-board the moving vehicle. Clustering has been used for analyzing the driver behavior.
- Tanner et al. [51] have developed EnVironment for On-Board Processing (*EVE*). The system mines data streams continuously generated from measurements of different on-board sensors in astronomical applications. Only interesting patterns are transferred to the ground stations for further analysis preserving the limited bandwidth. This system represents the typical case for astronomical applications. Huge amounts of data are generated and there is a need to analyze this streaming information at real time.
- Srivastava and Stroeve [50] have developed a NASA project for onboard detection of geophysical processes represented in snow, ice and clouds using kernel clustering methods. These techniques are used for data compression. The motivation of the project is to preserve the limited bandwidth needed to send image streams to the ground centers. The kernel methods have been chosen due to its low computational complexity in such resource-constrained environment.

5- Research Issues

Data stream mining is a stimulating field of study that has raised challenges and research issues to be addressed by the database and data mining communities. The following is a discussion of both addressed and open research issues [17, 21, 26, 37]. The following is a brief discussion of previously addressed issues:

Handling the continuous flow of data streams: this is a data management issue. Traditional database management systems are not capable of dealing with such continuous high data rate. Novel indexing, storage and querying techniques are required to handle this non-stopping fluctuated flow of information streams.

Minimizing energy consumption of the mobile device: Large amounts of data streams are generated in resource-constrained environments. Senor networks represent a typical example. These devices have short-life batteries. The design of techniques that are energy efficient is a crucial issue given that sending all the generated stream to a central site is energy inefficient in addition to its lack of scalability problem [8].

Unbounded memory requirements due to the continuous flow of data streams: machine learning techniques represent the main source of data mining algorithms. Most of machine learning methods require data to be resident in memory while executing the analysis algorithm. Due to the huge amounts of the generated streams, it is absolutely a very important concern to deign space efficient techniques that can have only one look or less over the incoming stream.

Required result accuracy: design a space and time efficient techniques should be accompanied with acceptable result accuracy. Approximation algorithms as mentioned earlier can guarantee error bounds. Also sampling techniques adopt the same concept as it has been used in *VFML* [16].

Transferring data mining results over a wireless network with a limited bandwidth: knowledge structure representation is another essential research problem. After extracting models and patterns locally from data stream generators, it is essential to transfer these structures to the user. Kargupta et al. [37] have addressed this problem by using Fourier transformations to efficiently send mining results over limited bandwidth links.

Modeling changes of mining results over time: in some cases, the user is not interested in mining data stream results, but how these results are changed over time. If the number of clusters generated for example is changed, it might represent some changes in the dynamics of the arriving stream. Dynamics of data streams using changes in the knowledge structures generated would benefit many temporal-based analysis applications.

Developing algorithms for mining results' changes: this is related to the previous issue. Traditional data mining algorithms do not produce any results that show the change of the results over time. This issue has been addressed in *MAIDS* [10].

Visualization of data mining results on small screens of mobile devices: visualization of traditional data mining results on a desktop is still a research issue. Visualization in small screens of a PDA for example is a real challenge. Imagine a businessman and data are being streamed and analyzed on his PDA. Such results should be efficiently visualized in a way that enables

him to take a quick decision. This issue has been addressed in [37]

The above are the addressed research issues in mining data streams. Open Issues in the field are discussed in the following:

Interactive mining environment to satisfy user requirements: mining data streams is a highly application oriented field. The user requirements are considered a vital research problem to be addressed.

The integration between data stream management systems [4, 40] and the ubiquitous data stream mining approaches: it is an essential issue that should be addressed to realize a fully functioning ubiquitous mining. The integration among storage, querying, mining and reasoning over streaming information would realize robust streaming systems that could be used in different applications. Current database management systems have achieved this goal over static stored datasets.

The needs of real world applications: the relationship between the proposed techniques and the needs of the real world applications is another important issue. Some of the proposed techniques attempt to improve computational complexity of the mining algorithms with some margin error without taking care to the real needs of the applications that will use the proposed approach. Since data mining is an applied scientific discipline, the requirements of the applications should be stated clearly in order to achieve the analysis objectives.

Data stream pre-processing: the data pre-processing in the stream mining process should also be taken into consideration. That is how to design a light-weight pre-processing techniques that can guarantee quality of the mining results. Data pre-processing consumes most of the time in the knowledge discovery process. The challenge here is to automate such a process and integrate it with the mining techniques.

Model overfitting: the overfitting problem in data stream has not been addressed so far in the literature. Using some techniques such as cross validation is very costly in the case of data streams. Novel techniques are required to avoid model overfitting.

Data stream mining technology: the technological issue of mining data streams is also an important one. How to represent the data in such an environment in a compressed way? And which platforms are best to suit such special real-time applications? Hardware issues are of special concerns. Small devices are not designed for complex computations. Currently emulators are used to do this task and it is a real burden over data stream mining applications that run in resource-constrained environments. Novel hardware solutions are required to address this issue

The formalization of real-time accuracy evaluation: that is to provide the user by a feedback by the current achieved accuracy with relation to the available resources and being able to adjust according to the available resources.

The data stream computing formalization: mining of data streams is required to be formalized within a theory of data stream computation [32]. This formalization would facilitate the design and development of algorithms based on a concrete mathematical foundation. Approximation techniques and statistical learning theory represent the potential basis for such a theory. Approximation techniques could provide the solution, and using statistical learning theory would provide the loss function of the mining problem.

The above issues represent the grand challenges to the data mining community in this essential field. There is a real need inspired by the potential applications in astronomy and scientific laboratories [23] as well as business applications to address the above research problems.

6- Summary

The dissemination of data stream phenomenon has necessitated the development of stream mining algorithms. The area has attracted the attention of data mining community. The proposed techniques have their roots in statistics and theoretical computer science. Data-based and task-based techniques are the two categories of data stream mining algorithms. Based on these two categories, a number of clustering, classification, frequency counting and time series analysis have been developed. Systems have been implemented to use these techniques in real applications.

Mining data streams is still in its infancy state. Addressed along with open issues in data stream mining are discussed in this paper. Further developments would be realized over the next few years to address these problems. Having these systems that address the above research issues developed, that would accelerate the science discovery in physical and astronomical applications [23], in addition to business and financial ones [38] that would improve the real-time decision making process.

References

[1] C. Aggarwal, J. Han, J. Wang, P. S. Yu, A Framework for Clustering Evolving Data Streams, Proc. 2003 Int. Conf. on Very Large Data Bases, Berlin, Germany, Sept. 2003.

[2] C. Aggarwal, J. Han, J. Wang, and P. S. Yu, A Framework for Projected Clustering of High Dimensional Data Streams, Proc. 2004 Int. Conf. on Very Large Data Bases, Toronto, Canada, 2004.

[3] C. Aggarwal, J. Han, J. Wang, and P. S. Yu, On Demand Classification of Data Streams, Proc. 2004 Int.

Conf. on Knowledge Discovery and Data Mining, Seattle, WA, Aug. 2004.
[4] A. Arasu, B. Babcock. S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosenstein, and J. Widom. STREAM: The Stanford Stream Data Manager Demonstration description - short overview of system status and plans; in Proc. of the ACM Intl Conf. on Management of Data, June 2003.
[5] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In Proceedings of PODS, 2002.
[6] B. Babcock, M. Datar, and R. Motwani. Load Shedding Techniques for Data Stream Systems (short paper) In Proc. of the 2003 Workshop on Management and Processing of Data Streams, June 2003
[7] B. Babcock, M. Datar, R. Motwani, L. O'Callaghan: Maintaining Variance and k-Medians over Data Stream Windows, Proceedings of the 22nd Symposium on Principles of Database Systems, 2003
[8] R. Bhargava, H. Kargupta, and M. Powers, Energy Consumption in Data Analysis for On-board and Distributed Applications, Proceedings of the ICML'03 workshop on Machine Learning Technologies for Autonomous Space Applications, 2003.
[9] M. Burl, Ch. Fowlkes, J. Roden, A. Stechert, and S. Mukhtar, Diamond Eye: A distributed architecture for image data mining, in SPIE DMKD, Orlando, April 1999.
[10] Y. D. Cai, D. Clutter, G. Pape, J. Han, M. Welge, L. Auvil. MAIDS: Mining Alarming Incidents from Data Streams. Proceedings of the 23rd ACM SIGMOD International Conference on Management of Data, June 13-18, 2004, Paris, France.
[11] M. Charikar, L. O'Callaghan, and R. Panigrahy. Better streaming algorithms for clustering problems In Proc. of 35th ACM Symposium on Theory of Computing, 2003.
[12] Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. Multi-Dimensional Regression Analysis of Time-Series Data Streams In VLDB Conference, 2002.
[13] G. Cormode, S. Muthukrishnan What's hot and what's not: tracking most frequent items dynamically. PODS 2003: 296-306
[14] Q. Ding, Q. Ding, and W. Perrizo, Decision Tree Classification of Spatial Data Streams Using Peano Count Trees, Proceedings of the ACM Symposium on Applied Computing, Madrid, Spain, March 2002.
[15] P. Domingos and G. Hulten. Mining High-Speed Data Streams. In Proceedings of the Association for Computing Machinery Sixth International Conference on Knowledge Discovery and Data Mining, 2000.
[16] P. Domingos and G. Hulten, A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering, Proceedings of the Eighteenth International Conference on Machine Learning, 2001, Williamstown, MA, Morgan Kaufmann.
[17] G. Dong, J. Han, L.V.S. Lakshmanan, J. Pei, H. Wang and P.S. Yu. Online mining of changes from data streams: Research problems and preliminary results, In Proceedings of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams. In cooperation with the 2003 ACM-SIGMOD International Conference on Management of Data, San Diego, CA, June 8, 2003.
[18] V. Ganti, Johannes Gehrke, Raghu Ramakrishnan: Mining Data Streams under Block Evolution. SIGKDD Explorations 3(2), 2002.
[19] M. Garofalakis, Johannes Gehrke, Rajeev Rastogi: Querying and mining data streams: you only get one look a tutorial. SIGMOD Conference 2002: 635
[20] C. Giannella, J. Han, J. Pei, X. Yan, and P.S. Yu, Mining Frequent Patterns in Data Streams at Multiple Time Granularities, in H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha (eds.), Next Generation Data Mining, AAAI/MIT, 2003.
[21] Gaber, M, M., Krishnaswamy, S., and Zaslavsky, A., On-board Mining of Data Streams in Sensor Networks, Accepted as a chapter in the forthcoming book Advanced Methods of Knowledge Discovery from Complex Data, (Eds.) Sanghamitra Badhyopadhyay, Ujjwal Maulik, Lawrence Holder and Diane Cook, Springer Verlag, to appear
[22] Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., A Cost-Efficient Model for Ubiquitous Data Stream Mining, the Tenth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Perugia Italy, July 4-9.
[23] Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., Towards an Adaptive Approach for Mining Data Streams in Resource Constrained Environments, the Proceedings of Sixth International Conference on Data Warehousing and Knowledge Discovery - Industry Track (DaWak 2004), Zaragoza, Spain, 30 August - 3 September, Lecture Notes in Computer Science (LNCS), Springer Verlag.
[24] Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., Resource-Aware Knowledge Discovery in Data Streams, the Proceedings of First International Workshop on Knowledge Discovery in Data Streams, to be held in conjunction with the 15th European Conference on Machine Learning and the 8th European Conference on the Principals and Practice of Knowledge Discovery in Databases, Pisa, Italy, 2004.
[25] A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, M. Strauss: One-Pass Wavelet Decompositions of Data Streams. TKDE 15(3), 2003
[26] L. Golab and M. T. Ozsu. Issues in Data Stream Management. In SIGMOD Record, Volume 32, Number 2, June 2003.
[27] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. In Proceedings of

the Annual Symposium on Foundations of Computer Science. IEEE, November 2000.
[28] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, Clustering Data Streams: Theory and Practice TKDE special issue on clustering, vol. 15, 2003.
[29] V. Guralnik and J. Srivastava. Event detection from time series data. In ACM KDD, 1999.
[30] D. J. Hand, Statistics and Data Mining: Intersecting Disciplines *ACM SIGKDD Explorations*, 1, 1, pp. 16-19, June 1999.
[31]Hand D.J., Mannila H., and Smyth P. (2001) *Principles of data mining*, MIT Press.
[32] M. Henzinger, P. Raghavan and S. Rajagopalan, Computing on data streams , Technical Note 1998-011, Digital Systems Research Center, Palo Alto, CA, May 1998
[33] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmki, and H.T.T. Toivonen. Time series segmentation for context recognition in mobile devices. In Proceedings of the 2001 IEEE International Conference on Data Mining, pp. 203-210, San Jos, California, USA, 2001.
[34] Hoffmann F., Hand D.J., Adams N., Fisher D., and Guimaraes G. (eds) (2001) *Advances in Intelligent Data Analysis*. Springer.
[35] G. Hulten, L. Spencer, and P. Domingos. Mining Time-Changing Data Streams. ACM SIGKDD 2001.
[36] P. Indyk, N. Koudas, and S. Muthukrishnan. Identifying Representative Trends in Massive Time Series Data Sets Using Sketches. In Proc. of the 26th Int. Conf. on Very Large Data Bases, Cairo, Egypt, 2000.
[37] Kargupta, H., Park, B., Pittie, S., Liu, L., Kushraj, D. and Sarkar, K. (2002). MobiMine: Monitoring the Stock Market from a PDA. ACM SIGKDD Explorations. January 2002. Volume 3, Issue 2. Pages 37-46. ACM Press.
[38] H. Kargupta, R. Bhargava, K. Liu, M. Powers, P. Blair, S. Bushra, J. Dull, K. Sarkar, M. Klein, M. Vasa, and D. Handy, VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring, Proceedings of SIAM International Conference on Data Mining, 2004.
[39] E. Keogh, J. Lin, and W. Truppel. Clustering of Time Series Subsequences is Meaningless: Implications for Past and Future Research. In proceedings of the 3rd IEEE International Conference on Data Mining. Melbourne, FL. Nov 19-22, 2003.
[40] S. Krishnamurthy, S. Chandrasekaran, O. Cooper, A. Deshpande, M. Franklin, J. Hellerstein, W. Hong, S. Madden, V. Raman, F. Reiss, and M. Shah. TelegraphCQ: An Architectural Status Report. IEEE Data Engineering Bulletin, Vol 26(1), March 2003.
[41] M. Last, Online Classification of Nonstationary Data Streams, Intelligent Data Analysis, Vol. 6, No. 2, pp. 129-147, 2002.
[42] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. San Diego, CA. June 13, 2003.
[43] G. S. Manku and R. Motwani. Approximate frequency counts over data streams. In Proceedings of the 28th International Conference on Very Large Data Bases, Hong Kong, China, August 2002.
[44] S. Muthukrishnan (2003), Data streams: algorithms and applications. Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms.
[45] L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motwani. Streaming-data algorithms for high-quality clustering. Proceedings of IEEE International Conference on Data Engineering, March 2002.
[46] C. Ordonez. Clustering Binary Data Streams with K-means ACM DMKD 2003.
[47] B. Park and H. Kargupta. Distributed Data Mining: Algorithms, Systems, and Applications. To be published in the Data Mining Handbook. Editor: Nong Ye. 2002.
[48] S. Papadimitriou, C. Faloutsos, and A. Brockwell, Adaptive, Hands-Off Stream Mining, 29th International Conference on Very Large Data Bases VLDB, 2003.
[49] E. Perlman and A. Java. Predictive Mining of Time Series Data in Astronomy. In ASP Conf. Ser. 295: Astronomical Data Analysis Software and Systems XII, 2003.
[50] A. Srivastava and J. Stroeve, Onboard Detection of Snow, Ice, Clouds and Other Geophysical Processes Using Kernel Methods, Proceedings of the ICML'03 workshop on Machine Learning Technologies for Autonomous Space Applications
[51] S. Tanner, M. Alshayeb, E. Criswell, M. Iyer, A. McDowell, M. McEniry, K. Regner, EVE: On-Board Process Planning and Execution, Earth Science Technology Conference, Pasadena, CA, Jun. 11 - 14, 2002
[52] N. Tatbul, U. Cetintemel, S. Zdonik, M. Cherniack, M. Stonebraker. Load Shedding on Data Streams, In Proceedings of the Workshop on Management and Processing of Data Streams, San Diego, CA, USA, June 8, 2003.
[53] H. Wang, W. Fan, P. Yu and J. Han, Mining Concept-Drifting Data Streams using Ensemble Classifiers, in the 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Aug. 2003, Washington DC, USA.
[54] Y. Zhu and D. Shasha. StatStream: Statistical monitoring of thousands of data streams in real time. In VLDB 2002, pages 358-369.

Analytical Processing of XML Documents: Opportunities and Challenges

Rajesh R. Bordawekar Christian A. Lang
IBM T. J. Watson Research Center, Hawthorne, NY 10532
{bordaw,langc}@us.ibm.com

ABSTRACT

Online Analytical Processing (OLAP) has been a valuable tool for analyzing trends in business information. While the multi-dimensional cube model used by OLAP is ideal for analyzing structured business data, it is not suitable for representing and analyzing complex semi-structured data, such as XML documents. Need for analyzing XML documents is gaining urgency as XML has become the language of choice for data representation across a wide range of application domains. This paper describes a proposal for analyzing XML documents using the abstract XML tree model. We argue that OLAP's multi-dimensional aggregation operators can not express structurally complex analytical operations on XML documents. Hence, we outline new extensions to XQuery for supporting such complex analytical operations. Finally, we discuss various challenges in implementing XML analysis in a real system.

1. INTRODUCTION

Since its inception as a language for large-scale electronic publishing, Extensible Markup Language (XML) has emerged as the lingua franca for portable data representation. As a derivative of SGML, XML has been designed to represent both structured and semi-structured data. XML's ability to succinctly describe complex information can also be used for specifying application meta-data. XML's popularity is evident from its use in a wide spectrum of application domains: from document publication, to computational chemistry, health care and life sciences, multimedia encoding, geology, and e-commerce. Increasing popularity of web-based business processes and the emergence of web services has led to further acceptance of XML.

In spite of XML's wide-spread use, currently there are very few tools for analyzing XML data. XML data can be analyzed in two ways: (1) as semantically-rich text documents, and (2) as domain-specific data formulated using XML's semi-structured data model. Current efforts in XML analysis belong to the first category and use information retrieval techniques (e.g., keyword text searching) for knowledge discovery from XML documents [3, 18, 10]. To the best of our knowledge, there is no known work that analyzes XML data using domain-specific information. This is the focus of our current work.

An example of domain-specific analysis is Online Analytical Processing (OLAP) [8, 5], which has been extensively used by decision support systems. Such analysis is used to detect and predict trends in non-volatile time-varying business data. An OLAP system models the input data as a logical multidimensional *cube* with multiple *dimensions* which provide the context for analyzing *measures* of interest. Traditionally, measures are numeric values (e.g., units of sales or total sale amount) associated with the business data. Data analysis usually involves dimensional reduction of the input data using various aggregation functions, e.g., statistical (median, variance, etc.), physical (center of mass), and financial (volatility). Most database vendors support similar aggregation functions along with dimensional operators such as, `ROLLUP`, `GROUPBY`, and `CUBE`.

While OLAP is an effective tool for evaluating hierarchical relationships in structured data, its applicability is currently restricted to well-formulated business data that can be mapped to the multi-dimensional OLAP model. This prevents application of several useful OLAP features, e.g., grouping based on common data properties, structured aggregation, and trend analysis to XML data. There are three possible ways of using XML data in a data analysis system:

1. In the first approach, XML is used simply for external presentation of the OLAP results [20, 24]. The raw data is stored using either the relational (ROLAP) or the multi-dimensional (MOLAP) storage. Various data analysis operations (e.g., `CUBE` queries) are executed using the traditional multi-dimensional OLAP model.

2. In the second approach, input data is stored as XML documents. Relevant data is first extracted from the input XML documents using a XML processing language (e.g., XSLT, XQuery, or SQL/XML) and exported to the OLAP engine. The data analysis is still implemented using the multi-dimensional model. The results from the OLAP analysis may also be exported as XML documents.

3. The third approach uses XML both for data representation and processing. The data analysis engine represents the XML documents as trees using the tree-based (hierarchical) XML model and analyzes both the structure and the data values using an XML processing language.

In this paper, we examine analysis of an XML document using the tree-based XML model. In Section 2, we first review the XML data model and discuss why the multi-dimensional OLAP model is not suitable for analyzing XML documents. We then propose a new approach for analyzing XML documents using the XML data model and describe new operators for supporting structured aggregation over

XML data. Section 3.2 discusses various challenges in implementing the proposed XML analysis model in a real system. Section 4 discusses the related work. We present our conclusions and outline future work in Section 5.

2. ISSUES IN XML ANALYSIS

Traditional OLAP uses a regular multi-dimensional model where multiple *independent attributes* called dimensions *jointly* define the context for the corresponding numeric measures. Measures are those attributes of the data model that are used as input to the aggregation operations. Dimensions can have sub-attributes called, *members*, that exhibit *hierarchical non-recursive containment* relationships (e.g., the time dimension can have the following hierarchy[1] with members: year, quarter, month, days, and hours). Multi-dimensional OLAP is characterized by the following key features: (1) Input data organized into independent dimensions and numerical measures (e.g., using the star or snowflake schema on relational base tables), (2) Multi-dimensional array-like addressing of numeric measures and (3) Computations dominated by *structured* aggregation operations over numerical measures: (a) across levels of individual dimensions and (b) across dimensions.

Online analytical processing of XML documents raises issues that are substantially different from the traditional multi-dimensional OLAP. XML analysis differs both in the underlying data model and the prospective query patterns. We discuss these differences below:

Differences in the Data Model:

1. **Semantically rich contents:** XML is a flexible text format derived from SGML. An XML document is a text document whose textual entities are scoped in a hierarchy of self-descriptive markup tags. XML can be used to develop different domain-specific vocabularies that can encode the domain content via semantic markups and encode inherent relationships among the content entities via markup hierarchies. The XML data model views an XML document as a tree in which the internal nodes correspond to elements (denoting the markup), the leaves correspond to the textual content, and the tree edges correspond to the relationships among the content entities. Different axes in XML data can represent various relationships, e.g., *containment* (HAS-A) and *subclass* (IS-A) relationships.

   For analytical purposes, internal nodes of an XML tree (i.e., elements) can be viewed as members of scoped **dimensions**, where the dimension scope is determined by their parent elements, and values of the leaves can be viewed as the corresponding **measures**. In this model, dimension members are related to each other via XML's hierarchical structure. However, not all dimensions are mutually dependent, e.g., dimensions defined by **unique** siblings (and their subtrees) are independent within the scope of their parent dimension. Further, unlike traditional OLAP, classification between dimensions and measures is not rigid. Any XML `element` can be associated with a set of attributes that provide additional information on that element. Such information could also be used for analysis purposes. In other words, some *dimensions* could also be analyzed as *measures*. The push and pull operators [22] provide similar functionalities for relational OLAP systems.

2. **Semi-structured Data Model:** Unlike relational data, XML data (or documents) does not adhere to a rigid schema and can exhibit irregular structure. At the same time, all *well-formed* XML documents conform to an abstract XML tree whose nodes are ordered in an in-order, depth-first manner (called the document order). XML documents can have recursive hierarchies or hierarchies with different members. Thus, XML is an ideal representation of *semi-structured* data. The flexible structure of an XML document can be specified using a strongly-typed XML schema [7]. Potentially, more than one XML *instance* document can map to an XML schema. Unlike the multi-dimensional OLAP, the context of a measure is defined by the hierarchy in which it is scoped. In an XML document, a measure attribute can appear in more than one contexts (or hierarchies). Therefore, an analytical operation over a measure in one context may not be applicable for the same measure in another context. Finally, since XML nodes are ordered in the document order, *measures* themselves could be *semantically related* by the order relationship.

3. **Navigational addressing:** The abstract tree to represent the XML document is addressed using the XPath navigational language [7]. XPath navigates the abstract XML tree via five distinct axes. These axes support navigation on the tree over explicit parent-child edges and implicit edges such as sibling edges. Hence, any node of an XML tree can be addressed in a multitude of ways. This is in contrast to the rigid array-based addressing in the OLAP data model.

4. **Non-numeric measures:** Traditional OLAP involves analyzing only numeric measures (e.g., sales) of business data using aggregation functions. Since XML is also used for specifying non-business data (e.g., genome databases), it can have both numeric and non-numeric data (e.g., ATCG strings representing amino acid sequences) that need to be analyzed.

Differences in the Query Patterns:

1. **Order-dependent queries:** The XML data model enforces a strict document ordering of the XML nodes. The XML node ordering is exploited by XPath to support position-based queries on the XML tree, e.g., identify the first child of a node. Similar position-based queries could be used for analyzing ordered data sets whose ordering carries certain semantics. For example, consider an XML document that stores effects of a drug on a bio-metric parameter (e.g., white blood cell count) in a clinical drug study [13]. Figure 1 represents the corresponding abstract XML tree. Typical order-dependent analytical queries on this document can include: (1) For each asthma drug, compare the blood cell count after every usage with the corresponding count for the healthy case, (2) Determine those drugs whose second usage results in the maximum change in the white blood cell count, or (3) For

[1] A dimension can have more than one hierarchy.

all asthma drugs, find the maximum variation in the white blood cell count after the second usage.

2. **Generalized grouping:** Typical relational OLAP operations such as `GROUPBY`, `ROLLUP` or `CUBE` group tuples of a relation based on *values* of its column attributes. In XML analysis, one can also group XML entities based on their *path attributes* that encode entity relationships. Path attributes can be specified via XPath expressions or can use generalized tree patterns specified using regular path expressions. Section 3 presents a proposal of incorporating structural grouping features into XQuery.

3. **Queries on non-numeric measures:** Non-numeric (textual) measures could be used in two types of queries: (1) Structured queries which involve aggregation operations over strings, e.g., find the maximum or average length of the string measures, and (2) approximate queries which involve substring or string pattern matching. An example application is searching for similar images in MPEG-7 [9]. The MPEG-7 standard is based on XML and allows the storage of image and video features as strings. Similarity searching on images and videos is thereby transformed into similarity searching on strings.

4. **Approximate hierarchical queries:** Due to the semi-structured nature of XML documents, analyses based on tag names and path specifications are affected by structure changes, misspellings, or renamings. Consider, for example, the use of `surname` rather than `lastName` in an XPath expression. Similarly, the tag order in a path specification may be inverted, as in `//order/customer/address` versus `//customer/order/address`. In such cases, approximate tag and path matching based on semantic similarity [23] and sequence similarity [11] become a necessity.

5. **Hierarchical slice&dice:** In a traditional OLAP system, slicing involves reducing dimensions of a data cube and then projecting the data cube using the reduced dimension. Equivalently, an XML tree could be sliced over its independent dimensions by selectively eliminating the subtrees in those dimensions. Similarly, the dicing operation identifies and removes subtrees based on values derived from structural properties (e.g., depth of an XML node) or node values.

6. **Structural analytics:** In the traditional OLAP system, *what-next* analysis has been extensively used to predict future trends. The what-next analysis involves modifying values of certain measures and studying its impact on the overall data trends by using different aggregation functions. In XML analysis, one can evaluate the impact of relationships by modifying structures of the XML data. For example, consider an XML document describing the structure of an organization where the organization has many divisions, each division has many departments, each department has many groups, and each group consists of several employees. Each division has a fixed budget which gets percolated down the organization hierarchy according to a certain formula. Consider an analyst who wants to find out the impact of the organization hierarchy on a group's budget. She can rerun the budget computation by moving the group to another departmental hierarchy. Existing OLAP systems can not support such structural analytics.

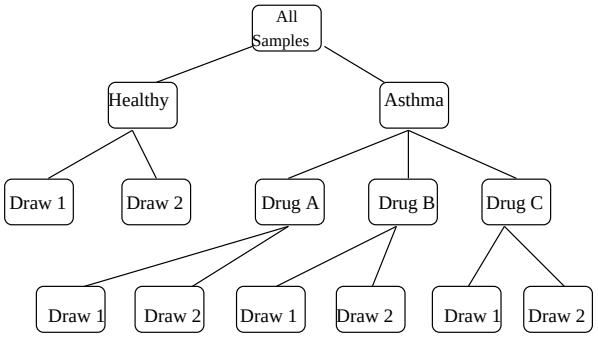


Figure 1: An Example of Order-dependent OLAP Query from a clinical-study application.

3. A MODEL FOR XML ANALYSIS

As described in Section 2, key requirements of an XML analysis system are: (1) ability to represent semi-structured data, (2) ability to support order-oriented queries, and (3) ability to support value and structural queries over numeric and non-numeric measures. The abstract tree model proposed by the XML standard [7] satisfies the first two requirements easily. Further, any XML document can be mapped to the abstract model without any additional formulations. Hence, we base our *logical* analytical model on XML's abstract tree model. Any XML document that is being analyzed is first parsed and translated into a logical tree-based representation (note that the actual physical representation of the document can vary). This logical tree can then be traversed using existing XML languages, e.g., XPath, XSLT, and XQuery. It is interesting to note that many XML analysis queries can be specified using these languages. For example, hierarchical slice & dice can be easily implemented using an XQuery application. XQuery could also be used for implementing structural analytics. Useful analytical features missing from XQuery are structured grouping and aggregation. We now propose simple yet powerful extensions to XQuery to support these features. We then discuss various challenges in implementing our analysis model in a real system.

3.1 Structured Aggregation Operators

As in traditional OLAP, we refer to the components of XML analysis as *analysis dimensions*. Our model comprises two types of analysis dimensions: value-based and structure-based dimensions. The former specify text contents in an XML document (which can be values of XML `attribute` or `text` nodes), while the latter specify tree patterns in an XML document (and therefore represent a set of valid pattern matches).

As an example, consider a human resources XML document storing the employee data (Figure 2). The organization consists of divisions (highest level), departments, and groups (lowest level). Each group again consists of employees that are members of this group. As indicated, each em-

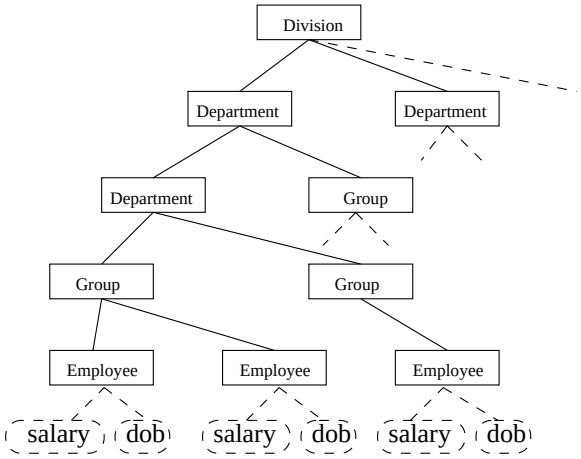


Figure 2: Human resources XML Document

ployee has attributes `salary` and `dob` (date of birth). For this example, a value-based analysis dimension would be `//employee/dob` which represents the set of all dates of birth of all employees. Similarly, a structure-based analysis dimension could be specified as `//division/{$x | $x/$y}/group`. This pattern (among others) matches to the following tree patterns:
`/division/department/group` and
`/division/department/department/group`. The first match binds the variable `$x` to the `department` nodes and the second match binds both variables `$x` and `$y` to `department`. Note that the pattern specification syntax is presented for illustrative purposes only. We envision the use of a notation that allows the specification of more general tree patterns as discussed by Chen et al. [6].

3.1.1 Structured XML Group-By

Consider an analytical query, *What is the average income per dob range?* over the human resources XML document. Consider that `dob` is already reported as ranges in the XML document. Then an XML query to answer this question could look like this (for simplicity, we use a slightly modified FLWOR [7] syntax.):

```
for $e in //employee
GROUP BY(//employee/dob)
let $s := avg( $e/salary )
return
         <ageGroup>
            <dob_range> $e/dob </dob_range>
            <avgSalary> $s </avgSalary>
         </ageGroup>
```

In this example, `//employee/dob` is a value-based analysis dimension. Note that one could also answer the question, *What is the average dob per salary range?*, by using `//employee/salary` as the analysis dimension. This is in contrast with the traditional OLAP where usually the set of measure attributes and dimensions is predefined and cannot be changed easily.

Consider another question, *What is the average salary per department level?*. This question requires the use of a structure-based analysis dimension:

```
var d,d1,d2=department
for $e in //employee
GROUP BY (//division/{$d | $d1/$d2}/employee )
let $s := avg( $e/salary ),
    $l := ( $d1 == NULL ? 1 : 2 )
return
         <levelGroup>
            <level> $l </level>
            <avgSalary> $s </avgSalary>
         </levelGroup>
```

In this case, the averaging in the `let`-clause is done over the `employee` nodes with the same number of `department` nodes on the path. Similarly, the following `group-by`

```
GROUP BY ( //division/{$d | $d1/$d2}/employee,
           //employee/dob ),
```

averages salaries over different age ranges in different department levels. Note that one can also use a Dewey-ID based numbering scheme [10] to encode the element level.

While simpler versions of the value-based grouping operations can be implemented using a nested XQuery (see Appendix G.2, XQuery Working Draft, 12 November 2003 [7] and [19]), current XQuery proposal can not express structural grouping operations.

3.1.2 Structured XML Roll-Up and Cube

We now describe structured roll-up and cube operations in the XML analysis model. As an example, assume that each group node in our example document (Figure 2), stores the corresponding budget information. An analyst may be interested in viewing the budgets per group but also at higher levels. This is similar to a traditional OLAP roll-up operation and could be written as follows:

```
var d, d1=department
for $b in //budget
GROUP BY ROLLUP( //division//group,
//division/$d//group, //division/$d/$d1/group )
let $s := sum( $b/value() ),
    $n := $b/../name()
return
     <levelGroup>
       <name> $n </name>
       <overallBudget> $s </overallBudget>
      </levelGroup>
```

As in the traditional OLAP, the `ROLLUP` clause specifies the order in which the roll-up occurs. In this example, the roll-up sequence is

```
( //division//group, //division/$d//group,
  //division/$d/$d2/group)
( //division//group, //division/$d//group )
( //division//group )
```

The result of this `ROLLUP` would therefore be the sum of budgets of the groups on the lowest level, followed by the sum of budgets of the groups on the second-highest level, followed by highest level groups. Note that the `ROLLUP` clause in the example uses the same variable `$d` in the last two dimensions. This ensures that the roll-up operation correctly uses the department hierarchy for aggregation.

We note here that the `ROLLUP` operator does not require the dimensions to be dependent as in this example. One can equally well imagine a roll-up such as

```
GROUP BY ROLLUP({//division/{$d | $d1/$d2}/employee},
//employee/dob)
```

which will compute aggregates first over employees in the same age-group within the same level and then over employees in the same group level.

Drill-down and cube operators can be defined similarly and will therefore be skipped for brevity. The interested reader is referred to Gray et al. [8] who discuss the generalization of group-by to roll-up and cube in more detail for the traditional OLAP setting.

Finally, we propose a special operator, called *topological roll-up* for performing roll-up aggregation along an XML hierarchy. Instead of specifying

```
GROUP BY ROLLUP(//division//group,
//division/$d//group, //division/$d/$d2/group)
```

one can use the `topological` specifier as follows:

```
GROUP BY TOPOLOGICAL ROLLUP(//division/$d/$d2/group)
```

The complete roll-up as discussed above could then be generated automatically by topological sorting of the XML structures. Similarly, drill-down and cube operators could be extended with the "`topological`" keyword to indicate automatic expansion or contraction of the XML structure. Thus the precise schema of the XML documents need not be known to the user. The simple expression

```
GROUP BY TOPOLOGICAL CUBE(//division/$d1/$d2/employee,
 //division/$d1/$d2/budget)
```

would then correspond to the much longer expression

```
GROUP BY CUBE(//division/employee,
//division/$d1/employee, //division/$d1/$d2/employee,
//division/budget, //division/$d1/budget,
//division/$d1/$d2/budget)
```

3.2 Implementation Challenges

We now discuss various challenges in implementing a real-life analysis system based on the proposed hierarchical XML analysis model.

1. **Supporting new XML analytics operators:** The XML analysis model proposed in Section 3 introduces several new aggregation operators such as the structural group by, topological roll-ups, and cubes. Efficient implementation of these operators in the XQuery framework would be challenging. Another area that needs further investigation is support for approximate structural and value queries.

2. **Supporting large XML documents:** Decision support systems traditionally deal with large datasets stored in data warehouses. In practice, terabyte-sized OLAP cubes are not uncommon. Similarly, data sources such as the Protein Data Bank (PDB) can generate XML documents with sizes of 100 GByte and more. Therefore, it is imperative that any XML analysis system is able to load and process large complex XML documents efficiently. Current XML processing systems tend to materialize an entire document in memory, which is clearly impractical. Techniques such as XML projection [17] are needed to materialize XML documents with smaller memory footprints.

3. **Creating complex XML views:** Until now, we have considered analysis of a single large XML document. In practice, there might be a large number of XML documents, potentially with different schemas. In such cases, one may need to compute XML views using multiple *base* XML documents. In theory, XQuery can be used for generating XML views from multiple base documents, but in reality, most current XQuery implementations can work on only one or two XML documents.

4. **Support for visualizing complex hierarchical data:** Visualization forms an important aspect of decision support systems. Visualizing large hierarchical datasets poses new challenges, e.g., how to view a large tree in real time with limited memory resources?, how to support zoom-in and zoom-out on the data tree?, how to support context-directed navigation?, or how to support XPath-based slice-and-dice?

5. **Integrating with a traditional OLAP system:** It is imperative that any XML analysis system should be integrated with a traditional OLAP system. The base XML data can be stored either in a relational database or as text documents. The former approach (ROLAP) allows easier integration with a traditional OLAP processor, while the latter, which can be termed XOLAP, allows flexibility of using existing XML processing languages (e.g., XQuery or XSLT) directly.

4. RELATED WORK

Over the years, Online Analytical Processing (OLAP) has been studied extensively. We report here a few related efforts. Vassiliadis and Sellis [25] present a survey of logical models for OLAP databases. Gray et al. [8] first introduced the OLAP `CUBE` operator. Chaudhuri and Dayal [5] present an overview of relevant concepts in data warehousing and decision support systems. Most database vendors support OLAP in their database systems. The OLAP Report [21] presents a detailed study of current industrial OLAP offerings. Current work in using XML for OLAP applications involves using XML for representing external data. To the best of our knowledge, no one has investigated exploiting XML's tree model for analytical purposes. Recently, Pedersen et al. have been exploring the integration of XML data with the traditional OLAP processing [20]. Jensen et al. describe how to specify multi-dimensional OLAP cubes over source XML data [15].

Recently, several researchers have proposed extensions to relational databases for supporting complex OLAP functionalities. Hurtado and Mendelzon [12] and Jagadish et al. [14] have investigated OLAP processing over heterogeneous hierarchies defined over relational data. Chaudhuri et al. [4] and Babcock et al. [1] have studied approximate query processing in the context of aggregation queries. Barbara and Sullivan have proposed Quasi-Cubes, for computing approximate answers in multidimensional cubes [2]. These approaches use approximation to reduce computation time over *precise* data. In our case, the source XML data is inherently imprecise. Lerner and Shasha recently proposed extensions to SQL for supporting order-dependent queries (AQuery) [16]. Carmel et al. have investigated approximate searching of XML documents using structural templates (called XML

fragments) [3]. Navarro and Baeza-Yates have proposed a model to query documents by their content and structure [18].

5. CONCLUSIONS AND FUTURE WORK

In this paper, we investigated various issues in analyzing XML documents. We showed that the traditional multidimensional OLAP model is unsuitable for XML analysis due to its weakness in representing semantically-rich, semi-structured XML data. We proposed a new logical model for XML analysis based on the abstract tree-structured XML representation. We demonstrated the practicality of the new model model by proposing new structural aggregation extensions for XQuery.

We are developing a prototype XML analysis engine to investigate various tradeoffs in data mapping and query translation schemes. In particular, we are evaluating relational-based and XML-based backend engines for functionality, space usage, and execution costs.

6. REFERENCES

[1] B. Babcock, S. Chaudhuri, and G. Das. Dynamic sample selection for approximate query processing. In *Proceedings of the 2003 ACM SIGMOD international conference on on Management of data*, pages 539–550. ACM Press, 2003.

[2] D. Barbara and M. Sullivan. Quasi-Cubes: Exploiting Approximations in Multidimensional Databases. *ACM SIGMOD Record*, 26(3):12–17, 1997.

[3] D. Carmel, Y. S. Maarek, M. Mandelbrod, Y. Mass, and A. Soffer. Searching XML documents via XML fragments. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 151–158. ACM Press, 2003.

[4] S. Chaudhuri, G. Das, and V. Narasayya. A robust, optimization-based approach for approximate answering of aggregate queries. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, pages 295–306. ACM Press, 2001.

[5] S. Chaudhuri and U. Dayal. An Overview of Data Warehousing and OLAP Technology. *Data Mining and Knowledge Discovery*, 26(1):65–74, 1997.

[6] Z. Chen, H. V. Jagadish, L. V. S. Lakshmanan, and S. Paparizos. From Tree Patterns to Generalized Tree Patterns: On Efficient Evaluation of XQuery. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB)*, pages 237–248, September 2003.

[7] World Wide Web Consortium. W3C Architecture Domain: XML. `www.w3c.org/xml`. Online Documents.

[8] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals. *Data Mining and Knowledge Discovery*, 1(1):29–53, March 1997.

[9] Moving Pictures Experts Group. MPEG Standards. www.chiariglione.org/mpeg.

[10] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram. XRANK: Ranked keyword search over XML documents. In *Proceedings of the 2003 ACM SIGMOD international conference on on Management of data*, pages 16–27. ACM Press, 2003.

[11] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, January 1997.

[12] C. A. Hurtado and A. O. Mendelzon. Reasoning about Summarizability in Heterogeneous Multidimensional Schemas. In *Proceedings of the International Conference on Database Theory*, pages 375–389, 2001.

[13] N. Huyn. Data Analysis and Mining in the Life Sciences. *ACM SIGMOD Record*, 30(3):76–85, 2001.

[14] H. V. Jagadish, L. V. S. Lakshmanan, and D. Srivastava. What can Hierarchies do for Data Warehouses? In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, pages 530–541, September 1999.

[15] M. R. Jensen, T. H. Moller, and T. B. Pedersen. Specifying OLAP Cubes on XML Data. In *Proceedings of the 13th International Conference on Scientific and Statistical Database Management*, pages 18–20, July 2001.

[16] A. Lerner and D. Shasha. Aquery: Query language for ordered data, optimization techniques, and experiments. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB)*, pages 345–356, September 2003.

[17] A. Marian and J. Simeon. Projecting XML Documents. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB)*, pages 213–224, September 2003.

[18] G. Navarro and R. Baeza-Yates. Proximal nodes: a model to query document databases by content and structure. *ACM Trans. Inf. Syst.*, 15(4):400–435, 1997.

[19] S. Paparizos, S. Al-Khalifa, H. V. Jagadish, L. V. S. Lakshmanan, A. Nierman, D. Srivastava, and Y. Wu. Grouping in XML. In *EDBT Workshops 2002*, pages 128–147, 2002.

[20] D. Pedersen, K. Riis, and T. B. Pedersen. Query Optimization for OLAP-XML Federations. In *Proceedings of DOLAP 2002, ACM Fifth International Workshop on Data Warehousing and OLAP*, pages 57–64, November 2002.

[21] N. Pendse. The OLAP Report. Online Document. www.olapreport.com.

[22] E. Pourabbas and M. Rafanelli. Hierarchies and Relative Operators in the OLAP Environment. *ACM SIGMOD Record*, 29(1):33–37, 2000.

[23] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *In Proceedings of IJCAI*, pages 448–453, 1995.

[24] J. Trujillo, S. Lujan-Mora, and I. Song. Applying UML and XML for designing and interchanging information for data warehouses and OLAP. *Journal of Database Management*, 15(1):41–72, 2004.

[25] P. Vassiliadis and T. Sellis. A Survey of Logical Models for OLAP Databases. *ACM SIGMOD Record*, 28(4):64–49, 1999.

On Six Degrees of Separation in DBLP-DB and More

Ergin Elmacioglu
Department of Computer Science and Engineering
The Pennsylvania State University, PA 16802
elmaciog@cse.psu.edu

Dongwon Lee
School of Information Sciences and Technology
The Pennsylvania State University, PA 16802
dongwon@psu.edu

ABSTRACT

An extensive bibliometric study on the db community using the collaboration network constructed from DBLP data is presented. Among many, we have found that (1) the average distance of all db scholars in the network has been stabilized to about 6 for the last 15 years, coinciding with the so-called six degrees of separation phenomenon; (2) In sync with Lotka's law on the frequency of publications, the db community also shows that a few number of scholars publish a large number of papers, while the majority of authors publish a small number of papers (i.e., following the power-law with exponent about -2); and (3) with the increasing demand to publish more, scholars collaborate more often than before (i.e., 3.93 collaborators per scholar and with steadily increasing clustering coefficients).

1. INTRODUCTION

Social network analysis is an active research field in the social sciences where researchers try to understand social influence and groupings of a set of people or groups. Its origin is in general believed to be due to S. Milgram [7] in 1967 who identified the so-called "*six degrees of separation*" phenomenon based on an experiment – any two people in the United States are connected through about 6 intermediate acquaintances, implying we live in a rather small-world. Since then, sociologists and psychologists have found evidence for a wide range of small-world phenomena arising in other social and physical networks (e.g., power grids, airline time tables, food chain, World-Wide Web, Erdos number). Inspired by some of the recent attempts to apply social network analysis to our own db community [8, 11], in this paper, we analyze the collaboration network made of by database researchers, and see if there exists any interesting patterns underlying the db community and their publication behavior.

2. SETUP

Since DBLP [3] is a high-quality citation digital library that has a near complete coverage on database literature, we chose to use DBLP as the data set for our analysis of the db community. In particular, we examined citation data in DBLP from 1968 to 2003, and hand-picked publication venues (19 journals and 81 conferences, symposiums and workshops) that we believed to be closely-related to the db community (shown in Table 1). Note that we intentionally excluded venues related to Information Retrieval or Digital Library, but included ones related to Data Mining. We also did not include venues that have some database papers as well as papers from many other fields (e.g., J. ACM, Comm. ACM, and WWW). Hereafter, we will refer to this collection as **DBLP-DB**. DBLP-DB contained 32,689 authors and 38,773 papers.

Table1: The list of publication venues in DBLP-DB from 1968 to 2003.

Conferences/Symposiums/Workshops (81)
ADB, ADBIS, ADBT, ADC, ARTDB, Berkeley Workshop, BNCOD, CDB, CIDR, CIKM, CISM, CISMOD, COMAD, COODBSE, CoopIS, DAISD, DANTE, DASFAA, DaWaK, DBPL, DBSEC, DDB, DDW, DEXA, DIWeB, DMDW, DMKD, DNIS, DOLAP, DOOD, DPDS, DS, EDBT, EDS, EFIS/EFDBS, ER, EWDW, FODO, FoIKS, FQAS, Future Databases, GIS, HPTS, IADT, ICDE, ICDM, ICDT, ICOD, IDA, IDC(W), IDEAL, IDEAS, IDS, IGIS, IWDM, IW-MMDBMS, JCDKB, KDD, KR, KRDB, LID, MDA/MDM, MFDBS, MLDM, MSS, NLDB, OODBS, OOIS, PAKDD, PKDD, PODS, RIDE, RIDS, RTDB, SBBD, SDM-SIAM, Semantics in Databases, SIGMOD, SSD, SSDBM, SWDB, TDB, TSDM, UIDIS, VDB, VLDB, WebDB, WIDM, WISE, XP, XSym
Journals (19)
ACM TODS, ACM TOIS, DKE, Data Base, DMKD, DPD, IEEE Data Eng. Bulletin, IEEE TKDE, Info. Processing and Management, Info. Processing Letters, Info. Sciences, Info. Systems, J. of Cooperative Info. Systems, J. of Database Management, JIIS, KAIS, SIGKDD Explorations, SIGMOD Record, VLDB J.

The *collaboration network* (or graph) consists of nodes of authors and edges connecting any two authors if they co-authored one or more papers. Note that DBLP itself does not have a notion of "unique key" such as DOI (Digital Object Identifier). Instead, DBLP depends on the name of authors to distinguish them. Therefore, the classical *name authority control problem* may arise (i.e., same author with various spellings or different authors with the same spelling). We could minimize this problem as Newman [10] did by conducting two experiments – one with full names (``John Doe'') and the other with the first initial of the first name followed by the last name (``J. Doe'') – and use these as the upper and lower bounds. However since it is known that their effect on the quality of citation analysis

is negligible [10], we did not do any special pre-processing to handle such cases. For the visualization of our network analysis, we used Pajek [2] and NetDraw [9].

3. STATISTICS ABOUT AUTHORS

First, we do various statistical analysis related to authors of papers. Figure 1 shows the number of "new authors" (ones who publish a paper in DBLP-DB for the first time) as a function of year. As shown, the db community steadily grows each year by the addition of new authors (at the 10% rate after 1985). In 2003 alone, there are more than 3,000 new authors who joined the db community, some of whom are novice graduate students or veteran scholars from similar fields.

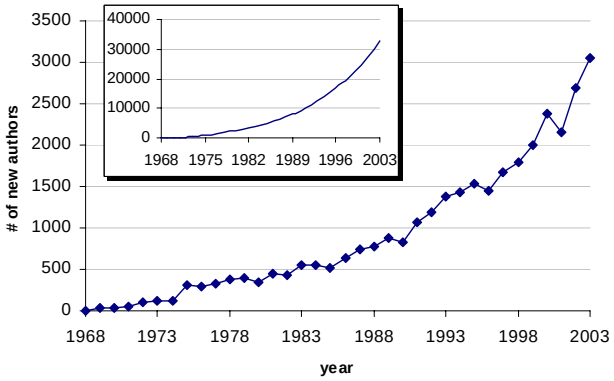


Figure 1. Number of new authors who joined the network each year. Inset: cumulative number of authors up to the year indicated.

Next, we examine how active the db community is. The "active authors" are those who publish at least one paper in a given year. Figure 2 illustrates the number of active authors each year. For instance, in 2003, there are only about 6,000 active authors (out of 32,689 authors). Interestingly, almost half of the active authors in any given year are "new authors". Moreover, new authors are steadily contributing to about 60% of papers each year (since 1982).

Those could be new graduate students entering into the community for the first time by collaborating with their advisors. The remaining 40% of the publications is contributed purely by the existing authors or co-authors, which increases the density of the collaboration network.

Figure 3 illustrates the average number of papers per author for a given year (i.e., # of papers / # of authors). After 1980, the value starts to stabilize around 0.3 paper per author ratio, implying that the productivity rate of the db community as a whole remains intact over time. This makes sense since only small fractions of the community (about 18%-20%) are active each year and they can publish only a limited number of papers.

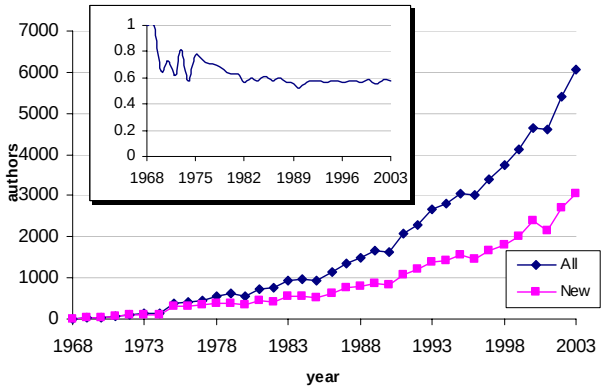


Figure 2. Number active authors each year. Inset: percentage of the papers published by new authors each year

Lotka's Law describes the frequency of publications by authors by "the number of authors making *n* contributions is about $1/n^2$ of those making one; and the proportion of all contributors, that make a single contribution, is about 60 percent" [6]. He showed that such a distribution follows a power law with an exponent approximately -2. Figure 4 shows the distribution of numbers of papers per author on log-log scales for our database.

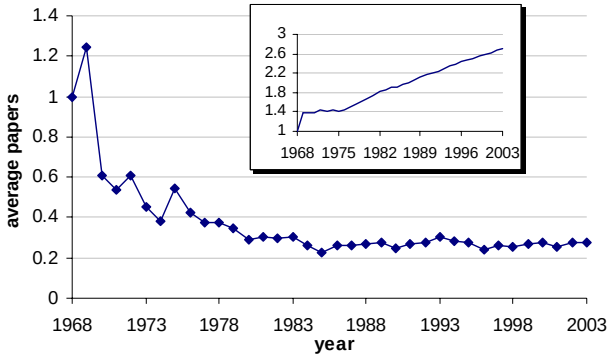


Figure 3. Average number papers per author each year. Inset: cumulative number of papers/author up the year indicated.

Consistent with Lotka's Law, 20849 authors (64%) have only one paper whereas a small number of authors publish a large number of papers (the fat tail on the right hand side indicates this). In fact, in DBLP-DB, there are only 18 authors who published more than 100 papers. Furthermore, the exponent of the graph is -2.15 which is very close to that found by Lotka. Top-10 authors with the highest number of publications in DBLP-DB are shown in Table 4.

Now, we examine the number of collaborators of authors. Figure 5 illustrates the results of this measure. The average number of collaborators per author (from 1968 to 2003) is 3.93 and tends to increase steadily. Compared to

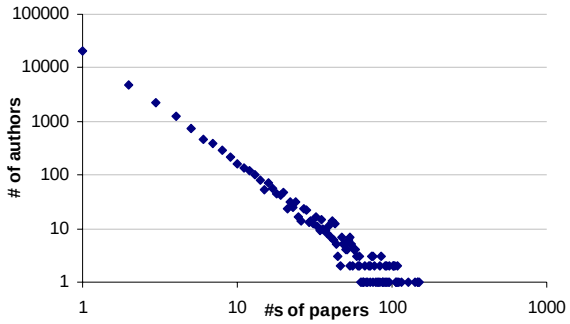


Figure 4. Distribution of numbers of papers per author, as of 2003.

other scientific communities that involves large-scale experimentation (e.g., high-energy physics), this average number of collaborators is rather small.

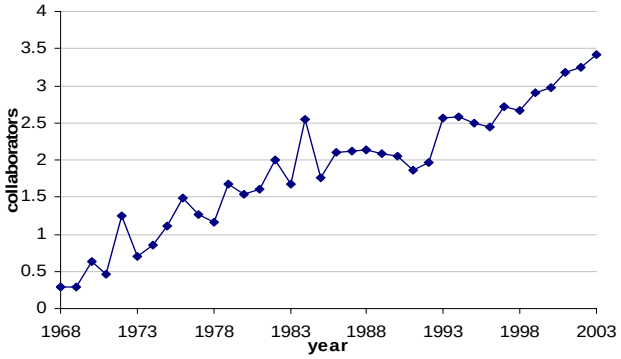


Figure 5. Average number of collaborators per author each year

The steady increase of the average number of collaborators can be hypothesized as follows: (1) the so-called "Publish or Perish" pressure drives scholars to seek more effective ways to increase the number of publications such as collaborative research; and (2) the rapid development and deployment of new communication mediums (email, messenger, web board, or web camera) makes remote collaborations much easier than before.

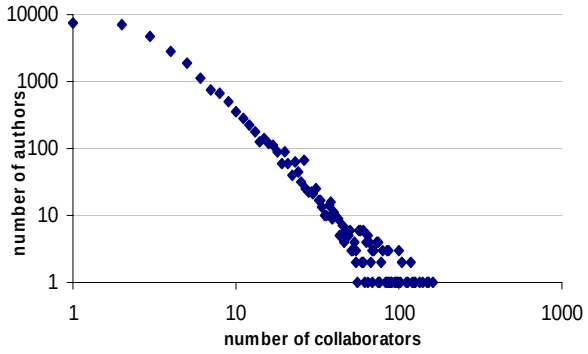


Figure 6. Distribution of number of collaborators per author, as of 2003.

The distribution of number of collaborators per author is shown in Figure 6. It also exhibits the power-law tail with exponent -2.3. The second column of Table 4 shows the authors with the largest number of collaborators. Many of these authors are ranked high in the centrality measures described in section 8.

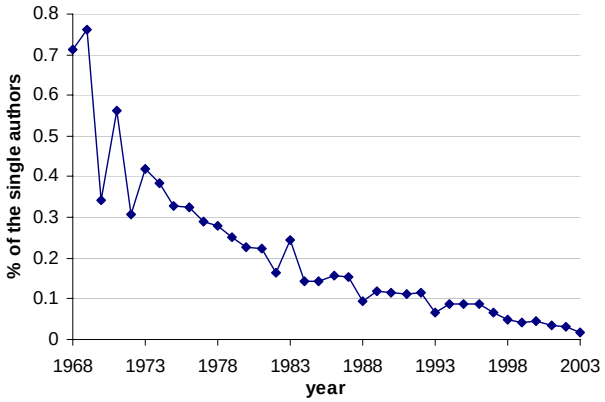


Figure 7. Percentage of the active single authors per year

Finally, we analyzed if there are authors with no collaborators in DBLP-DB. There are 3,073 such authors which constitute 9.4% of the db community. However, 84% of all these single authors have only one paper. There are also a few scholars who have written more than 10 papers just by themselves without any collaboration. One particular author, "*Levent V. Orman*", has written 14 papers alone, shown in Table 2. However, due to the pressure for increased productivity, such single authors are diminishing. Figure 7 shows the percentage of single authors to active authors each year, clearly decreasing over time, and more interestingly, it exhibits the symmetric pattern from the increasing pattern of the number of collaborators in Figure5.

Table 2: Publications of the most productive author with no collaborators.

Year	Title
1982	A familial model of data for a multilevel schema framework.
1984	A Multilevel Design Arct. for decision support systems
1984	Nested set languages for functional databases.
1985	Functions in Information Systems.
1985	Design criteria for functional data bases.
1986	Functional data model design.
1986	Redundancy in functional databases.
1991	Complexity of database languages.
1991	A visual data model.
1993	Knowledge Management by Example.
1996	Queries = Examples + Counterexamples.
1998	Differential Relational Calculus for Integrity Maintenance.
1998	Storage and Retrieval of Database Constraints.
2001	Transaction Repair for Integrity Enforcement.

4. STATISTICS ABOUT PAPERS

In 2003 alone, there are more than 3,000 db papers published (Figure 8), and at the end of 2004, there will be roughly 10% more db papers published. This is largely due to the increased number of authors. Since the average number of papers per author is fixed to 0.3 per year (Figure 3), the number of db papers is approximately in sync with the number of db authors, especially active ones (Figure 2).

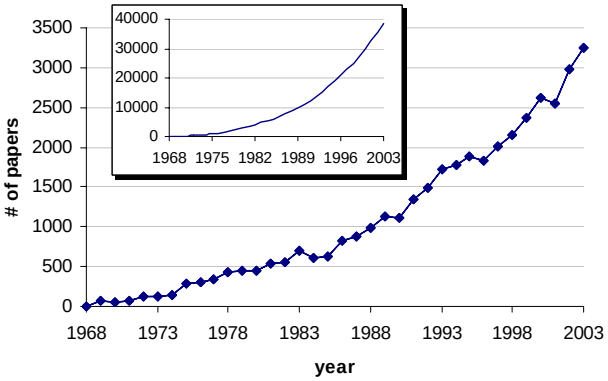


Figure 8. Number of papers per year. Inset: cumulative number of papers published up to the year indicated.

The average number of authors per paper in the db community tends to increase each year, yielding almost 2.8 co-authors per paper as of 2003 (Figure 9). Although there are a significant number of papers with only a single author, there are more papers written by two authors (13557 papers). This can be seen in the distribution of this measure in Figure 10, which has a power law tail with an exponent -3.68. The largest number of authors on a single paper is 27. The figure clearly shows that there is an increasing tendency for collaboration among authors which also causes papers to have more co-authors.

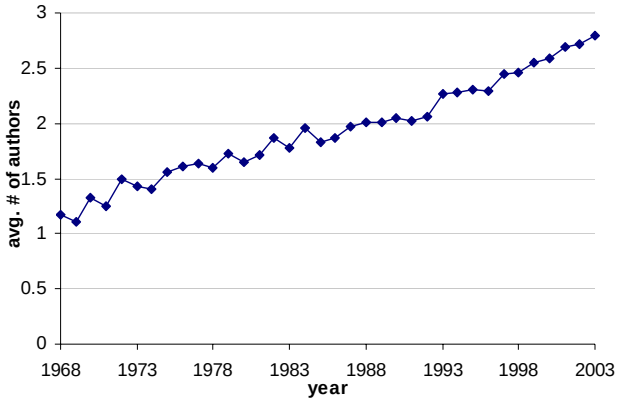


Figure 9. Average number of authors per paper each year

Next, we looked at how publication venues in DBLP-DB are inter-related to each other using co-authorship information. By examining the pattern where the db scholars publish their papers, one can see, for instance, which publications venues have a similar theme or taste. Figure 11 is a graph where (1) a node is a publication venue in DBLP-DB, the size of which is proportional to the number of papers in it, and (2) an edge between venues X and Y reflects the Jaccard distance, $|A \cap B| / |A \cup B|$, where A and B are author sets of venues X and Y. The higher the Jaccard distance is (i.e., more authors are common between venues), the thicker the edge becomes. Table 3 lists top-10 pairs of database publication outlets.

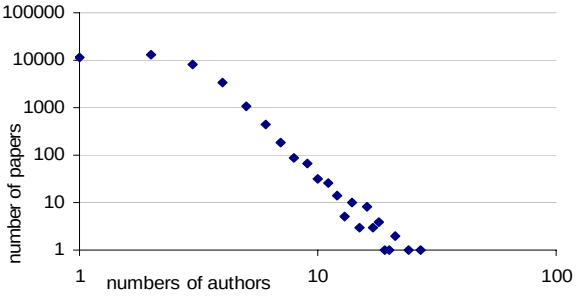


Figure 10. Distribution of number of authors per paper (2003).

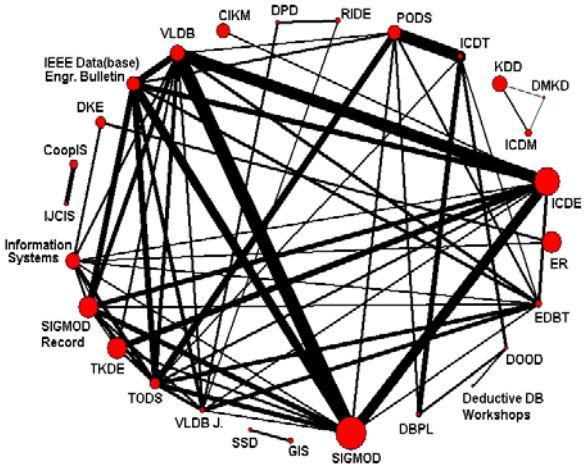


Figure 11: Venue relations (only venues with at least 100 papers and edges with at least 0.1 Jaccard distance are shown. The size of a node is proportional to the number of papers in the venue, while the thickness of edge is proportional to the overlap of authors between venues).

Table 3: Top-10 pairs of venues with the highest Jaccard distances

Similar venue pair	Distance
SIGMOD - VLDB	0.2229
ICDT - PODS	0.1971
ICDE - VLDB	0.1948
ICDE - SIGMOD	0.1817
SIGMOD - IEEE Data Eng. Bulletin	0.1736
VLDB - IEEE Data Eng. Bulletin	0.1559
PODS - TODS	0.1557
SIGMOD Rec. - IEEE Data Eng. Bulletin	0.1502
ICDE - TKDE	0.1498
TODS - IEEE Data Eng. Bulletin	0.1441

Table 4: The top-10 db authors with the highest number of papers, number of co-authors, and clustering coefficients (Clustering coeff. I with 60 as the threshold number of co-authors and clustering coeff. II with 60 as the threshold number of papers). For the number of papers and co-authors columns, values in the parenthesis are for the whole DBLP data set.

Number of papers		Number of co-authors		Clustering coeff. (I)		Clustering coeff. (II)	
151 (293)	H. Garcia-Molina	163 (192)	M. Stonebraker	0.1627	J. F. Roddick (60)	0.2019	A. Segev (68)
147 (201)	J. Han	152 (181)	M.J. Carey	0.1534	M. L. Brodie (65)	0.1498	D. B. Lomet (102)
146 (207)	M. Stonebraker	150 (206)	D. Maier	0.1441	R. Rastogi (60)	0.1441	R. Rastogi (74)
142 (326)	P.S. Yu	139 (200)	H. Garcia-Molina	0.1384	T. Milo (60)	0.1275	Y. Manolopoulos (62)
128 (293)	E. Bertino	134 (165)	D.J. DeWitt	0.1378	B. G. Lindsay (85)	0.1240	J. Widom (90)
115 (150)	R. Agrawal	125 (155)	J. Han	0.1336	D. Florescu (63)	0.1206	J. D. Ullman (82)
111 (163)	E. Rundensteiner	124 (222)	E. Bertino	0.1321	S. Sudarshan (68)	0.1173	P. A. Bernstein (72)
109 (192)	D. Agrawal	120 (178)	C. Faloutsos	0.1279	J. Hellerstein (73)	0.1159	M. Lenzerini (64)
109 (153)	M.J. Carey	119 (180)	G. Wiederhold	0.1271	H. Pirahesh (84)	0.1117	C.S. Jensen (79)
108 (147)	D.J. DeWitt	119 (148)	U. Dayal	0.1240	J. Widom (73)	0.1112	J.F. Naughton (89)

5. THE GIANT COMPONENT

The *giant component* of a graph is the largest subset of interconnected nodes in the graph. The rest of the nodes usually form much smaller components, typically of size *O(log n)*, where *n* is the total number of nodes [10]. In order to determine if such a component exists in our collaboration graph, we measured the relative size of the largest component which is simply the ratio of the nodes in the component to the all nodes in the graph. The growth of the giant component of our graph is shown in Figure 12 as a function of time.

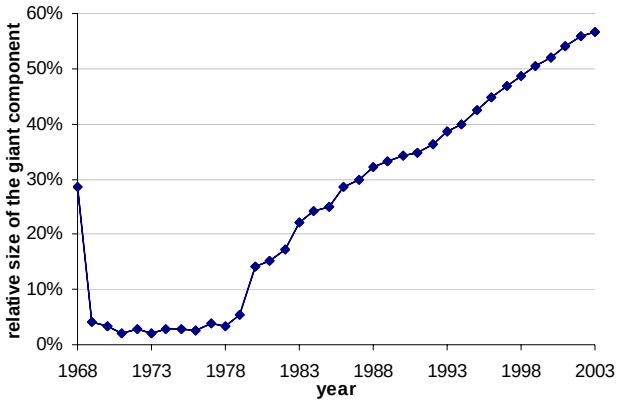


Figure 12: Relative size of the giant component, determined for the cumulative data up to the year indicated.

In the initial years, the size of the giant component of the graph is much smaller compared to the total number of nodes available in the graph, covering only about 5% of the whole graph although new authors keep joining to the db community. Yet, those authors help cluster other large components in the graph. In 1980, those clusters started to form larger components. After the size of the giant component exceeds 30%, it constantly increases up to the end of the period analyzed.

Although tendency for more collaboration in the community helps the smaller components to be connected to the giant component, the main increase stems from the new authors. In recent years the db community grows 10% by the addition of new authors, who tend to collaborate with existing authors (e.g., graduate students collaborating with their advisor) to write a paper instead of making a contribution alone. (Figures 1 and 7: the number of new authors, the number of single authors).

As of 2003, the size of the giant component is 18542, 57% of the whole db community. This is a rather low figure since the db community is expected to be a tight one. In addition, the second largest component is much smaller; it includes only 51 authors, who work on very particular subjects and publish mostly in 'Information Sciences' journal. The collaboration graph has 424 "isolated" components with 5-9 authors and 2892 components with 2-4 authors.

6. CLUSTERING COEFFICIENTS

Given a node *v*, the *neighborhood* of *v*, *N(v)*, is a subgraph that consists of the nodes adjacent to the node *v*. Furthermore, let us denote the edges and nodes in *N(v)* by *E(N(v))* and *K(N(v))*, respectively. Then, the clustering coefficient of *v*, $\gamma(v)$, is:

$$\gamma(v) = \frac{|E(N(v))|}{|E\max(N(v))|}$$

When the neighborhood is fully-connected (i.e., clique), it has $|E\max(N(v))| = \frac{|K(N(v))|(|K(N(v))|-1)}{2}$ edges. Therefore, the clustering coefficient measures how many edges actually occur compared to the fully-connected case [14]. The clustering coefficient of a graph *G*, $\gamma(G)$, is the average clustering coefficients of all nodes in *G*.

The clustering coefficient can be also viewed as "transitivity" which describes the interactions among trios of nodes in a network [10] – the degree to which a scholar's collaborators have collaborated with each other. In co-authorship networks, this measure implies how much authors are willing to collaborate with each other. The clustering coefficient of the giant component in the db co-

authors graph is shown in Figure 13 as a function of year (shown from 1972 when the network started to form a relatively giant component).

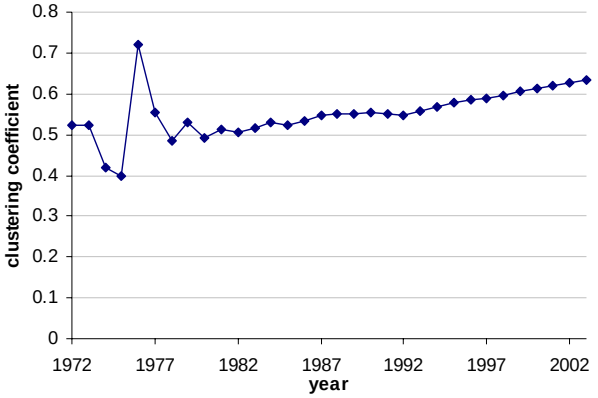


Figure 13: Clustering coefficient of the giant component, determined for the cumulative data up to the year indicated.

Over the years, the clustering coefficients tend to increase steadily, reaching 0.63 in 2003. This rather high value of the clustering coefficient is expected since DBLP-DB is after all a tight community of people working on databases only. Moreover, the increasing clustering coefficient is in sync with the tendency for more collaboration in recent years. The last two columns of Table 4 lists the top-10 authors with the highest clustering coefficients for different cases; the first ranking only considers authors with at least 60 collaborators while the second one lists authors with more than 60 papers (numbers in parenthesis).

7. GEODESIC

In a co-authorship network, two authors could know each other through their collaborators. In other words, there could be several interaction paths between two of them not directly but through a number of the other authors in the network. The path(s) with the minimum number of edges between any given pair of authors in the network is called shortest path or *geodesic* of the pair. Then the average distance in a network is the average of all pair-wise geodesics of authors in the network. Social networks often have small average distances compared to the number of nodes in the networks, which is first described by Milgram [7] and now referred to as "small-world effect". Figure 14 shows the evolution of the averages distance in the db community over the given period.

After the initial fluctuations, the average distance reaches to its maximum value, 8, in 1983. This seems natural since in the beginning of the time period analyzed, the growth of the community is rapid (i.e. each year, an increase between 40% - 80 % from the previous year). New authors are probably responsible for making the community expansion since the collaboration during that period was not very active. After 1983, however, it starts to decrease and eventually stabilizes around 6 for the last 15 years. Interestingly Milgram also found an average distance of six hops for his social network experiment. The relatively low value, 6, is probably a good sign since scientific discoveries can be disseminated rather fast [10].

The *diameter* of a graph, the maximum of the pair-wise distances in the giant component, of the db community is 20 as of 2003: "*A.Baczko - F.Seredynski - P.Bouvry - J.Blazewicz - P.Dell'Olmo - H.Kellerer - A.Caprara - D.Maio - P.Tiberio - S.J.Finkelstein - I.S.Mumick - O.Shmueli - F.Gavril - J.Urrutia - V.Estivill-Castro - L.Brankovic - M.Miller - P.D.Manuel - J.AlGhamdi - M.Sarfraz - K.Salah*".

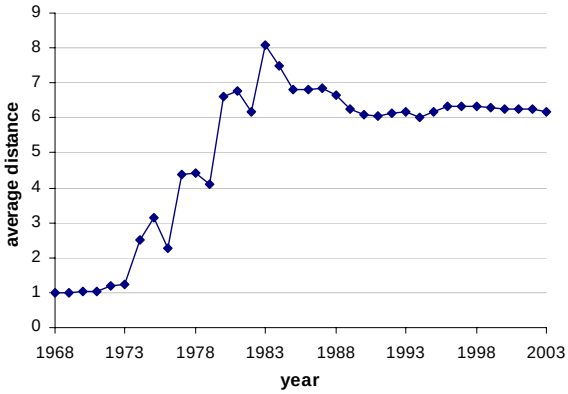


Figure 14: Average distance of the network, computed on the cumulative data up to the year indicated.

8. CENTRALITY

One of the interesting aspects of co-authorship network is to identify the most "central" scholars in the network. Authors who are the most prominent in the community are often (certainly not always) located in the strategic locations of the co-authorship network, which may allow them: (1) to communicate directly with many other authors, (2) to be close to many other authors, or (3) to be as intermediary in the interactions of many other pair of authors. There are several methods which aim to quantify authors' locations in [12]. For our work, we used the *closeness* and *betweenness* measures to quantify the prominent db scholars.

Table 5: Authors with the highest betweenness and closeness scores

Betweenness		Closeness	
0.054620	G. Wiederhold	0.268216	U. Dayal
0.048295	U. Dayal	0.262397	G. Wiederhold
0.045001	J. Han	0.256737	R. T. Snodgrass
0.038067	Y. Kambayashi	0.256555	D. Maier
0.030376	E. Bertino	0.256332	K. A. Ross
0.029406	H. Lu	0.256261	H. Garcia-Molina
0.027622	H.-J. Schek	0.256247	S. Ceri
0.026841	M. Jarke	0.256003	H.-J. Schek
0.026526	R. Agrawal	0.254401	M. J. Carey
0.026504	S. Ceri	0.253945	M. Stonebraker

8.1 Closeness Centrality

The *closeness* can be defined as how close an author is on average to all other authors. Authors with low closeness values could be viewed as those who can access new information quicker than others and similarly, information originating from those authors can be disseminated to others quicker [10]. Formally, the closeness of a node *v* in a connected graph G is defined as follows:

$$C(v) = \frac{n-1}{\sum_{v,w \in G} d(v,w)}$$

where *d(v,w)* is the pair-wise geodesic and *n* is the number of all nodes reachable from *v* in *G*. That means that, it is 1 over the average of the shortest paths from *v* to all other nodes in *G*.

The second column of Table 5 lists the top-10 individuals with the highest closeness scores. Furthermore, all those 10 scholars are closely connected to each other through collaborations (Figure 15), which could be viewed as a "core" component of the db community.

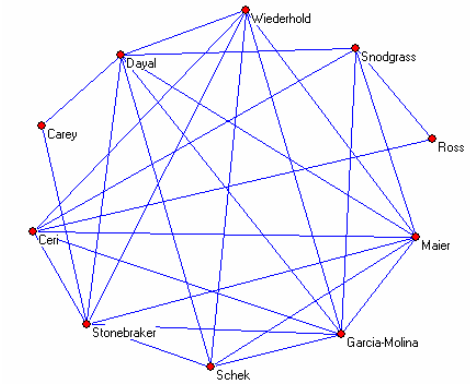


Figure 15: Connectivity of top10 authors with the highest closeness

8.2 Betweenness Centrality

Sometimes the interactions between any two non-directly connected authors (i.e., who never collaborated before) might depend on the authors who connect them through their shortest path(s). These authors potentially play an important role in the network by controlling the flow of interactions. Hence the authors who lie between most of the shortest paths of the pairs of authors could be viewed as the central people in the community. This notion, known as the *betweenness* of a node *v*, *B(v)*, measures the number of geodesics between pairs of nodes passing through *v*, and formally defined as follows [5]:

$$B(v) = \sum_{v,w,x \in G} \frac{d(w,x;v)}{d(w,x)}$$

where *d(w,x)* is a geodesic between *w* and *x*, and *d(w,x;v)* is a geodesic between *w* and *x* passing through *v*. The equation can be also interpreted as the sum of all probabilities a shortest path between each pair of nodes *w* and *x* passes through node *v*. The first column of Table 5 shows the top-10 authors with the highest betweenness scores.

8.3 Weighted Measures

So far, we have not differentiated whether or not two authors have single or multiple collaborations – as long as there is a single co-authored paper, two authors are linked in the collaboration graph. People have recognized this and tried to incorporate weight such that the more collaboration two authors have, the stronger link exists between them [1, 10]. Newman in [10] defines such a collaboration network as follows:

$$\sum_{j(\neq i)} w_{ij} = \sum_{k} \sum_{j(\neq i)} \frac{\partial_i^k \partial_j^k}{n_k - 1}$$

In this model, w_{ij} represent the collaboration weight between two authors *i* and *j*. ∂i^k is 1 if author *i* is a co-author of paper *k* and n_k is the total number of co-authors of paper *k*. That is, if authors *i* and *j* co-authored a paper *k*, each one should divide his time equally between the other *n-1* co-authors. Then, the sum of all collaboration weights w_{ij} between two authors defines the total strength of that tie.

Table 6: Closeness and betweenness for the weighted collaboration graph.

Weighted closeness			
score	**name**	**# of papers**	**# of co-authors**
0.19262862	H. V. Jagadish	106	102
0.19192969	Divesh Srivastava	85	91
0.19161295	Umeshwar Dayal	103	119
0.19156657	Raghu Ramakrishnan	90	76
0.19137470	Rakesh Agrawal	115	94
0.19029398	Surajit Chaudhuri	83	67
0.19005407	Jiawei Han	147	125
0.19004566	L.V. S. Lakshmanan	66	58
0.18976747	Hector Garcia-Molina	151	139
0.18934639	Qiming Chen	35	17
Weighted betweenness			
score	**name**	**# of papers**	**# of co-authors**
0.48422084	Umeshwar Dayal	103	119
0.46723451	Jiawei Han	147	125
0.33170557	H. V. Jagadish	106	102
0.28326387	Yahiko Kambayashi	105	99
0.28119593	Elisa Bertino	128	124
0.26476805	Hongjun Lu	97	99
0.26315179	Hector Garcia-Molina	151	139
0.25839473	Raghu Ramakrishnan	90	76
0.23657271	Surajit Chaudhuri	83	67
0.22207486	Rakesh Agrawal	115	94

We regenerated our network according to this model and considered the distance value between two authors as the inverse of their collaboration weight. The new weighted

rankings of closeness and betweenness measures can be seen in Table 6 for this network. Interestingly, authors who tend to collaborate often with a small number of people only are ranked high (e.g., scholars in research labs or a small set of collaborators).

8.4 Caveats

Although the bibliometric analysis described above is meaningful in many cases, it is worthwhile to point out that these are not without problems (Dickson even argues "measuring scientific productivity by tracking the publication record of researchers is widely acknowledged to be hazardous and imperfect" [4]). One such problem visibly occurs in our study as well.

Note that since the raw data set, DBLP-DB, contains only papers in the pre-selected database-related publication venues, all measures tend to favor authors whose expertise is focused on the database field alone (penalizing scholars whose expertise is diverse and inter-disciplinary). In addition, the measures such as closeness or betweenness cannot identify scholars who made a critical contribution to the community with only a small number of publications or collaborators.

For instance, consider the following four distinguished database scholars: (1) *E. F. Codd*: the inventor of the relational data model, (2) *Jim Gray*: Turing award winner, (3) *Peter P. Chen*: the inventor of the ER model, and (4) *Jeffrey D. Ullman*: a renowned computer scientist at Stanford University. As shown in Table 7, Chen and Codd are ranked very low (e.g., 347-th and 3638-th in their betweenness ranks) in the betweenness and closeness ranks due to their small number of publications. On the other hand, although both Ullman and Gray have a substantial number of publications (233 for Ullman and 121 for Gray), since their contributions in general are very diverse, ranging from algorithms and automata to databases and programming languages, and to even physics, only about 1/3 of them are included in DBLP-DB.

Table 7: Statistics of four authors for DBLP-DB. Figures in the parenthesis are for the entire DBLP data set.

	# of papers	# of co-authors	Betwn. rank	Close. rank
Ullman	82(233)	87(131)	67	17
Chen	33(43)	24(27)	347	837
Gray	43(121)	97(179)	83	29
Codd	23(47)	5(15)	3638	7829

9. CONCLUSIONS

In this paper we analyzed the collaboration network of scientists who publish in the database area. We presented a large number of statistics including how number of papers per author, authors per paper and number of collaborators change over the time period analyzed. We found that distributions of these statistics follow a scale-free power law distribution. We also looked at the evolution of other properties including average distance, clustering co-efficient and size of the giant component. The results imply that the db community seems to be a "small-world" by having a very small average distance between authors and being highly-clustered. These results may be helpful for further efforts on the db community such as modeling the network growth that may allow us to predict the approximate network behavior at any given time.

10. REFERENCES

[1] Barabasi, A. L. et al. *Evolution of the social network of scientific collaborations.* Physica A 311, 590-614 (2002).

[2] Batagelj, V. and Mrvar, A. *Pajek – A program for large network analysis.* Connections 21(2), 47-57 (1998). http://vlado.fmf.uni-lj.si/pub/networks/pajek/

[3] DBLP, *Computer Science Bibliography.* http://www.informatik.uni-trier.de/~ley/

[4] Dickson, D. *The Virtue of Science by Numbers.* SciDev.Net (August 2003). http://www.scidev.net/Editorials/index.cfm?fuseaction=readEditorials&itemid=83&language=1

[5] Freeman, L. C. *A Set of Measures of Centrality Based on Betweenness.* Sociometry 40, 35-41 (1977).

[6] Lotka, A. J. *The frequency distribution of scientific production.* J. Walsh. Acad. Sci. 16, 317-323 (1926).

[7] Milgram, S. *The small world problem.* Psychology Today 2, 60-70 (1967).

[8] Nascimento, M. A., Sander, J. and Pound J. *Analysis of SIGMOD's co-authorship graph.* SIGMOD Record 32(3): 8-10 (2003).

[9] NetDraw, *Network Visualization Software.* http://www.analytictech.com/netdraw.htm

[10] Newman, M. E. J. *Who is the best connected scientist? A study of scientific coauthorship networks.* Phys. Rev. E64 016131(2001); Phys. Rev. E64 016132 (2001).

[11] Snodgrass, R. T. *Journal relevance.* SIGMOD Record 32(3): 11-15 (2003).

[12] Wasserman, S. and Faust, K. *Social Network Analysis.* Cambridge University Press, Cambridge (1994).

[13] Watts, D. J. and Strogatz S. H. *Collective dynamics of 'small-world' networks.* Nature 393, 440-442 (1998).

[14] Watts, D. J. *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton University Press, Princeton (1999).

Semantic Characterizations of Navigational XPath

Maarten Marx and Maarten de Rijke
Informatics Institute, University of Amsterdam
Kruislaan 403, 1098 SJ, Amsterdam, The Netherlands
{marx,mdr}@science.uva.nl

Abstract

We give semantic characterizations of the expressive power of navigational XPath (a.k.a. Core XPath) in terms of first order logic. XPath can be used to specify sets of nodes and sets of paths in an XML document tree. We consider both uses. For sets of nodes, XPath is equally expressive as first order logic in two variables. For paths, XPath can be defined using four simple connectives, which together yield the class of first order definable relations which are safe for bisimulation. Furthermore, we give a characterization of the XPath expressible paths in terms of conjunctive queries.

1 Introduction

XPath 1.0 [17] is a variable free language used for selecting nodes from XML documents. XPath plays a crucial role in other XML technologies such as XSLT [21], XQuery [20] and XML schema constraints [19]. The recently proposed XPath 2.0 language [18] is much more expressive. It contains variables which are used in if-then-else, for, and quantified expressions. The available axis relations are the same in both versions of XPath. What is missing at present is a clear characterization of the expressive power of XPath, be it either semantical or with reference to some well established existing (logical) formalism. As far as we know, Benedikt, Fan and Kuper [2] were the first and only to give characterizations, but only for positive fragments of XPath, and without considering the sibling axis relations. Their analysis can rather simply be expanded with the sibling axis, but adding negation asks for a different approach. This paper aims at filling this gap.

Characterizations of the kind we are after are useful in understanding and (re-)designing the language. They are also useful because they allow us to transfer known results and techniques to the world of XPath. Vianu [16] provides several examples to this effect. All characterizations we give with respect to other languages are constructive and given in terms of translations. An important issue in such comparisons is the succinctness of one language with respect to another. We only touch on this briefly.

We use the abstraction to the logical core of XPath 1.0 (called *Core XPath)* developed in [7, 8]. Below we will often speak of XPath instead of Core XPath. Core XPath is interpreted on XML document tree models. The central expression in XPath is the location path

$$\mathsf{axis} :: \mathit{node\_label}[\mathsf{filter}],$$

which, when evaluated at node n, yields an answer set consisting of nodes n' such that the axis relation goes from n to n', the node tag of n' is *node_label*, and the expression filter evaluates to true at n'. Alternatively, $\mathsf{axis} :: \mathit{node\_label}[\mathsf{filter}]$ can be viewed as denoting a binary relation, consisting of all nodes (n, n') which stand in the above relation.

XPath serves two purposes. First and foremost, it is used to select nodes from a document. This use is formalized by the notion of answer set. We study the expressive power of XPath with respect to defining answer sets in Section 3. Our main result is that Core XPath is as expressive as first order logic restricted to two variables in the signature with three binary relations corresponding to the `child`, `descendant` and `following_sibling` axis relations and unary predicates corresponding to the node tags.

The second use of XPath is as a set of binary atoms in more expressive languages with variables such as XQuery. For instance, we might want to select all nodes x satisfying

$$(1)\quad \exists y(x\ \texttt{descendant} :: \texttt{A}\ y \wedge \neg\, x\ \texttt{descendant} :: \texttt{B}/\texttt{descendant} :: *\ y).$$

That is, the set of all points which start a path without B nodes ending in an A node. We study this use in Sections 4 and 5. With respect to the expressive power of the relations expressed by Core XPath we establish the following:

1. The set of relations expressible in Core XPath is closed under intersection but not under complementation.
2. The Core XPath definable relations are exactly those that are definable as unions of conjunctive

queries whose atoms correspond to the XPath axis relations and to XPath's filter expressions.

3. The Core XPath definable relations are exactly those that can be defined from its axis and node-tag tests by composition, union, and taking the counterdomain[1] of a relation.

The paper is organized as follows. The next section defines Core XPath. Sections 3 and 4 are about the expressive power of XPath for selecting sets of nodes, and selecting sets of paths, respectively. Section 5 establishes a minimal set of connectives for XPath.

Related work

The paper most closely related to this work is [2], which characterizes positive XPath without sibling axis as existential positive first order logic. Similar results, stated in terms of conjunctive queries are obtained in [9]. Characterizations in terms of automata models have been given in [3, 15, 13, 14].

Connections with temporal logic have been observed by [7, 12] which sketch an embedding of the forward looking fragment of XPath into CTL. [1] exploits embeddings of subsets of XPath into computation tree logic to enable the use of model checking for query evaluation. [11] defines an extension of XPath in which every first order definable relation can be expressed. Closure under complementation is the distinguishing property of such languages: for expansions of Core XPath, it is equivalent to having full first order expressivity. Several authors have considered extensions far beyond XPath 1.0, trying to capture all of monadic second order logic.

2 Core XPath

[8] proposes a fragment of XPath 1.0 which can be seen as its logical core, but lacks much of the functionality that accounts for little expressive power. In effect, it supports all XPath's axis relations, except for the attribute and namespace axis relations, it allows sequencing and taking unions of path expressions and full booleans in the filter expressions. It is called Core XPath, also referred to as *navigational XPath*. A similar logical abstraction is made in [2]. As the focus of this paper is expressive power, we discuss XPath restricted to its logical core.

For the definition of the XPath language and its semantics, we follow the presentation of XPath in [8]. The expressions obey the standard W3C unabbreviated XPath 1.0 syntax. The semantics is as in [2, 8], in line with the standard XPath semantics [22].

[1]The counterdomain of a binary relation R (notation: $\sim R$) is the set $\{(x, y) \mid x = y \wedge \neg\exists z\, xRz\}$.

Definition 1 The syntax of the Core XPath language is defined by the grammar in Table 1, where "locpath" (pronounced as *location path*) is the start production, "axis" denotes axis relations and "ntst" denotes tags labeling document nodes or the star '*' that matches all tags (these are called node tests). The "fexpr" will be called *filter expressions* after their use as filters in location paths. By an XPath expression we always mean a "locpath."

The semantics of XPath expressions is given with respect to an XML document modeled as a finite *node labeled sibling ordered tree*[2] (tree for short). Each node in the tree is labeled with a set of primitive symbols from some alphabet Λ. Sibling ordered trees come with two binary relations, the child relation, denoted by $R_\downarrow$, and the immediate_right_sibling relation, denoted by $R_\rightarrow$. Together with their inverses $R_\uparrow$ and $R_\leftarrow$ they are used to interpret the axis relations. We denote such trees as first order structures $(N, R_\downarrow, R_\rightarrow, P_i)_{i\in\Lambda}$.

Each location path denotes a binary relation (a set of paths). The meaning of the filter expressions is given by the predicate $\mathcal{E}(n, \mathsf{fexpr})$ which assigns a boolean value. Thus a filter expression fexpr is most naturally viewed as denoting a set of nodes: all n such that $\mathcal{E}(n, \mathsf{fexpr})$ is true. For examples, we refer to [8]. Given a tree $\mathfrak{M}$ and an expression R, the denotation or meaning of R in $\mathfrak{M}$ is written as $[\![R]\!]_{\mathfrak{M}}$. Table 2 contains the definition of $[\![\cdot]\!]_{\mathfrak{M}}$.

As discussed, one of the purposes of XPath is to select sets of nodes. For this purpose, the notion of an *answer set* is defined. For R an XPath expression, and $\mathfrak{M}$ a model, $answer_{\mathfrak{M}}(R) = \{n \mid \exists n'(n', n) \in [\![R]\!]_{\mathfrak{M}}\}$. Thus the answer set of R consists of all nodes which are reachable by the path R from some point in the tree.

Even Core XPath contains a bit of syntactic sugar. From Table 2 it is immediately clear that both `following` and `preceding` are definable. Also, the use of / in front of an expression can be eliminated, as follows:

$$/R \equiv \texttt{ancestor\_or\_self} :: *[\texttt{not parent} :: *]/R.$$

As our analysis considers expressive power, we may safely assume that these three do not occur in expressions and we do so without mentioning it.

3 The Answer Sets of XPath

We show that on ordered trees, Core XPath is as expressive as first order logic in two variables over the

[2]A sibling ordered tree is a structure isomorphic to $(N, R_\downarrow, R_\rightarrow)$ where N is a set of finite sequences of natural numbers closed under taking initial segments, and for any sequence s, if $s \cdot k \in N$, then either $k = 0$ or $s \cdot k - 1 \in N$. For $n, n' \in N$, $nR_\downarrow n'$ holds iff $n' = n \cdot k$ for k a natural number; $nR_\rightarrow n'$ holds iff $n = s \cdot k$ and $n' = s \cdot k + 1$.

```
locpath ::= axis'::'ntst | axis'::'ntst'['fexpr']' | '/'locpath | locpath'/'locpath |
            locpath '|' locpath
  fexpr ::= locpath | not fexpr | fexpr and fexpr | fexpr or fexpr
   axis ::= self | child | parent | descendant | descendant_or_self | ancestor | ancestor_or_self |
            following_sibling | preceding_sibling | following | preceding.
```

Table 1: Syntax of Core XPath.

$$
\begin{aligned}
[\![\texttt{axis} :: P_i]\!]_{\mathfrak{M}} &= \{(n,n') \mid n[\![\texttt{axis}]\!]_{\mathfrak{M}} n' \text{ and } P_i(n')\} \\
[\![\texttt{axis} :: P_i[e]]\!]_{\mathfrak{M}} &= \{(n,n') \mid n[\![\texttt{axis}]\!]_{\mathfrak{M}} n' \text{ and } P_i(n') \text{ and } \mathcal{E}_{\mathfrak{M}}(n',e)\} \\
[\![/\text{locpath}]\!]_{\mathfrak{M}} &= \{(n,n') \mid (\mathsf{root},n') \in [\![\text{locpath}]\!]_{\mathfrak{M}}\} \\
[\![\text{locpath}/\text{locpath}]\!]_{\mathfrak{M}} &= [\![\text{locpath}]\!]_{\mathfrak{M}} \circ [\![\text{locpath}]\!]_{\mathfrak{M}} \\
[\![\text{locpath} \mid \text{locpath}]\!]_{\mathfrak{M}} &= [\![\text{locpath}]\!]_{\mathfrak{M}} \cup [\![\text{locpath}]\!]_{\mathfrak{M}}
\end{aligned}
$$

$$
\begin{aligned}
[\![\mathsf{self}]\!]_{\mathfrak{M}} &:= \{(x,y) \mid x = y\} \\
[\![\texttt{child}]\!]_{\mathfrak{M}} &:= R_{\downarrow} \\
[\![\texttt{parent}]\!]_{\mathfrak{M}} &:= [\![\texttt{child}]\!]^{-1}_{\mathfrak{M}} \\
[\![\texttt{descendant}]\!]_{\mathfrak{M}} &:= [\![\texttt{child}]\!]^{+}_{\mathfrak{M}} \\
[\![\texttt{descendant\_or\_self}]\!]_{\mathfrak{M}} &:= [\![\texttt{child}]\!]^{*}_{\mathfrak{M}} \\
[\![\texttt{ancestor}]\!]_{\mathfrak{M}} &:= [\![\texttt{descendant}]\!]^{-1}_{\mathfrak{M}} \\
[\![\texttt{ancestor\_or\_self}]\!]_{\mathfrak{M}} &:= [\![\texttt{descendant\_or\_self}]\!]^{-1}_{\mathfrak{M}} \\
[\![\texttt{following\_sibling}]\!]_{\mathfrak{M}} &:= R^{+}_{\rightarrow} \\
[\![\texttt{preceding\_sibling}]\!]_{\mathfrak{M}} &:= [\![\texttt{following\_sibling}]\!]^{-1}_{\mathfrak{M}} \\
[\![\texttt{following}]\!]_{\mathfrak{M}} &:= [\![\texttt{ancestor\_or\_self}]\!]_{\mathfrak{M}} \circ [\![\texttt{following\_sibling}]\!]_{\mathfrak{M}} \circ [\![\texttt{descendant\_or\_self}]\!]_{\mathfrak{M}} \\
[\![\texttt{preceding}]\!]_{\mathfrak{M}} &:= [\![\texttt{ancestor\_or\_self}]\!]_{\mathfrak{M}} \circ [\![\texttt{preceding\_sibling}]\!]_{\mathfrak{M}} \circ [\![\texttt{descendant\_or\_self}]\!]_{\mathfrak{M}}
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{E}_{\mathfrak{M}}(n, \text{locpath}) = \mathit{true} &\iff \exists n' : (n,n') \in [\![\text{locpath}]\!]_{\mathfrak{M}} \\
\mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}_1 \texttt{ and } \text{fexpr}_2) = \mathit{true} &\iff \mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}_1) = \mathit{true} \text{ and } \mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}_2) = \mathit{true} \\
\mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}_1 \texttt{ or } \text{fexpr}_2) = \mathit{true} &\iff \mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}_1) = \mathit{true} \text{ or } \mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}_2) = \mathit{true} \\
\mathcal{E}_{\mathfrak{M}}(n, \texttt{not } \text{fexpr}) = \mathit{true} &\iff \mathcal{E}_{\mathfrak{M}}(n, \text{fexpr}) = \mathit{false}
\end{aligned}
$$

Table 2: The semantics of Core XPath.

signature with predicates corresponding to the `child`, `descendant`, and `following_sibling` axis relations. More precisely, we show that for every XPath expression R, there exists an XPath filter expression A such that, on every model $\mathfrak{M}$,

(2) $\quad \mathit{answer}_{\mathfrak{M}}(R) = \{n \mid \mathcal{E}_{\mathfrak{M}}(n, A) = \mathit{true}\}.$

Then, we show that every first order formula $\phi(x)$ in the signature just mentioned is equivalent to an XPath filter expression A in the sense that for every model $\mathfrak{M}$, and for every node n,

(3) $\quad \mathfrak{M} \models \phi(n)$ if and only if $\mathcal{E}_{\mathfrak{M}}(n, A) = \mathit{true}$.

First, though, we fix our terminology. We work with first order logic over node labeled ordered trees in a signature with unary predicates from $\Lambda = \{P_1, P_2, \ldots\}$ corresponding to the node tags, and with a number of binary predicates corresponding to "moves" in a tree. We use the predicates `child`, `descendant` and `following_sibling`. Let $\mathrm{FO}_{\mathsf{tree}}$ be the first order language in this signature. $\mathrm{FO}^2_{\mathsf{tree}} \subset \mathrm{FO}_{\mathsf{tree}}$ denotes the set of first order formulas $\phi(x)$ in which at most x occurs free, and which contain at most two variables.

Theorem 2 *For every formula $\phi(x)$ in $\mathrm{FO}^2_{\mathsf{tree}}$ with unary predicates from Λ, there exists a Core XPath expression R written with node tags Λ, such that on every tree $\mathfrak{M}$, $\mathit{answer}_{\mathfrak{M}}(R) = \{n \mid \mathfrak{M} \models \phi(n)\}$, and conversely.*

Proof. First we show that for any Core XPath expression R there exists a Core XPath filter expression A, whose size is linear in the size of R, such that for each model $\mathfrak{M}$,

(2) $\quad \mathit{answer}_{\mathfrak{M}}(R) = \{n \mid \mathcal{E}_{\mathfrak{M}}(n, A) = \mathit{true}\}.$

Consider an arbitrary XPath expression R. Obtain A by applying the converse operator $(\cdot)^{-1}$ as follows:

$$
\begin{aligned}
(S \mid T)^{-1} &\equiv S^{-1} \mid T^{-1} \\
(S/T)^{-1} &\equiv T^{-1}/S^{-1} \\
(\texttt{axis} :: \mathtt{P_i}[B])^{-1} &\equiv \mathsf{self} :: \mathtt{P_i}[B]/\texttt{axis}^{-1} :: *,
\end{aligned}
$$

with axis^{-1} having the obvious meaning. Then (2) holds.

Now we can show the easy side of Theorem 2. Let R be a Core XPath expression and let $A = R^{-1}$. Apply

the standard translation well-known from modal logic (cf. [4]) to A to obtain the desired first order formula. The translation is just the definition of $\mathcal{E}$ from Table 2 written in first order logic.

The hard direction follows more or less directly from the argument used to show a similar statement for linear orders, characterizing temporal logic with only unary temporal connectives by Etessami, Vardi and Wilke [6]. Let $\phi(x)$ be the first order formula. We will provide an XPath filter expression A such that (3) holds. Whence $/\texttt{descendant\_or\_self} :: *[A]$ is the desired absolute XPath expression. The proof is a copy of the one for linear temporal logic in [6, Theorem 1]. The only real change needed is in the set of order types: they are given in the right hand side of Table 3, together with the needed translations (A' denotes the translation of A). The other change is rather cosmetic. For A an atom, $A(x)$ needs to be translated using the self axis as $\texttt{self} :: \texttt{A}$. Thus, for instance, $\exists y(y\,\texttt{child}\,x \wedge A(y))$ translates to $\texttt{parent} :: *[\texttt{self} :: \texttt{A}]$. Translating $\phi(x)$, the result of this process is a filter expression A for which in any model $\mathfrak{M}$, for every node n, $\mathcal{E}_{\mathfrak{M}}(n, A)$ equals true iff $\mathfrak{M} \models \phi(n)$. QED

We note that, as in [6], the size of the filter expression is exponential in the size of the first order formula. [6] shows that this is unavoidable, even finite linear structures, so also on trees this bound is tight.

The first statement (2) in the above proof shows that Core XPath is as expressive as its filter expressions. Interestingly, Core XPath's filter expressions were introduced already in [5] for exactly the same purpose as the XPath language: specifying sets of nodes in finite ordered trees. The only difference is that the language of [5] does not have the asymmetry between the vertical and the horizontal axis relations: the immediate left and right sibling relations are also present. [5] provides a complete axiomatization, in a logic called LOFT (Logic Of Finite Trees), which might be of interest for query rewriting.

4 The Paths of XPath

In the previous section, we characterized the answer sets of XPath. We now turn to the sets of paths that can be defined in XPath; they too admit an elegant characterization which we provide here. First, we define the appropriate first order language.

A *conjunctive path query* is a conjunctive query of the form

$$Q(x, y) :- R_1, \ldots, R_n, \phi_1, \ldots, \phi_m,$$

in which the R_i are relations from the signature $\{\texttt{descendant}, \texttt{child}, \texttt{following\_sibling}\}$ and all of the ϕ_i are formulas in $\mathrm{FO}^2_{\mathsf{tree}}$ in one free variable. An example is

$$Q(x, y) :- z\,\texttt{descendant}\,x, z\,\texttt{following\_sibling}\,z', z'\,\texttt{descendant}\,y, P_1(z), P_2(y),$$

which is equivalent to the XPath expression

$$\texttt{ancestor} :: \texttt{P}_1/\texttt{following\_sibling} :: */\texttt{descendant} :: \texttt{P}_2.$$

With a *union of conjunctive path queries* we mean a disjunction of such queries with all of them the same two free variables x and y. For example,

$$\texttt{descendant} :: \texttt{P}_2 \mid \texttt{parent} :: */\texttt{ancestor} :: \texttt{P}_1$$

is equivalent to the union of the two queries

$$\begin{aligned} Q_1(x, y) &:- x\,\texttt{descendant}\,y, P_2(y). \\ Q_2(x, y) &:- z\,\texttt{child}\,x, z\,\texttt{ancestor}\,y, P_1(y). \end{aligned}$$

From Theorem 2 and some simple syntactic manipulation we immediately obtain

Proposition 3 *Every XPath expression is equivalent to a union of conjunctive path queries.*

The converse also holds, which gives us a characterization of the XPath definable sets of paths.

Theorem 4 *For every union of conjunctive path queries $Q(x, y)$ there exists a Core XPath expression R such that for every model $\mathfrak{M}$, $\{(n, n') \mid \mathfrak{M} \models Q(n, n')\} = [\![R]\!]_{\mathfrak{M}}$.*

Proof. By [2] or [9] every positive existential first order formula in two free variables is equivalent to a positive XPath expression. We can treat the first order formulas ϕ_i in a query as atomic symbols P_i, obtain the equivalent XPath expression and use (3) to substitute the P_i by XPath filter expressions which are equivalent to ϕ_i. QED

4.1 Structural Properties of XPath

Benedikt, Fan and Kuper [2] have given an in-depth analysis of a number of structural properties of fragments of XPath. Their fragments are all positive (no negations inside the filters) and restricted to the "vertical" axis relations defined along the tree order. All their fragments allowing filter expressions are closed under intersection, while none is closed under complementation. Here, we show that this is also true for full XPath. From Theorem 4 and Proposition 3 we obtain

Theorem 5 *Core XPath is closed under intersections. That is, for every two Core XPath expressions A, B, there exists a Core XPath expression C such that on every model $\mathfrak{M}$, $[\![A]\!]_{\mathfrak{M}} \cap [\![B]\!]_{\mathfrak{M}} = [\![C]\!]_{\mathfrak{M}}$.*

$\tau(x,y)$	$\exists y(\tau(x,y) \wedge A(y))$
$x = y$	self :: $*[A']$
x child y	child :: $*[A']$
y child x	parent :: $*[A']$
x following_sibling y	following_sibling :: $*[A']$
y following_sibling x	preceding_sibling :: $*[A']$
x descendant $y \wedge \neg x$ child y	child :: $*$/descendant :: $*[A']$
y descendant $x \wedge \neg y$ child x	parent :: $*$/ancestor :: $*[A']$.

Table 3: Order types and their translation

On the other hand, unfortunately,

Theorem 6 *Core XPath is not closed under complementation.*

PROOF. Suppose it was. We will derive a contradiction. Then (1) would be expressible. (1) is equivalent to the first-order formula

$$\exists y(x \texttt{ descendant } y \wedge A(y) \wedge \forall z((x \texttt{ descendant } z \wedge z \texttt{ descendant } y) \rightarrow \neg B(z))).$$

A standard argument shows that this set cannot be specified using less than three variables. This contradicts Theorem 2 which states that the answer set of every XPath expression is equivalent to a first order formula in two variables. QED

5 The Connectives of XPath

In this section we look at the connectives of XPath and argue that they are very well chosen. We disregard the following and preceding axis relations as well as absolute expressions (those are expressions starting with a /) as they are just syntactic sugar. What are the connectives of XPath? This question is not trivial. Clearly, there is composition ('/') and union ('|') of paths. Then there is composition with a filter expression ('[F]'). And inside the filter expressions all boolean connectives are allowed. This set can be streamlined as follows. Consider the following definition of path formulas:

$$R ::= \texttt{axis} \mid ?P_i \mid R/R \mid R|R \mid \sim R, \quad (4)$$

for axis one of XPath's axis relations, P_i a tagname, and the following meaning for the two new connectives:

$$\begin{aligned} [\![?P_i]\!]_{\mathfrak{M}} &= \{(x,x) \mid x \text{ is labelled with } P_i\} \\ [\![\sim R]\!]_{\mathfrak{M}} &= \{(x,y) \mid x = y \text{ and } \neg\exists z\ (x,z) \in [\![R]\!]_{\mathfrak{M}}\}. \end{aligned}$$

We call this language SCX (short for *Short Core XPath*). $?P_i$ simply tests whether a node has tag P_i. Thus child :: $\texttt{P}_\texttt{i}$ can be written as child/$?P_i$. The unary operator $\sim$ is sometimes called *counterdomain*. For instance, $\sim$child defines the set of all pairs (x,x) for x a leaf, and $\sim$parent the singleton $\{(\mathsf{root},\mathsf{root})\}$.

Below we explain why this set of connectives is so nice. First we show that this definition is equivalent in a very strong sense to that of XPath.

Theorem 7 *There exist linear translations t_1, t_2 with t_1 : XPath $\longrightarrow$ SCX and t_2 : SCX $\longrightarrow$ XPath such that for all models $\mathfrak{M}$, the following hold:*

- *for every XPath expression R, $[\![R]\!]_{\mathfrak{M}} = [\![t_1(R)]\!]_{\mathfrak{M}}$,*
- *for every* SCX *expression R, $[\![R]\!]_{\mathfrak{M}} = [\![t_2(R)]\!]_{\mathfrak{M}}$.*

PROOF. Because the counterdomain of a relation R is definable in XPath as self :: $*$[not R], every relation defined in (4) can be expressed as an XPath formula. For the other side, first observe that axis :: $\texttt{P}_\texttt{i}$ and axis/$?P_i$ are equivalent. As both languages are closed under composition and union, we only have to show that all filter expressions are expressible. With the following equivalences we can extend ? to all filter expressions (cf. [4, Lemma 2.82]):

$$\begin{aligned} ?(\texttt{axis} :: \texttt{P}_\texttt{i}) &\equiv \sim\sim(\texttt{axis}/?P_i) \\ ?(\texttt{axis} :: *) &\equiv \sim\sim(\texttt{axis}/(?P_i \mid \sim ?P_i) \\ ?(\texttt{axis} :: \texttt{P}_\texttt{i}[A]) &\equiv \sim\sim(\texttt{axis}/?P_i/?A) \\ ?(\text{not } A) &\equiv \sim ?A \\ ?(A \text{ and } B) &\equiv ?A \,/\, ?B \\ ?(A \text{ or } B) &\equiv ?A \mid ?B. \end{aligned}$$

A simple semantic argument shows the correctness of these equations. QED

So we can conclude that the "true" set of XPath connectives consists of testing a node tag, composition, union and counterdomain. This set of connectives between binary relations is closely connected to the notion of bisimulation, as exemplified in Theorem 8 below. Before we state the result, we need a couple of definitions.

For P a set of tag names, and R a set of relation names, let $B_{P,R}$ denote the P,R bisimulation relation. Let D, D' be first order models and $B_{P,R} \subseteq |D| \times |D'|$, with $|D|$ denoting the domain of D. We call $B_{P,R}$ a P,R *bisimulation* if, whenever $xB_{P,R}y$, then the following conditions hold, for all relations $S \in R$,

tag x and y have the same tag names, for all tag names in P;

forth if there exists an $x' \in D$ such that xSx', then there exists a $y' \in D'$ such that ySy' and $x'B_{P,R}y'$;

back similarly for $y' \in D'$.

Let $\alpha(x, y)$ be a first order formula in the signature with unary predicates P and binary relations R. We say that $\alpha(x, y)$ is *safe for* P, R *bisimulations* if the back and forth clauses of the bisimulation definition hold for $\alpha(x, y)$, for all P, R bisimulations. In words, if $\alpha(x, y)$ is safe for bisimulations, it acts like a morphism with respect to bisimulations. It is easy to see that all relations defined in (4) are safe for bisimulations respecting the node tags and the atomic axis relations. The other direction is known as Van Benthem's safety theorem (see [4, Theorem 2.83]):

Theorem 8 (Van Benthem) *Let $\alpha(x, y)$ be as above. If $\alpha(x, y)$ is safe for P, R bisimulations it can be defined by the grammar in (4).*

Why is this result so important? XPath is a language in which we can specify relations between nodes, and in several applications it is used in this way. Theorems 8 and 7 together guarantee that XPath is in a well defined sense *complete*: every relation which is safe for bisimulations respecting node tags and XPath's axis relations can be defined in XPath.

6 Conclusions

We have given semantic characterizations of navigational XPath in terms of natural fragments of first order logic. Besides that, we looked at the connectives of XPath and argued that they are nicely chosen. We conclude that the navigational part of XPath is a very well designed language. On ordered trees it corresponds to a natural fragment of first order logic. This holds both for the sets of nodes and the sets of paths definable in XPath.

The only negative aspect we discovered concerning XPath is that it is not closed under complementation. Thus first order logic is more expressive than XPath, both in defining sets of nodes and sets of paths. Marx [10] showed that expanding XPath with conditional axis relations[3] yields expressive completeness for answer sets. Marx [11] shows that the same language is also complete for expressing every first order definable set of paths.

[3] A conditional axis relation is of the form $(\texttt{child} :: \text{ntst}[\text{fexpr}])^*$ which denotes the reflexive and transitive closure of the relation denoted by $\texttt{child} :: \text{ntst}[\text{fexpr}]$. Using this we can express the set of nodes in (1) by

$$\texttt{self} :: *[(\texttt{child} :: *[\texttt{not self} :: \texttt{B}])^*/\texttt{child} :: \texttt{A}].$$

Acknowledgments

We want thank Loredana Afanasiev, Jan Hidders, and Petrucio Viana for valuable feedback. Maarten Marx was supported by the Netherlands Organization for Scientific Research (NWO), under project numbers 612.000.106 and 017.001.190. Maarten de Rijke was supported by grants from NWO, under project numbers 365-20-005, 220-80-001, 612.069.006, 612.000.106, 612.000.207, 612.066.302, 264-70-050, and 017.001.190.

References

[1] L. Afanasiev, M. Francheschet, M. Marx, and M. de Rijke. CTL Model Checking for Processing Simple XPath Queries. In *Proc. TIME 2004*, 2004.

[2] M. Benedikt, W. Fan, and G. Kuper. Structural properties of XPath fragments. In *Proceedings. ICDT 2003*, 2003.

[3] G. Bex, S. Maneth, and F. Neven. A formal model for an expressive fragment of XSLT. *Information Systems*, 27(1):21–39, 2002.

[4] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001.

[5] P. Blackburn, W. Meyer-Viol, and M. de Rijke. A proof system for finite trees. In *CSL'96*, pages 86–105, 1996.

[6] K. Etessami, M. Vardi, and Th. Wilke. First-order logic with two variables and unary temporal logic.

[7] G. Gottlob and C. Koch. Monadic queries over tree-structured data. In *Proc. LICS*, Copenhagen, 2002.

[8] G. Gottlob, C. Koch, and R. Pichler. The complexity of XPath query evaluation. In *PODS'03*, pages 179–190, 2003.

[9] G. Gottlob, C. Koch, and K. Schulz. Conjunctive queries over trees. In *Proc. PODS*, pages 189–200, 2004.

[10] M. Marx. Conditional XPath, the first order complete XPath dialect. In *Proc. PODS'04*, pages 13–22, 2004.

[11] M. Marx. First order paths in ordered trees. In T. Eiter and L. Libkin, editors, *Proc. ICDT 2005*, volume 3363 of *LNCS*, pages 114–128, 2005.

[12] G. Miklau and D. Suciu. Containment and equivalence for an XPath fragment. In *Proc. PODS'02*, pages 65–76, 2002.

[13] T. Milo, D. Suciu, and V. Vianu. Typechecking for XML transformers. In *Proc. PODS*, pages 11–22. ACM, 2000.

[14] M. Murata. Extended path expressions for XML. In *Proc. PODS*, 2001.

[15] F. Neven and T. Schwentick. Expressive and efficient pattern languages for tree-structured data. In *Proc. PODS*, pages 145–156. ACM, 2000.

[16] V. Vianu. A Web odyssey: from Codd to XML. In *Proc. PODS*, pages 1–15. ACM Press, 2001.

[17] W3C. XML path language (XPath): Version 1.0. http://www.w3.org/TR/xpath.html.

[18] W3C. XML path language (XPath): Version 2.0. http://www.w3.org/TR/xpath20/.

[19] W3C. XML schema part 1: Structures. http://www.w3.org/TR/xmlschema-1.

[20] W3C. XQuery 1.0: A query language for XML. http://www.w3.org/TR//xquery/.

[21] W3C. XSL transformations language (XSLT): Version 2.0. http://www.w3.org/TR/xslt20/.

[22] P. Wadler. Two semantics for XPath. Technical report, Bell Labs, 2000.

Nested Intervals Tree Encoding in SQL

Vadim Tropashko
Oracle Corp.

Abstract

Nested Intervals generalize Nested Sets. They are immune to hierarchy reorganization problem. They allow answering ancestor path hierarchical queries algorithmically - without accessing the stored hierarchy relation.

1 Introduction

There are several SQL techniques to query graph structures, in general, and trees, in particular [2]. They can be classified into 2 major categories: *Hierarchical/recursive SQL extensions* and *Tree encodings*. This article focuses upon tree encodings.

Tree encodings methods themselves can be split into 2 groups: *Materialized Path* and *Nested Sets.*

Materialized Path is nearly ubiquitous encoding, where each tree node is labeled with the path from the node to the root. UNIX global filenames is well known showcase for this idea. Materialized Path could be either represented as character string of unique sibling identifiers (concatenated with some separator), or enveloped into user defined type [5].

Querying trees with Materialized Path technique doesn't appear especially elegant. It implies either string matching like this

```
select e1.ename from emp e1, emp e2
where e2.path like e1.path || '%'
and e2.name = 'FORD'
```

or leveraging complex data types that are realm of Object-Relational Databases. The alternative tree encoding - Nested Sets [2] labels each node with just a pair of integers. Ancestor-descendant relationship is reflected by subset relation between intervals of integers, which provides very intuitive base for hierarchical queries.

Although Nested Sets are certainly appealing to many database developers, they have 2 fundamental disadvantages:

1. The encoding is volatile. In a word, roughly half of the tree nodes should be relabeled whenever a new node were inserted.

2. Querying ranges is asymmetric from performance perspective. It is easy to answer if a point falls inside some interval, but it is hard to index a set of intervals that contain a given point. For nested sets this translates into a difficulty answering queries about node's ancestors.

[6] introduced *Nested Intervals* that generalize Nested Sets. Since Nested Sets encoding with integers admits only finite gaps for new node insertions, it is natural to use dense domain such as rational numbers. One particular encoding schema with *Dyadic rational numbers* was developed in the rest of the article, and was a subject of further improvements in the follow up articles. Dyadic rational encoding has many nice theoretical properties, and essentially is a numeric reflection of Materialized Path. It has, however, one significant flaw from practical perspective. Dyadic fractions utilize domain of integer numbers rather uneconomically, so that numeric overflow prevents tree scaling to any significant size.

In general, Nested Intervals allow a certain freedom choosing particular encoding schema. [7] developed alternative encoding with *Farey fractions*. The development continued in [8].

This article expands the perspective. It demonstrates why both methods are natural choices, and describes the mapping between those tree encodings. It goes on exploring different ways of establishing interval structure. The major result is introducing Path Matrices and exposing their properties.

Similar idea of leveraging Stern-Brocot tree is briefly mentioned in [1]. The article, however,

contains just a hint, while pursuing some other venue. [3] is, perhaps, the earliest reference in the database literature referring to Continued Fractions encoding. The manuscript is unavailable to the author to draw detailed comparison with his method.

2 Nested Intervals Queries

Nested Intervals encode each tree node with a pair of numbers *head* and *tail*. Interval for a child node is always contained within parent interval. With this labeling transitive closure could be queried like this

```
select e1.ename, e2.ename
from emp e1, emp e2
where e2.head >= e1.head
and e2.head < e1.tail
```

Next, the subtree of all the descendants of a node could be found by just restricting the above view with a single table predicate

```
select e2.ename from emp e1, emp e2
where e2.head >= e1.head
and e2.head < e1.tail
and e1.ename = 'SCOTT'
```

The ancestor path can be queried symmetrically

```
select e1.ename from emp e1, emp e2
where e2.head >= e1.head
and e2.head < e1.tail
and e2.ename = 'SCOTT'
```

There is a subtle problem with the last query, however. Finding all the intervals that cover a given point is difficult. Although there are specialized indexing schemes like R-Tree, none of them is as universally accepted as B-Tree.

Compare this to the descendants query, assuming that the subtree of *SCOTT*'s subordinates is small. The execution path in this case is very efficient: first, *e1* record is fetched by the unique index, and then all the *e2* records are fetched by index range scan.

The details of Nested Intervals encoding are developed in the next sections. The encoding is *algorithmic*. Given a child node label, the parent encoding can be calculated, not queried. Therefore, the whole path to the root node can be calculated. Hence, if we know tree node encodings for all nodes on the path then, the nodes themselves can be efficiently queried in the database.

3 Interval Halving

The easiest way to nest intervals is splitting parent interval into two halves. If we start with the points 0 and 1 and continue on halving the intervals iteratively then, what kind of numbers on the interval boundaries would be produced? Clearly, the ones whose denominator is power of 2, or simply dyadic fractions [4].

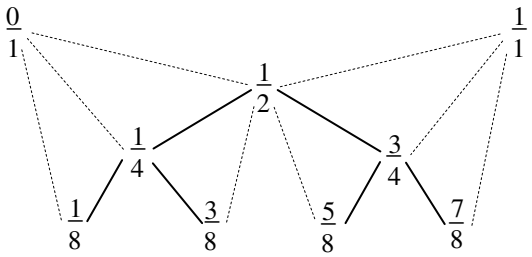


Fig.1. Dyadic Fractions at Conway tree.

When splitting the interval `[head,tail]` into two, the point on the boundary is the *average* `(head+tail)/2`. Alternatively, we could have chosen the *mediant*:

$$\frac{head\_numer + tail\_numer}{head\_denom + tail\_denom}$$

If we start with the points 0 and 1 and continue on, then the *Stern-Brocot tree* of *Farey fractions* would be produced.

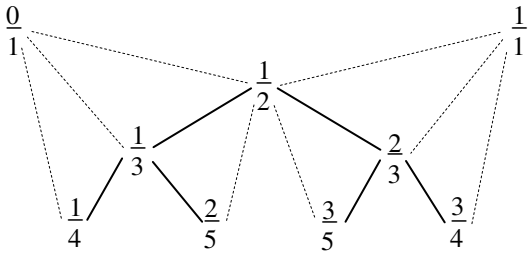


Fig.2. Farey fractions at Stern-Brocot tree.

The bijection between Dyadic and Stern-Brocot tree is defined by the following Minkowski Question Mark function `?: [0,1]→[0,1]` [9].

If x has binary expansion `.00...011...100...011...1...`, where there are `a` zeros in the first block, then `b` ones in the second, then `c` zeros, and so on, then `?(x)` is the (simple) continued fraction

$$\cfrac{1}{a+1+\cfrac{1}{b+\cfrac{1}{c+\cfrac{1}{so\_on}}}}$$

Thus, for example, if `x = 1/4`, its two binary expansions `.0100000...` and `.00111111...` yield the two expressions

$$\cfrac{1}{1+1+\cfrac{1}{1+\cfrac{1}{\infty}}} = \cfrac{1}{2+1+\cfrac{1}{\infty}}$$

Therefore, `?(1/4)=1/3`. Note that the node ¼ is positioned in the Dyadic tree on Fig.1 in the same place where the node 1/3 is in Stern-Brocot tree on Fig.2.

4 Nested Interval Structure

In previous section we developed two alternative but isomorphic systems how to generate interval boundary points. What intervals should we consider? Clearly, including all possible intervals into our system would be too much. In Farey case (Fig.2), for example, the interval `[1/3,1/2]` would have at least two parents: `[1/3,2/3]` and `[0/1,1/1]`.

What if we limit the scope to only those intervals that correspond to the edges at Fig.1? If we consider solid and dashed lines, then there still would be too many intervals. Consider the interval `[1/3,2/5]` (Fig.2). How many siblings does it have? Well, no more than one: `[2/5,1/2]`. Indeed, no other interval has `[1/3,1/2]` as a parent.

If we consider solid lines only (with two additional convenience intervals at the top), then we'll get a system of Nested Intervals shown at Fig.3.

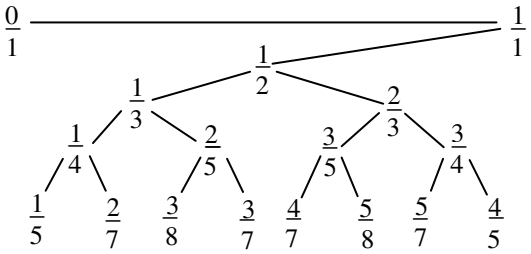


Fig.3. Simple Farey interval structure.

Dyadic Nested Interval structure is isomorphic to Fig.3 - we omit the picture in order to save space. The reason why we preferred Farey over Dyadic case would become evident in the last section. It would also become apparent why it's called "simple".

The other possible way to introduce Nested Interval structure is shown at Fig.4, this time with dyadic encoding.

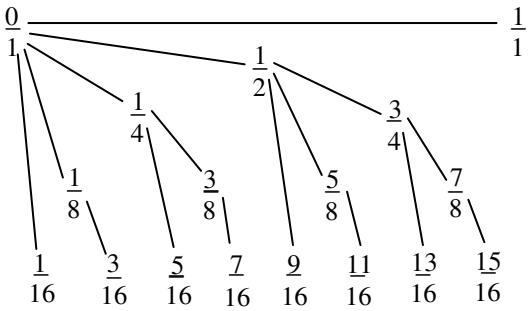


Fig.4. Monotonic dyadic interval structure.

Algorithms for navigating Dyadic Nested Intervals are almost obvious:

1. Younger sibling `[head,tail]` encoding is:

$$\left[\frac{2\ head\_numer\ +1}{2\ head\_denom}, \frac{2\ tail\_numer\ +1}{2\ tail\_denom}\right]$$

2. Older sibling `[head,tail]`:

$$\left[\frac{head\_numer\ -1}{head\_denom}, \frac{tail\_numer\ -1}{tail\_denom}\right]$$

(Be careful, however, when applying this rule to the first child)

3. Parent of the first child:

$$\left[\frac{head\_numer}{head\_denom}, \frac{tail\_numer + 1}{tail\_denom}\right]$$

Farey Intervals are little bit more sophisticated.

5 The Path Matrix

Let's study simple Farey interval structure (Fig.3) in more detail. Consider the interval `[5/7,3/4]`. It is the first child of `[2/3,3/4]`. Then, `[2/3,3/4]` is the second child of `[1/2,1/1]`. Finally, `[1/2,1/1]` is the first child of `[0/1,1/1]`. Therefore, the materialized path encoding of `[5/7,3/4]` is `1.2.1`. However, we have 4 intervals, i.e. 4 nodes, while materialized path has the length 3. How can that be?

Could interval `[0/1,1/1]` be considered as somebody's else child too? Well, yes, and no. The obvious parent candidate is `[0/1,1/0]`. Then, the interval `[1/1,2/1]` is the second child of `[0/1,1/0]`! The amended materialized path encoding for `[5/7,3/4]` is `1.1.2.1` and, by the way, we also are able to find Farey interval encodings for materialized paths beginning with natural numbers other than `1`.

Here is formal procedure converting Farey encoding into materialized path. Start with Farey interval written as 2x2 matrix:

$$\begin{bmatrix} 3 & 5 \\ 4 & 7 \end{bmatrix}$$

Note that we switched the fractions. The purpose is to keep the highest integer in the lower right corner. Next, write

```
5 = 3 * ⌊5/3⌋ + 5 mod 3 = 3*1 + 2
7 = 4 * ⌊7/4⌋ + 7 mod 4 = 4*1 + 3
```

The integer division result (which is `1` - the same in both cases) is the first element of the materialized path encoding. We add the column with the remainders to the left

```
2   3   5
3   4   7
```

Matrix on the left

$$\begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix}$$

corresponds to the interval `[2/3,3/4]` - the parent of our original interval.

Continuing to the left we get

```
0   1   2   3   5
1   1   3   4   7
```

together with the sequence of integer division results `1,1,2,` and `1`. We stop as soon as zero in the top left corner appears.

Next, we can expand this *number wall* up. The rule is the same but applied to rows instead of columns. In our example, we write

```
7 = 5*⌊7/5⌋ + 7 mod 5 = 5*1 + 2
4 = 3*⌊4/3⌋ + 4 mod 3 = 3*1 + 1
3 = 2*⌊3/2⌋ + 3 mod 2 = 2*1 + 1
1 = 1*⌊1/1⌋ + 1 mod 1 = 1*1 + 0
```

Adding the row `0,1,1,2` at the top results in

```
    0   1   1   2
0   1   2   3   5
1   1   3   4   7
```

Continuing this process, we get the following number wall

```
            0   1
        0   1   1
    0   1   1   2
0   1   2   3   5
1   1   3   4   7
```

Note, that all 2x2 sub-matrices at this wall have determinant `1` or `-1`. This property enforces the unique way of completing the wall to square matrix

```
1  0  1  0  1
0  1  0  1  1
1  0  1  1  2
0  1  2  3  5
1  1  3  4  7
```

We refer to the number wall that we just have built as the *Path Matrix*. It enjoys many nice properties.

1. The numbers at the main antidiagonal are all `1`s.
2. The sequence of numbers below the main antidiagonal is materialized path. The path is oriented from right to left.
3. Adjacent 2x2 sub-matrices can be multiplied as shown on Fig.5

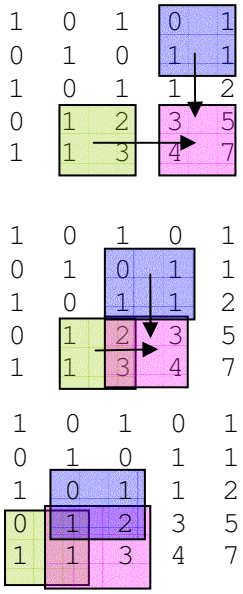


Fig.5. Multiplying adjacent matrices.

The matrix identity in the middle case on Fig.5, for example, is

$$\begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix}$$

Multiplying overlapping matrices, similar to the last case

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix}$$

works for matrices on the main antidiagonal only.

Iterative application of matrix multiplication property gives rise to the following matrix decomposition

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 5 \\ 4 & 7 \end{bmatrix}$$

Each of the matrices on the left side corresponds to an elementary fragment of the materialized path `1.1.2.1`. Since these elementary matrices all have determinant `-1`, their multiple would always have determinant `-1` or `1` - the property that we noticed earlier.

The determinant property allows filling in the numbers in the Path Matrix in the other direction. Suppose we know materialized path and want to calculate corresponding Farey interval. One way is multiplying elementary matrices, by leveraging the above matrix decomposition identity. Alternatively, we can start with partially filled in Path Matrix. By properties 1 and 2 we have

```
1  0  1  0  1
0  1  0  1  1
1  0  1  1
0  1  2
1  1
```

We fill in empty positions as follows. Select 2x2 matrix that has 3 elements defined and the 4th element empty

```
1  0  1  0  1
0  1  0  1  1
1  0  1  1
0  1  2
1  1
```

Fill in the empty position to satisfy the determinant property. The sign of the determinant is alternating. It is negative if the matrix is positioned at even distance from main antidiagonal, and positive otherwise. In our case, the matrix is just one step away from the position at the main antidiagonal. Therefore, the value `x` at the empty position has to satisfy the equation

```
1*x - 1*2 = 1
```

hence, `x=3`, as expected.

After all the empty positions are filled, we can grab 2x2 Farey interval matrix at the lower right corner.

The final important property of the Path Matrix is that matrix transposition corresponds to materialized path inversion.

6 Continued Fractions

[8] suggests one more perspective into Farey interval encoding. Materialized path `1.1.2.1` can be naturally written as the simple continued fraction

$$\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{1+x}}}}$$

which can be simplified into Moebius function

$$\frac{4+3\,x}{7+5\,x}$$

Here the familiar 2x2 matrix from our example can be recognized.

Simple continued fractions have somewhat irritating feature that increasing any denominator, either increases the value of the number, or decreases it, depending on the parity of the position. *Reversed* (or *additive*) continued fractions

$$\cfrac{1}{1+1-\cfrac{1}{1+1-\cfrac{1}{2+1-\cfrac{1}{1+1-x}}}}$$

are monotonic. The path matrix theory for the additive continued fractions mimics the classic case described in section 6. One of the distinguished feature of additive continued fractions is that all the matrices have negative entries in the second column, and determinant equal to 1. There is no alternation anymore: interval encoding of the younger child always precedes the older one. (In the simple continued fractions case this was true for the odd levels, and reversely true for the even ones). Finally, additive continued fractions map into monotonic Farey interval structure.

References

[1] D. Aioanei, A. Malinaru. General trees persisted in relational databases. http://www.codeproject.com/cs/database/persisting_trees.asp?print=true

[2] J. Celko. Joe Celko's Trees and Hierarchies in SQL for Smarties. Morgan Kaufmann.

[3] P. Ciaccia, D. Maio, and P. Tiberio. A method for hierarchy processing in relational systems. Information Systems, 14(2):93-105, 1989.

[4] J. Conway. On Numbers and Games. New York: Academic Press, Inc.

[5] J. Roy. 2003. Using the Node Data Type to Solve Problems with Hierarchies in DB2 Universal Database http://www106.ibm.com/developerworks/db2/library/techarticle/0302roy/0302roy.html

[6] V. Tropashko. Trees in SQL: Nested Sets and Materialized Path. http://www.dbazine.com/tropashko4.shtml

[7] V. Tropashko. Nested Intervals with Farey Fractions. http://arxiv.org/html/cs.DB/0401014

[8] V. Tropashko. Nested Intervals Tree Encoding with Continued Fractions. http://arxiv.org/pdf/cs.DB/0402051

[9] L. Vepstas. The Minkowski Question Mark and the Modular Group SL(2,Z). http://www.linas.org/math/chap-minkowski/chap-minkowski.html

The Indiana Center for Database Systems at Purdue University

Mourad Ouzzani Walid G. Aref Elisa Bertino Ann Christine Catlin
Christopher W. Clifton Wing-Kai Hon Ahmed K. Elmagarmid Arif Ghafoor
Susanne E. Hambrusch Sunil Prabhakar Jeffrey S. Vitter Xiang Zhang
Web Site: http://www.cs.purdue.edu/icds Purdue University, West Lafayette IN

1 Introduction

The Indiana Center for Database Systems (ICDS) at Purdue University has embarked in an ambitious endeavor to become a premiere world-class database research center. This goal is substantiated by the diversity of its research topics, the large and diverse funding base, and the steady publication trend in top conferences and journals. ICDS was founded with an initial grant from the State of Indiana Corporation of Science and Technology in 1990. Since then it has grown to now have 9 faculty members and about 30 total researchers. This report describes the major research projects underway at ICDS as well as efforts to move research toward practice.

2 Multimedia

Several projects have been launched to overcome challenges in organizing, storing, mining, and delivering multimedia data in a secure and timely manner.

2.1 Video Database Management

The vdbms Project [9, 3] has developed a video-enhanced database system that supports comprehensive and efficient database management, including: feature-based preprocessing for video content representation and indexing, video and meta-data storage, video query processing, buffer and storage management, and continuous video streaming. In this project, we view video as a well-defined data type with its own description, parameters, and applicable methods. Supporting video operations (storing, searching by content, and streaming) and new query types (query by example and multi-feature similarity search) requires major changes in many of the traditional system components. Specifically, the storage and buffer manager will have to deal with huge volumes of data with real-time constraints. Query processing has to consider the video methods and operators in generating, optimizing and executing query plans.

2.2 Semantic Video Databases

Several content-based video retrieval systems have been proposed in the past, but they still suffer from the following challenging problems: semantic gap, semantic video concept modeling, semantic video classification, and concept-oriented video database indexing and access. We propose a framework [8] that includes semantic-sensitive video content representation, semantic video concept interpretation, a novel semantic video-classifier training framework, and a concept-oriented video database organization technique to enable semantic-sensitive video retrieval and browsing.

2.3 Video Mining

To achieve efficient video indexing and access, we introduce strategies for video content structure and event mining [26]. The video shot segmentation and representative frame selection strategies are first utilized to parse the continuous video streams into physical units. Video shot grouping, group merging, and scene clustering schemes are then proposed to organize the video shots into a hierarchical structure using clustered scenes, scenes, groups, and shots, in increasing granularity from top to bottom. Audio and video processing techniques are then integrated to mine event information from the detected scenes. Finally, the acquired video content structure and events are integrated to construct a scalable video skimming tool.

2.4 Distributed Multimedia Systems

A large number of emerging Web-based applications require Distributed Multimedia Systems (DMIS) infrastructures. Examples of such applications abound in the domains of medicine, manufacturing, and e-commerce. Development of DMIS needs a broad range of technological solutions for organizing, storing, and delivering multimedia information in an integrated, secure and timely manner with guaranteed end-to-end quality of presentation (QoP). Management of integrated end-to-end QoP and ensuring information security in DMIS present formidable research and implementation challenges. These challenges encompass all the sub-system components of a DMIS. The ultimate objective relies on the performance and allocation of resources of each of the DMIS sub-system components including networks, databases, and end-systems. Several projects addressing these challenges are being pursued.

3 Security and Privacy

We have initiated several projects to address security and privacy issues in database systems, data mining, and data sharing.

3.1 Privacy-preserving DBMSs

This project investigates ways to extend current DBMS architectures, models and languages to support high-assurance privacy and achieve fine-grained access control. A first result has been the development of a model for labeling data with purpose information [4]. Our approach has several key features: (1) hierarchical organization of purposes according to an ontology, (2) support for positive and negative purposes, (3) support for the association of sets of purposes with a single data item, and (4) support for a variable range of purpose labeling granularities. The model has been implemented through the use of query modification techniques. The approach, initially developed for relational data, has since been extended to complex data, such as XML, and integrated with a role-based access control model [25]. We are currently investigating innovative access control techniques for relational database systems, to be able to enforce accesses to data at a very fine granularity level. In this context, we have proposed the innovative notion of micro-views, which allows one to fine-tune the values returned by queries.

3.2 Privacy-preserving Data Mining

Data mining relies on the collection of massive amounts of data - but this often collides with privacy considerations [18]. We propose to address this challenge in the distributed case. The solutions involve algorithms that share some information to calculate correct results where the shared information can be shown not to disclose private data. Our contribution in this context has been manifold: (1) We propose a toolkit of components that can be combined for specific privacy-preserving data mining applications. (2) We propose secure mining of association rules over horizontally partitioned data [17]. The methods incorporate cryptographic techniques to minimize the information shared, while adding little overhead to the mining task. (3) We propose a method for k-means clustering [23] when different sites contain different attributes for a common set of entities. (4) We propose a method to apply classification rules without revealing either the data or the rules. In addition, the rules can be verified not to use any "forbidden" criteria. A key strength of ICDS in this area is its collaboration with the Krannert School of Management to identify problems and develop solutions where incentives discourage cheating.

3.3 Privacy-preserving Data Sharing

Integrating and sharing data from multiple sources has been a long-standing challenge. Data integration is seriously hampered by an inability to ensure privacy. Problems include fear of disclosing confidential information as well as regulations protecting individual privacy. This project proposes to develop the technology needed to create and manage federated databases while controlling the disclosure of private data [7]. To address the above problems, we plan to develop solutions to several fundamental problems in privacy-preserving data integration and sharing, including (1) a privacy framework that is flexible and clear to the end users, (2) schema-matching techniques that expose the source data and schemas, and (3) querying across sources while ensuring that results do not violate privacy policy, disclosure of sources, and preventing private information leakage from answering a set of queries.

3.4 Watermarking

By enabling relatively cost-free, fast, and accurate access channels to information in digital form, computers have radically changed the way we think and express ideas. As increasingly more information is produced, packaged and delivered in digital form, one of its main features threatens to become its worst enemy: zero-cost verbatim copies. Digital Watermarking deploys Information Hiding as a method of Rights Protection to conceal an indelible "rights witness" (watermark) within the digital Work to be protected. We analyze digital watermarking from a higher level, domain-independent perspective [22]. We propose a theoretical model and ask: are there any limitations to what watermarking can do? What are these and when can they be reached? We then propose, design and analyze watermarking solutions for (1) numeric sets, (2) numeric relational data, (3) categorical data, (4) streams and (5) semi-structured data.

3.5 Untrusted Private Databases

Entities such as corporations often use databases to store confidential operational data. There is, however, a need to share this data with outside entities in a limited fashion. External entities may not fully trust data owners to share private data without malicious modification. We therefore need to enforce constraints on a private database without having access to the database. We are developing protocols to help a third party enforce constraints over an untrusted private database and obtain a guarantee that the constraints are enforced. One particular application of this research is privacy-preserving query result verification. In this model, we assume that the database periodically commits itself without revealing its contents. Subsequent queries are guaranteed to return correct results with respect to the committed state of the database. In addition to the correctness of the protocol, we are concerned about the overhead of the solution and the degree of exposure of private data in order to guarantee correctness.

4 Top-k Query Processing

Ranking queries produce results that are ordered on some computed score. Typically, in these queries, users are usually interested only in the top-k results. Current relational query processors do not handle ranking queries efficiently, especially when joins are involved. We developed rank-join algorithms [15, 14] that use the individual orders of its inputs to produce join results ordered on a user-specified scoring function. The idea is to rank the join results progressively during the join operation. We introduce physical query operators based on variants of ripple join.

Rank-join operators progressively rank the join results while performing the join operation. We propose a rank-aware query optimization framework [16] that fully integrates rank-join operators into relational query engines. The framework is based on extending the System R dynamic programming algorithm in both enumeration and pruning. Unlike traditional join operators, optimizing for rank-join operators depends on estimating the input cardinality of these operators. We introduce a probabilistic model for estimating the input cardinality, and hence the cost of a rank-join operator.

5 Data Streaming

The number of applications dealing with streams of data produced in an unpredictable and bursty fashion is growing in several areas including network traffic management, high-throughput instruments, and sensors. Nile [13] is a data stream management system under development at Purdue. Nile supports extended SQL operators that handle sliding-window execution. Specifically, Nile supports the efficient and correct pipelined execution of sliding window queries over multiple data streams. The correct execution is enforced via the Negative Tuple Approach [13]. Negative tuples are tuples that are generated whenever a tuple expires out of a sliding window. Each operator in the pipeline needs to react differently whenever it receives a negative tuple to counteract the effect of the corresponding positive tuple that just expired out of the window. Although negative tuples guarantee correct execution of query pipelines, they induce performance overhead. We are looking into optimization techniques to reduce the performance overhead induced by negative tuples. Another interesting feature of Nile is its predicate windows. In contrast to sliding windows that limit the focus of queries on streams to the most recent tuples, predicate windows can select tuples of interest that meet a certain select or join predicate.

We are currently working on several other extensions to Nile: (1) Shared execution of concurrent continuous queries over data streams [12]. (2) A summary manager based on the concept of *promising tuples* – we define shared summaries over data streams enabling the query engine to avoid looking at the original streams. Summaries are built using promising tuples which represent tuples that are "judged" important. (3) Context and profile-aware query processing – we are investigating context and profile awareness at the engine level with a generic extensible interface to define new contexts and modules for context-aware processing within the engine. (4) Native online data mining – Nile will have embedded online data mining operators that would operate on incoming streams to discover "interesting phenomena". This capability raises several issues like change detection in the streams and the need to cluster streams based on their behavior.

6 Spatio-Temporal Databases

The tremendous increase of cellular phones, GPS-like devices, and RFIDs results in highly dynamic environments where objects and queries are continuously moving. We are investigating issues related to spatio-temporal queries and location-aware services. The PLACE (Pervasive Location Aware Computing Environments) project focuses on the efficient support of location-aware services. The PLACE server extends data streaming management systems to support location-aware environments. The query processor includes: (1) new incremental spatio-temporal operators, (2) extended semantics for sliding window queries, and (3) a shared execution framework for scalable execution.

We propose two algorithms: SINA [20] for evaluating concurrent continuous spatio-temporal queries and SEA-CNN [24] for answering a collection of continuous concurrent k-nearest neighbor queries (CKNN). SINA achieves scalability by employing a shared execution paradigm where the execution of continuous spatio-temporal queries is abstracted as a spatial join between a set of moving objects and a set of moving queries. SINA employs an incremental evaluation that is achieved by computing only the updates of the previously reported answer. Experimental results show that SINA is scalable and is more efficient than other index-based spatio-temporal algorithms. SEA-CNN has two important features: incremental evaluation and shared execution. SEA-CNN achieves both efficiency and scalability in the presence of a set of concurrent queries. Furthermore, SEA-CNN does not make any assumption about the movement of objects or the mutability of the objects or the queries. Experimentation shows that SEA-CNN is highly scalable and is more efficient than other R-tree-based CKNN techniques.

7 Managing Data Uncertainty

Due to limited bandwidth and battery power, it is infeasible for a system to keep track of the actual values continuously produced by sensors. Database queries, in this case, may produce incorrect answers. We model

the uncertainty inherent to dynamic sensor data with a range of possible values together with the probability distribution of the values within that range [6]. We also propose probabilistic queries, which evaluate uncertain data and produce answers with probabilistic guarantees. A query classification scheme is proposed, where for each class, query evaluation algorithms and answer quality are presented. We also study the semantics of probabilistic comparison operators that return imprecise comparison results. We illustrate how uncertainty management techniques for sensor data can be extended to location data in moving-object databases. We investigate the efficiency issues of probabilistic query evaluation. In particular, we propose efficient algorithms to enhance the performance of nearest-neighbor queries, range queries, and joins. We are currently developing a prototype for handling uncertain data and probabilistic queries. Uncertain data types are treated as a first-class data type and an extended version of SQL allows probabilistic queries to be specified.

8 Indexing Techniques

8.1 Space Partitioning Trees

Emerging database applications require the use of new indexing structures beyond B-trees and R-trees. Examples are the k-D tree, the trie, the quadtree, and their variants. A common feature of all these indexes is that they recursively divide the space into partitions. We developed an extensible index structure, called SP-GiST [1, 2, 10], to support the class of space partitioning un-balanced trees. Simple method implementations are provided demonstrating how SP-GiST can behave as a k-D tree, a trie, a quadtree, or any of their variants. Issues related to clustering tree nodes into pages as well as concurrency control for SP-GiST are addressed. A dynamic minimum-height clustering technique is applied to minimize disk accesses and to make using such trees in database systems possible and efficient. SP-GiST is supported both in PostgreSQL and in a standalone version.

8.2 Indexing in High-Update Environments

Emerging applications such as sensor networks and location-based services require monitoring continuous changing data. In high update environments, traditional database indexes suffer from the need for frequent updates and result in poor performance. We propose new index structures that are tolerant to constant data updates and optimized to achieve the best overall performance for both querying and updating [6, 5].

For moving object data indexing, we proposed the Change-tolerant Rtree (CTRtree). We observe that objects often stay in a region (e.g., building) for an extended amount of time, and exploit this phenomenon to optimize an index for both updates and queries. We have developed the algorithms for creation and use of change tolerant Rtree. Experimental results establish the superior performance of the proposed index structures.

For sensor data indexing, we proposed the Mean Variance Tree (MVTree), which is built based on the mean and variance of the data instead of the actual data values that are in constant flux. We also proposed to take each constantly evolving data as a time series and use the time series methodology to analyze and model it. The time series model enables effective forecasts for the constantly evolving data. Based on the forecasted intervals, we developed the Forecasted Interval Index (FI-index). Experiments show that, compared to traditional indexing schemes, both the MVTree and the FI-Index substantially improve index updating performance while maintaining satisfactory query performance.

9 Efficient I/O for Data Intensive Applications

Recent trends in hardware development and data-intensive applications have resulted in a performance bottleneck for storing and retrieving data. The *I/O* communication between the fast internal memory and the slow external memory (such as disk) can be a bottleneck when processing massive amounts of data. The goal of this project is to develop techniques to alleviate the I/O bottleneck. One of the activities of the project has centered on the development of data placement, migration, and indexing techniques for video and multi-dimensional spatio-temporal data.

Problems involving massive amounts of geometric data are ubiquitous in spatial databases, geographic information systems, and virtual reality systems. For example, the NASA's Earth Observing System project produces petabytes of raster data per year. A major challenge is to develop mechanisms for processing the data, or else much of the data will be useless. In our work, we concentrate on the design and analysis of external memory data structures for batched and online problems involving geometric and spatial data. Many problems on geometric objects can be reduced to a small core of problems, such as computing intersections, convex hulls, and nearest neighbor search, for which we discuss useful paradigms for solving them in external memory.

10 Compressed Data Structures

10.1 Compressed Indexes for Text

This project targets the efficient indexing of massive documents. When a text T (of length n) is compressible, a natural question to ask is whether the index

can be compressed by a similar amount and still support fast searching of any substring in T. In addition, we would like to create indexes that implicitly encode the text as part of the indexing structure. Achieving this goal means that the text is no longer needed alongside the index. We also intend to support fast decompression starting from any position in the text and to support searching for arbitrary patterns in the text *without* scanning the entire compressed text. We develop a data structure that operates in nearly optimal space in terms of the entropy of the text T, while still supporting searches in $O(m \log |\Sigma| + polylog(n))$ time, where m is the length of the pattern being searched and $|\Sigma|$ is the size of the alphabet for text T. We also show several interesting tradeoffs and an adaptation of our data structuring technique for use only in compression.

10.2 Data-Aware Indexes for Set Data

In this project [11], we studied a way for representing a set S of items from $[1, n]$. Our target is to minimize the space usage, while trying to support 'nearly-optimal' time for queries on S. Our queries include: (1) rank(i): how many items in S that is less than i? (2) select(j): what is the j-th smallest item in S? and (3) member(k): is k an item in S? We use a popular data-aware method called gap encoding for space compression. In the practical case when the size of S is small (precisely, $|S| = o(n)$), we show that our data structure is a better alternative to the smallest existing one, matching its space and query time for rank(i), but supports faster query for select(j) and member(k).

11 Bioinformatics

The objective of our efforts in bioinformatics is to address different issues where database support is needed.

11.1 Database Support in Proteomics

High throughput instruments like mass spectrometers (MS) are becoming commonplace in life sciences. Current MS techniques may require repeated experimental runs consuming time and limited biological samples while demanding larger data storage resources. In this project, we propose a novel strategy for managing MS data that acquires data only for interesting molecules. In this strategy, a MS is operated primarily under the single MS mode. The system performs a rapid initial analysis of the full mass spectrum as it is being acquired and, based on the results, can predict the direction of the experiment and determine the data needed for the next step. If a potentially interesting ion is found, the instrument is directed to switch to the tandem MS/MS mode to acquire sequence information on the potentially interesting ion. We are also investigating other issues including prediction of upcoming spectra based on statistical methods and data mining, spectra indexing, and access to external databases.

11.2 Protein Annotation

One of the problems of current biological databases is that computational function annotations are sometimes not clearly distinguished from those with experimental evidence, and consequently, erroneous annotations are propagated in the databases by computational methods. Moreover, computational function annotation is informative and indispensable in the current large and growing biological databases. To manage the quality of computational function annotations, we address the following specific goals in this project: (1) Quantifying annotation reliability and errors. We will establish quantitative values related to annotation reliability for each protein family. (2) Providing benchmark data sets for computational annotation methods, including curated multiple sequence alignments of protein families. (3) Clarifying annotation dependency. (4) Providing re-annotation of genes associated with the reliability values. The system will support the functionality to allow users to generate their own private annotations. With a quantitative measure of the error rate, the system will provide aggressive function annotation, which will be useful to some researchers. (5) Identifying hot spots for experimental function verification from annotation dependencies.

12 Web Services

The Web is evolving from a passive medium for publishing data to a more active platform for conducting business using Web services. Complex applications accessing diverse Web services (e.g., a travel package) require an integrated and efficient way to manipulate and deliver Web services. We propose a query infrastructure [21] that offers complex query facilities over Web services. Users submit declarative queries that are resolved through the combined invocations of different Web service operations. As large numbers of Web services are expected to compete in offering "similar" functionalities, we propose an optimization model based on Quality of Web Service (QoWS). We propose to monitor QoWS to measure the Web services fluctuations and rate them. We are also investigating ways to enable life scientists pose complex queries which will transparently search and dynamically compose different bioinformatics tools represented by Web services.

13 From Research to Practice

ICDS has performed several large-scale, multi-year efforts that brings research from ICDS and elsewhere to bear on real-world problems giving students training in database development techniques in a much greater depth than is possible in the classroom.

13.1 Dynamic Maintenance

Knowledge projection aims to revolutionize maintenance operations for Navy vessels by developing tech-

nologies needed to create high performance knowledge bases that represent, capture, analyze, and deliver maintenance knowledge. In this project, we aim at transforming shipboard maintenance into a dynamic, pro-active process that links real-time shipboard troubleshooting actions, diagnostic data, tacit sailor experience, shore-based expertise, and failure resolution directly into knowledge discovery. The project uses XML, web services, and database technologies for its underlying knowledge base support structure, and the knowledge projection concept guarantees a real-time end-to-end solution that extends from the shipboard maintainers to the shore-based experts.

13.2 Digital Government

The core of our research in digital government (a joint project with Virginia Tech) is the development of techniques to efficiently access government services and databases [19]. We propose a framework that enables uniform access to a large number of government services and databases called *WebDG*. Its main features can be summarized as follows. Databases are organized as *distributed ontologies* to accelerate their discovery. Each ontology contains databases that share the same domain of interest. We wrap applications by *Web services* to cater for the dynamic *discovery* and *composition* of welfare programs. To preserve citizens' privacy, we propose a model based on the citizen's privacy preference. Users are granted *privacy credentials* that define their access scope.

References

[1] W. G. Aref and I. F. Ilyas. An extensible index for spatial databases. In *SSDBM*, 2001.

[2] W. G. Aref and I. F. Ilyas. Sp-gist: An extensible database index for supporting space partitioning trees. *J. Intell. Inf. Syst.*, 17(2-3), 2001.

[3] E. Bertino, J. Fan, E. Ferrari, M.-S. Hacid, A. K. Elmagarmid, and X. Zhu. A hierarchical access control model for video database systems. *ACM Trans. Inf. Syst.*, 21(2), 2003.

[4] J. Byun, E.Bertino, and N.Li. Purpose based access control of complex data for privacy protection. In *Proc. 10th ACM Symposium on Access Control Models and Technologies*, Stockhol, Sweden, June 2005.

[5] R. Cheng, Y. Xia, S. Prabhakar, and R. Shah. Change tolerant indexing for constantly evolving data. In *ICDE 2005*.

[6] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *VLDB 2004*.

[7] C. Clifton, A. Doan, A. Elmagarmid, M. Kantarcıoğlu, G. Schadow, D. Suciu, and J. Vaidya. Privacy preserving data integration and sharing. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, Paris, France, 2004.

[8] J. Fan, H. Luo, and A. K. Elmagarmid. Concept-oriented indexing of video databases: Towards semantic sensitive retrieval and browsing. *IEEE Trans. on Image Processing*, 13(5), 2004.

[9] J. Fan, X. Zhu, A. Elmagarmid, W. Aref, and L. Wu. Classview: Hierarchical video shot classification, indexing, and accessing. *IEEE Trans. on Multimedia*, 6(1), 2004.

[10] T. M. Ghanem, R. Shah, M. F. Mokbel, W. G. Aref, and J. S. Vitter. Bulk operations for space-partitioning trees. In *ICDE 2004*.

[11] R. Grossi, A. Gupta, and J. S. Vitter. High-order entropy-compressed text indexes. In *SODA*, 2003.

[12] M. A. Hammad, M. J. Franklin, W. G. Aref, and A. K. Elmagarmid. Scheduling for shared window joins over data streams. In *VLDB 2003*.

[13] M. A. Hammad, M. F. Mokbel, M. H. Ali, W. G. Aref, A. C. Catlin, A. K. Elmagarmid, M. Eltabakh, M. G. Elfeky, T. M. Ghanem, R. Gwadera, I. F. Ilyas, M. S. Marzouk, and X. Xiong. Nile: A query processing engine for data streams. In *ICDE 2004*.

[14] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid. Supporting top-k join queries in relational databases. In *VLDB 2003*.

[15] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid. Joining ranked inputs in practice. In *VLDB*, 2002.

[16] I. F. Ilyas, R. Shah, W. G. Aref, J. S. Vitter, and A. K. Elmagarmid. Rank-aware query optimization. In *SIGMOD 2004*.

[17] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE TKDE*, 16(9), 2004.

[18] M. Kantarcioglu, J. Jin, and C. Clifton. When do data mining results violate privacy? In *KDD*, 2004.

[19] B. Medjahed, A. Rezgui, A. Bouguettaya, and M. Ouzzani. Infrastructure for E-Government Web Services. *IEEE Internet Computing*, 7(1), 2003.

[20] M. F. Mokbel, X. Xiong, and W. G. Aref. Sina: Scalable incremental processing of continuous queries in spatio-temporal databases. In *SIGMOD 2004*.

[21] M. Ouzzani and A. Bouguettaya. Efficient Access to Web Services. *IEEE Internet Computing*, 8(2), 2004.

[22] R. Sion, M. J. Atallah, and S. Prabhakar. Rights protection for relational data. In *SIGMOD 2003*.

[23] J. Vaidya and C. Clifton. Privacy-preserving -means clustering over vertically partitioned data. In *KDD*, 2003.

[24] X. Xiong, M. F. Mokbel, and W. G. Aref. Sea-cnn: Scalable processing of continuous k-nearest neighbor queries in spatio-temporal databases. In *ICDE 2005*.

[25] Y.Koglin, G.Mella, E.Bertino, and E.Ferrari. An update protocol for xml documents in distributed and cooperative systems. In *ICDCS 2005*.

[26] X. Zhu, A. E. X. Wu, A. Feng, and L. Wu. Video data mining: Semantic indexing and event detection from the association perspective. *IEEE TKDE*, 17(5), 2005.

Report on the Ninth Conference on Software Engineering and Databases (JISBD 2004)

Juan Hernández
University of Extremadura, Spain
juanher@unex.es

Ernesto Pimentel
University of Malaga, Spain
ernesto@lcc.uma.es

Ambrosio Toval
University of Murcia, Spain
atoval@um.es

1. Introduction

The ninth edition of the Conference on Software Engineering and Databases (named JISBD after its initials in Spanish) was held in Málaga (Spain) November 10-12, 2004. This conference had its origins in two previous, separate events comprising two scientific communities closely related to the subjects of the conferences: on the one hand the Spanish Software Engineering Conference, held for the first time in Sevilla (Spain) in 1996, and, on the other, the Spanish Conference on Research and Education in Databases, held in A Coruña (Spain), also for the first time in the same year. These two events were joined in 1999, giving rise to the Fourth Conference on Software Engineering and Databases, for the first time with this name, which has been preserved to date.

Since those early years, some other important changes have also happened, particularly concerning the scope and audience of the conference. In the beginning, the two events mentioned above involved only Spanish research groups. This was initially extended to include Portuguese research groups, and later to attract the Latin American scientific community. Every year, the Conference brought together the most important research groups from Spain, Portugal, and Latin America working in Software Engineering and Databases related matters, and since 2002 the conference accepts papers written in Spanish, Portuguese or English, thus widening the call for contributions to the whole scientific community.

2. The Ninth Conference on Software Engineering and Databases (JISBD 2004)

In this section, we will summarize the goals and structure of the conference as well as giving some statistics regarding the submissions and main results of the technical sessions of this latest edition.

2.1. Goals and structure

The main goal of the Conference on Software Engineering and Databases is to provide a scientific and professional forum where Spanish, Portuguese and Latin American researchers and practitioners in Software Engineering and Databases can come together and discuss, share and exchange ideas, problems and experiences and results, and thus foster a profitable collaborative debate among the different sectors and research groups in these areas.

The conference was mainly interested in high quality papers within the scope of the two fields mentioned. These were selected through a rigorous process, although space was also available for good contributions from new groups (short paper sessions), papers describing recent or preliminary results (workshops), and for training in either novel topics or topics of broad interest for the Software Engineering and Database communities, for researchers and engineers (tutorials).

The topics for the Technical sessions of this edition (see the JISBD 2004 web site [3]) were structured in six main sessions dedicated to regular scientific and experience papers, and short papers. Section 2.2 of this report discusses this important part of the conference.

Two invited talks completed the scientific program: the keynote speech "*Towards Active Software*" was given by Dr. Ivar Jacobson, formerly main researcher at Rational and currently working at Jaczone AB. Dr. Jacobson's contributions in the Software Engineering realm are well known. Jacobson emphasized his vision of making the software process active rather than passive. Active software acts as an advisor to the user, it adapts the software to the user's experience and preferences. It can learn from the user and it automates clerical activities. Dr. Jacobson's very revealing talk showed how active software today is becoming a reality.

The main speech within the Databases field was "*Enhancing the Web with DB Technology*", by Dr. Timos Sellis, currently a Full Professor at the National Technical University of Athens (NTUA), in Greece. Timos highlighted that recent advances related to the Semantic Web as well as the explosion of applications requiring

dynamic data extracted from databases both call for several new extensions. In his talk, Timos presented novel models and techniques for managing Web data and discussed why a new kind of proxy that satisfies requests concerning dynamic web objects is needed. He also considered issues of proxies topology, query rewriting, etc.

Another plenary activity was a panel, devoted to presenting and discussing the main Information and Computing Technologies research funding programs within Spain and the European Union.

Finally, the workshops mentioned below completed the program of the conference. The workshops gave authors and participants an opportunity to present and discuss ideas that are topical and innovative in the Software Engineering and Databases areas in an atmosphere that fostered interaction, exchange and problem solving. Hot topics constituted the titles of the workshops approved: MDA, Standardization and Quality, Emerging Solutions for Data Warehousing Systems, Web Services-based Applications, Aspect-Oriented Software Development, Decision Support in Software Engineering, Multi-agent Systems Development and Security in Software Engineering and Databases.

Most of these events were held on the day before the main conference, November 9, 2005.

2.2. Technical Program

Some participation statistics and insights of the technical sessions are provided below.

Submissions, acceptance and participation

121 submissions were received and evaluated. Details of submissions are shown in the pie chart in Figure 1. Each submission was revised by at least three reviewers and finally 38 papers were selected out of the 121 submitted (31,4% acceptance). In addition, 14 papers were chosen for the short paper session.

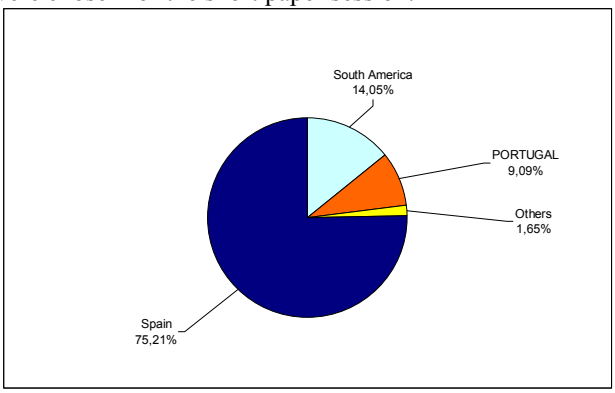


Figure 1. Submissions received at JISBD'2004

Concerning participation, the workshops attracted more than 80 researchers and practitioners, and 140 participants attended the main conference.

Observations on the Technical Program

Although submitted papers covered all the topics mentioned in the CfP, we can point to certain topics that were the focus of most submissions. These observations show us the main fields where the Spanish, Portuguese and Latin America research communities are focusing their work.

In both the Software Engineering and Database areas, the issues that attracted most submissions were: Aspect Oriented Software Development; Quality, measurement and estimation of products and software production processes; Software architectures and Data Modelling; XML and semi-structured data and Semantic Web; and Data mining, Data streaming and Data Warehousing. Below, we summarize some of the most interesting insights on the matters that occupied most space slots: Aspect-Oriented Software Development; Quality, measurement and estimation of products and software production processes and XML and semi-structured data, and Semantic Web.

Aspect-Oriented Software Development.- Undoubtedly, Aspect-Oriented Software Development (AOSD) has recently become a buzzword, gluing together a set of emergent technologies that seek modularizations of software systems. The selected papers that covered this topic of the Conference showed the growing interest of the JISBD community for AOSD technology, emphasizing that aspect-orientation should be taken into account in all the stages of the software lifecycle, from requirements to code level, and applying aspect-oriented reverse engineering techniques later. Besides, aspect orientation techniques are currently being applied in different application domains, such as ambient intelligence, e-commerce, databases and distributed systems and middleware. Hot topics related to this technology, leading to interesting discussions in the technical talks were the integration of aspects in MDA, aspect-oriented software architectures, dynamic weaving, and the use of aspect-orientation on the implementation of design patterns.

Quality, measurement and estimation of products and software production processes.- The contributions presented in this session confirmed the growing interest of the SW and DB communities in requirements, quality and measuring issues. Particular attention was paid to the standards CMM, CMMi, ISO/IEC 15504 (SPICE) and ISO 9001:2000. Problems detected with these standards included the lack of suitable methods for small and middle-sized businesses, the excessive rigidity of some of these approaches and the need to improve the

incorporation and selection of metrics in the organizations. Novel methods and techniques to deal with these problems were proposed and particularized to SMSB as well as to CBD projects, including metrics and independent validation procedures. Applying agile methods in a CMM context, even at Levels 4 and 5 was shown not only as feasible but effective. Regarding DB centric On-Line Transaction Processing (OLTP) systems, a set of general guidelines for the definition and validation of dependability benchmarks was also proposed. Finally, risk management in software projects was also discussed, through a case study, using the SEI-CRM method in a large software project. Most of the results presented in this session combined research and innovation aspects and were backed by experimental, quantitative or qualitative validation.

XML and semi-structured data, and Semantic Web.- Other hot topics dealt with in depth by several conference contributions were XML and semi-structured data, and semantic web. In particular, some sessions were devoted to discussing the possibility of integrating new query languages in web applications, mainly those based on XML documents, and to analysing different systems currently using database technology for knowledge representation on the web. Other interesting conclusions were established as a consequence of some of the proposals presented during the conference, like compression techniques for well structured documents, interoperability of web applications through a specific architecture based on extending mediators with XML, etc.

All the contributions appear in the JISBD 2004 Proceedings [1] and a selection of the best papers has been published in a special issue of IEEE Latin America Transactions [2].

3. Conclusions

This ninth edition has revealed yet again that this is a consolidated and growing event that yearly attracts more researchers and professionals from countries beyond those originally present.

As for the division of the two fields, Software Engineering and Databases, the experience shows that the boundary between these two scientific fields is not always clear. In some cases the same problems are often dealt with using techniques coming from either SE or DB. In others, the same techniques are applied to solve problems related to any of these fields. The decision to join the two original events in 1999, as mentioned in the introduction, has proved, with the passage of time and particularly in this latest edition, to have had a positive effect that has reinforced both of these scientific communities as well as the links between them.

JISBD'2004 also demonstrated the popularity reached by workshops, where participation was very high.

4. Next step: the Tenth Conference on Software Engineering and Databases (JISBD 2005)

The Tenth edition of the Conference on Software Engineering and Databases will be held in Granada (Spain) next September 14-16, 2005. The goals, topics, structure and audience will be similar to those described above for JISBD 2004, but for this new edition the conference will be integrated for the first time within the I Federated Spanish Conference on Informatics, CEDI 2005. Proposals are sought for both commercial and in-house applications, tools and systems, as well as academic and corporate research.

5. References and Web sites

[1] Hernández, J.; Pimentel, E. Proceedings of the Software Engineering and Databases (JISBD 2004), (contributions in Spanish and English)

[2] IEEE LatinAmerica Transactions electronic journal, JISBD 2004 special edition, March 2005, vol. 3, issue 1. http://www.ewh.ieee.org/reg/9/etrans/Marzo2005/March05.htm

[3] JISBD 2004 (Ninth edition): http://jisbd2004.lcc.uma.es/

[4] JISBD 2005 (Tenth edition): http://cedi2005.ugr.es/isydb

[5] CEDI 2005 (multi-conference where JISBD 2005 is integrated): http://cedi2005.ugr.es/ (currently, the main pages of this web are in Spanish; some of the conferences included, - accessible through the "Symposia List" option- also provide the information in English and other languages. JISBD 2005 Call for Papers in English, Spanish and Portuguese.

Acknowledgments: Special thanks are given to the QUERCUS Software Engineering Group at the University of Extremadura (Spain) for their support to the electronic submission and review system both at JISBD 2004 and JISBD 2005.

Report on the 1st International Symposium on the Applications of Constraint Databases (CDB'04)

Bart Kuijpers
Limburgs Universitair Centrum
Belgium
bart.kuijpers@luc.ac.be

Peter Revesz
University of Nebraska
USA
revesz@cse.unl.edu

1 Introduction

The 1st International Symposium on the Applications of Constraint Databases (CDB'04) was held on June 12–13, 2004, just before the ACM SIGMOD and PODS conferences, in the Amphithéatre de Chimie of the Université Pierre et Marie Curie in Paris, France. We acted as program committee chairs and Irène Guessarian and Patrick Cégielski as local organization chairs.

This symposium has brought together a group of around 30 researchers from diverse areas that contributed to both the the application and the theory of constraint databases. It was a continuation and extension of previous workshops held in Friedrichshafen, Germany (1995), Cambridge, USA (1996), Delphi, Greece (1997), and Seattle, USA (1998) as well as of the work in the comprehensive volume "Constraint Databases" edited by G. Kuper, L. Libkin and J. Paredaens (2000) and the textbook "Introduction to Constraint Databases" by P. Revesz (2002).

Since the publication of the paper "Constraint query languages" by Kanellakis, Kuper and Revesz in 1990, the last decade has seen a growing interest in constraint database theory, query evaluation, and applications in a variety of conferences, journals, and books. The symposium opened new directions in constraint database research by addressing constraints over domains other than the reals, by contributing to a better implementation of constraint database systems, in particular to query evaluation, by addressing efficient quantifier elimination, and by describing novel applications of constraint databases.

The technical program of the symposium consisted of 3 invited talks, 10 presentations of selected contributed papers and a lively and fruitful panel discussion. The invited talks were held by Leonid Libkin (University of Toronto, Canada), Joos Heintz (Universities of Buenos Aires, Argentina and Cantabria, Spain) and Andreas Podelski (Max-Planck-Institut für Informatik, Germany). The 10 contributing papers were selected by an international program committee of 26 researchers from a field of 28 submissions. The panel participants included Alex Brodsky (George Mason University, USA), Joos Heintz, Andreas Podelski, Jan Van den Bussche (Limburgs Universitair Centrum, Belgium) and Moshe Vardi (Rice University, USA). The symposium proceedings were published by Springer-Verlag as Volume 3074 of the Lecture Notes in Computer Science series.

2 Invited talks

The invited talks covered a variety of topics in constraint databases.

Leonid Libkin, in his talk "Constraints and Queries over Strings and Trees," discussed constraint databases over discrete domains, such as strings and trees, both ranked and unranked. While early research concentrated mostly on continuous domains (due to applications of constraint databases in querying geographical data), the focus has recently switched to discrete domains. Libkin gave a survey of recent results on decidable constraints over strings and trees that arise from automatic structures, and of query languages based on such constraints.

Joos Heintz, in his talk and paper "Constraint Databases, Data Structures and Efficient Query Evaluation," addressed the difficulty of the effective evaluation of first-order queries, usually involving some form of quantifier elimination and discussed various aspects that influence the efficiency of the evaluation of queries expressible in first-order logic over the reals. The importance of data structures and their effect on the complexity of quantifier-elimination was emphasized and a novel data model that supports data exploration and visualization as well as efficient query evaluation was proposed. Finally, he showed that a particular kind of sample point query cannot be evaluated in polynomial time. This paper was joint work with Bart Kuijpers.

Andreas Podelski, in his talk "Constraint-Based Model Checking of ECA Rules," discussed the automatic verification of termination, confluence and other proper-

ties (safety and liveness) for Event Condition Action rules by combing recent approaches to constraint-based abstraction with query evaluation techniques for constraint databases. This talk was based in part on joint work with Hassan Ait-Kaci.

3 Contributed papers

The technical program was broken down into 4 sessions discussed here.

3.1 Spatial and spatio-temporal data

Spatial databases is a common application area of constraint databases. In recent years spatio-temporal data have been often modeled using constraints. The proceedings contains three technical papers on this topic.

Lixin Li, **Youming Li** and **Reinhard Piltner**, in their paper "A New Shape Function Based Spatiotemporal Interpolation Method," propose a new spatio-temporal interpolation method for 3-D space and 1-D time geographic data, based on shape functions. Instead of only manipulating the time dimension as in the earlier ST product and tetrahedral methods, their new method takes the original approach of combining 2-D shape functions in the (x,y) domain with the (z,t) domain shape functions.

Floris Geerts, in his paper "Moving objects and their equations of motion," deals with the representation of moving objects in databases. Moving objects are usually represented, when possible, through explicit descriptions of their trajectories. The author proposes instead a new data model based on encoding their equations of motion, more specifically by differential equations. He also discusses a query language for this data model.

Sofie Haesevoets, in her paper "A triangle-based logic for affine-invariant querying of two-dimensional spatial data," describes a triangle-based logic in which queries that are invariant under affinities of the ambient space can be formulated. She characterizes the expressive power of this logic and shows it to be equivalent to the affine-generic fragment of first-order logic over the reals. She also presents algorithms for computing an affine-invariant triangulation and covering.

3.2 Applications of constraint databases

Looking at specific applications is important for two reasons. First, they reveal the possibilities of constraint database applications, often applications that can not be done in relational database systems. Second, they test the limits of the current constraint data models and query languages and thereby stimulate their further extensions. The following specific applications raise important issues and provide big future challenges to researchers.

Maria Teresa Gómez López, **Rafael Ceballos Guerrero**, **Rafael Martínez Gasca** and **Carmello del Valle Sevilla**, in their paper "Applying Constraint Databases in the Determination of Potential Minimal Conflicts to Polynomial Model-based Diagnosis", apply constraint databases in the determination of potential minimal conflicts, which can be further used for polynomial model-based diagnosis.

Viswanathan Ramanathan and **Peter Revesz**, in their paper "Constraint Database Solutions to the Genome Map Assembly Problem," address the problem of reconstructing the entire genome sequence of an organism based on overlapping fragments of its genome. They look at several algorithms for this problem. Using extensive computer experiments, they show that their constraint automaton, which can be solved using a constraint database system, is much more efficient in solving the genome map assembly problem than is the common alternative solution based on overlap multigraphs. Even more surprisingly, the average case running time of their solution increases only linearly while the running time of the other solution increases exponentially with the size of real genome data input.

Carson Kai-Sang Leung proposes, in the paper "Dynamic FP-Tree Based Mining of Frequent Patterns Satisfying Succinct Constraints," a new dynamic FP-Tree mining algorithm to mine frequent itemsets satisfying succinct constraints. The proposed algorithm is dynamic, that is, the constraints can be changed during the mining process. Based on a classification of constraints he describes the cases of relaxing and tightening constraints and extensive evaluation results showing the effectiveness of the new approach.

3.3 Query optimization

Query optimization is concerned with making computationally efficient, in space and time, the evaluation of queries. Good query optimization techniques are essential for the implementation of constraint database systems. The symposium had two papers in this area.

Jan Chomicki, in the paper "Semantic Optimization of Preference Queries," discusses the problem of semantic query optimization for preference queries and treats this problem as a constraint reasoning problem. His techniques make use of integrity constraints, and make it possible to remove redundant occurrences of the winnow operator resulting in a more efficient algorithm for the computation of winnow. The paper also investigates the problem of propagating integrity constraints.

Anagh Lal and **Berthe Y. Choueiry** consider in their paper "Improving Join Computation Using Constraint

Processing Techniques" the important problem of efficient join computation during query evaluation. They model the join computation in relational databases as a constraint satisfaction problem, which they solve using their technique called dynamic bundling. With dynamic bundling the join computation can be performed with major savings in space and time.

3.4 The future of constraint databases

Implementation of constraint databases is, of course, a major practical concern. While there are several prototype systems developed at universities and research laboratories, such as the C^3, the DEDALE and the MLPQ systems, there are still no commercial implementations of constraint databases. However, this situation may change in the future, as explained in the following two papers.

Dina Goldin describes in "Taking Constraints out of Constraint Databases" how constraints can be eliminated from constraint databases, in the sense of reducing them to as simple a representation as used in relational database systems and geographic information systems. She proposes a 3-tier architecture for constraint databases, with an abstract layer for the infinite relational extent of the data and a concrete layer that admits both constraint-based and geometry-based representations of spatio-temporal data.

Mengchui Cai, from the DB2 group at the IBM Silicon Valley Laboratory, presents in the paper "Integrating constraint and relational database systems" a way of integrating constraint databases into relational database systems. His main insight is that existing relational database systems can be extended by special functions that call a constraint relational engine at the appropriate places within an extended SQL query, while the constraint data itself can be represented within specialized relational tables. This proposal may lead to a practical and seamless way of integrating constraint data with relation data.

4 Panel Discussion

The symposium was concluded by a panel on the future of constraint databases. The panelists were Alex Brodsky, Joos Heintz, Andreas Podelski, Jan Van den Bussche, and Moshe Vardi, and it was moderated by Peter Revesz.

Moshe Vardi opened the discussion by asking *how theory can be turned into practice*. To illustrate this point he looked at model checking, which is a very successful field with numerous applications that started from an underlying theory of automaton on infinite words and resulted in many algorithms and good heuristics with good complexity. On the other hand, the field of dependencies, which has no clear business problems to solve, studies certain fragments of first-order logic and resulted in many nice papers but remained without implementations and practice. Where is the area of constraint databases going? Will it turn out like the former or the later example? It is important to ask what are its applications, what problems it solves best, and how well compared to other possible solutions. For example, there is a possibility to develop heuristics that may lead to an efficient geographical information system based on constraint database theory.

Jan Van den Bussche mentioned that theory is still important to work on, and that it is impossible to see ahead whether a good theory will or will not have applications. As far as applications, he believes that bioinformatics could be a very fruitful application area of constraint databases.

Andreas Podelski emphasized that model checking in fact has a close relationship with constraint databases. Model checking could be solved by a combination of approximations and constraint database query evaluations. Hence he sees a potential for future growth in combining model checking with constraint databases.

Alex Brodsky described his company's experiences in software development that included in parts constraint processing with ideas based on constraint database techniques. He believes that more involvement of constraint databases in software development would be also possible with further improvement and commercialization of constraint database systems.

Joos Heintz was the last panelist to express his views. He also emphasized the need for combining theory and practice, but he also pointed to the need to be patient. He comes from mathematics to the area of constraint databases, and he believes that compared to many areas of mathematics, constraint databases looks much more promising for future applications and success.

There was a very active participation in the panel discussion by numerous members of the audience. Overall the panel was a lively and stimulating event.

5 Summary

The symposium had a very active program and participation. We were glad to see the symposium bring together many researchers in the field of constraint databases for a fruitful exchange of ideas. We also look forward to a continued growth in the field and to future events in the field of constraint databases.

Report on the International Workshop on Pattern Representation and Management (PaRMa'04)

Yannis Theodoridis[1], Panos Vassiliadis[2]

[1] Univ. of Piraeus, and Computer Technology Institute, Hellas, ytheod@unipi.gr

[2] Univ. of Ioannina, Hellas, pvassil@cs.uoi.gr

1 Introduction

The increasing ability to quickly collect and cheaply store large volumes of data, and the need for extracting concise information to be efficiently manipulated and intuitively analyzed, are posing new requirements for Database Management Systems (DBMS) in both industrial and scientific applications. A common approach to deal with huge data volumes is to reduce the available information to knowledge artifacts (i.e., clusters, rules, etc.), hereafter called *patterns*, through data processing methods (pattern recognition, data mining, knowledge extraction). Patterns reduce the number and size of the original information to manageable size while preserving as much as possible its hidden / interesting content. In order to efficiently and effectively deal with patterns, academic groups and industrial consortiums have recently devoted efforts towards modeling, storage, retrieval, analysis and manipulation of patterns with results mainly in the areas of *Inductive Databases* and *Pattern Base Management Systems* (PBMS).

This report focuses on the International Workshop on Pattern Representation and Management (PaRMa) held in conjunction with the EDBT'04 Conference in Heraklion, Crete, Hellas, on March 18th, 2004. A summary of the papers and the discussions held during the workshop is given.

The main theme for the workshop was "From Data to Patterns". In response to the call for papers, 15 papers were submitted. The Program Committee selected 9 papers based on their scientific contribution, novelty and relevance to the workshop. Due to its obvious database-centric perspective, three papers on modelling and pattern manipulation languages were accepted. A second area of interest involves the management of patterns with a particular focus on pattern similarity and prediction. Finally, following a long tradition in the area of data mining, the workshop program has included three papers on systems and algorithms for pattern extraction.

The electronic edition of the workshop proceedings is available from the CEUR series: http://ceur-ws.org/Vol-96/

2 Languages for patterns

Languages for defining and manipulating patterns are one of the most important issues in Pattern Base (PB) management. Three papers were presented on this topic, all attempting to introduce general approaches for pattern management.

Bertino et al. [2] propose a Pattern Manipulation Language (PML) and a Pattern Query Language (PQL). The PANDA model for pattern representation (based on the proposal of [8]) is adopted. In this approach, patterns are *instances-of* pattern types and *members-of* pattern classes. The model is generic enough to represent any type of pattern by means of a *structure* (e.g., an association rule has a head and a body), *underlying data* that it represents, a *formula* that annotates it with explicit semantics, and statistical *measures* (e.g., confidence and support). The concept of *mining function* (to specify the mining algorithm to be used for extraction) is also introduced. Then, the authors define PML and PQL. PML operations over patterns or classes include insertion, deletion and update. PQL is defined over classes and is closed. PQL operations include projection, selection, identity, shallow/deep value equality, similarity, drill-down and roll-up, decomposition, join, drill-through and covering (with the two last being *cross-over* operators for relating patterns with raw data).

Bouros et al. [3] propose *PatManQL*, a language to manipulate patterns and data in hierarchical catalogs. The paper presents modeling constructs to represent hierarchical catalogs, based on the concept of *Tree-Structured Relations* (TSRs). Then, the PatManQL language is presented, involving operators to manipulate path-like patterns (select, project, Cartesian product, union, intersection, difference over TSRs). Finally, the development of a system implementing the above operators is also given. The prototype includes an interpreter, a query execution engine, a storage mechanism (using XML files and MySQL relational tables) and a graphic result interface.

Rizzi [9] discusses the conceptual modeling of pattern-bases, based on UML. Using [8], the author addresses the issue of how can UML be used and extended to model PBs from the static / functional / dynamic points of view. *Static modeling* describes PBs from a structural point of view, with particular emphasis on expressing complex relationships between patterns. *Functional modeling* represents the processes used to establish the relationship between data and patterns (DFD, UML use case diagrams,

UML activity diagrams). *Dynamic modeling* describes how patterns and raw data are kept in synchronization (state charts, UML sequence diagrams). The overall conclusion is that a broad range of issues related to PBs can be expressively represented without giving up the syntax and semantics of UML.

3 Pattern similarity

Similarity is one of the key issues in pattern representation and management since a number of procedures and tuning mechanisms, including indexing, query optimization, update and synchronization, assume a formal definition of pattern similarity measure. Two papers were presented in this category.

Ntoutsi [6] discusses similarity issues in data and pattern spaces motivated by data mining and PBMS domains. Formally, the measurement of (dis-) similarity is mapped to graph assignment problem, which, although well-researched, is of high complexity (Hungarian method is $O(n^3)$). Experiments on data (100 transactions X 10 items per transaction) and the corresponding pattern sets were shown, uncovering the behavior of similarity measures as well as how similarity in the data space is related to similarity in the pattern space.

Bartolini et al. [1] propose a framework for the comparison of complex patterns. This framework is based on the PANDA logical model for pattern representation [8]. First, the similarity between simple patterns is defined as an aggregation formula between structure and measure components. Then, the similarity between complex patterns (based on simple ones) is defined as the aggregation of similarities between simple components. A prototype was also implemented defining foundation classes that guarantee the generality of the proposed framework. Case studies included a comparison of clustering schemas and a comparison of web sites (reduced to similarity between graphs).

4 Finding 'interesting' patterns

Although the efficient extraction of patterns has been studied extensively in the data mining literature, the detection of interesting patterns is another 'hot' task in the research agenda. Two papers were presented on this topic.

Keim and Wawryniuk [5] address the issue of extracting patterns from data which have categorical and numerical attributes (e.g., census data). After reviewing related work (distance- and mean- based association rules), the authors propose an algorithm that identifies *interesting itemsets*, where 'interestingness' means that the itemset has an impact on the numerical attribute, i.e., distribution different from overall distribution.

Heilmann et al. [4] tackle the problem of finding interesting spatial patterns in network data by exploiting ideas from the visualization and geo-spatial visualization areas. Visualization not only offers a high degree of user satisfaction and confidence but also yields better results. In fact, there exist correlations between data that cannot be expressed easily with some formal language whereas they are straightforward through a visual representation. The authors also presented some example for the effective visualization of network data in an important area of application, the analysis of e-mail traffic.

5 Systems - Applications

The study of patterns in different application domains reflects the need for efficient pattern representation and management. Two papers were presented in this category.

Pelekis et al. [7] propose *Fuzzy-Miner*, a fuzzy system for solving classification problems. Fuzzy-Miner consists of a fuzzification interface (that converts crisp input into fuzzy singletons), a knowledge base (that consists of a data base and a rule base with a set of if-then rules), a decision making logic module (which employs fuzzy if-then rules from the rule base to infer the output by a fuzzy reasoning system) and, finally, a defuzzification interface (that defuzzifies the output). The quality of the system, in terms of classification success, was experimentally evaluated using a real dataset (Athens Stock Exchange data).

Robardet et al. [9] exploit pattern databases for gene expression data analysis. In particular, the authors design inductive querying techniques that extract interesting sets of objects (biological situations) and associated sets of attributes (genes). The methodology utilizes a hierarchical classification method to cluster concepts and then the obtained hierarchy of concepts is visualized.

6 Panel Discussion

Following the general motivation for having a workshop on pattern management, a panel took place at the end of the workshop, in order to summarize the state of the art in this novel research field, as well as the main impressions from the workshop and discuss the research agenda of the topic. In this section we will briefly sketch the main highlights of the discussions; a more detailed report is available [11].

The panel was moderated by Panos Vassiliadis with the participation of Barbara Catania, Daniel Keim, Céline Robardet and Yannis Theodoridis. The panelists addressed the following issues:

- What do you think patterns are? Is it possible to devise a generic model for patterns?
- What do you think are the main "queries" and operations over patterns?
- How would you manage patterns?
- How would you prescribe a research agenda for pattern management?

In terms of discussing the nature of patterns, there was a (naturally expected) reference to the definition of [8] for a compact representation of data that is rich in semantics. At the same time, there was also a discussion of the inductive database approach for simple patterns vs. models. The discussion that took place converged towards two main

results: (a) there exist common characteristics in patterns (like structure, relationship to underlying data and so on) and (b) it is quite hard, yet visionary, to come up with a generic model for patterns.

The difficulty in providing a common, generic set of operations also appeared as the outcome of the second topic. During the discussion, it was pointed out that the way people handle patterns now is dependent on the domain of application. Still, one could possibly identify common characteristics, in the sense that in all application domains, people spend quite a lot of time either observing patterns or combining patterns with data in order to deduce knowledge. At the same time, the opinion that it is pointless to ask for *common*, but for *useful* or *important* operations, was also expressed. A vivid discussion followed, over the nature of the query paradigm and the languages that can be adopted for this purpose.

On the issue of how to build a system to manage patterns, the general feeling can be summarized as follows: (a) building such a system from scratch is not worthwhile; (b) on the contrary, Object-Relational technology should be the basis over which such a system should be built; (c) possibly, a mix of ORDB and semi-structured/XML database systems could outperform pure ORDB technology. An interesting observation was made by more than one panelist regarding the third point: it appears that although ORDB technology can manage patterns, this is not always the most efficient technique to follow

Arriving at the concluding question of the panel, the panelists where asked to express their opinions on the main topics / challenges / opportunities / pitfalls for research in pattern management. The main research areas that were identified were: (a) *visualization*; (b) *operations on patterns* (possibly departing from the traditional relational operations, if this is necessary in order to facilitate typical/interesting user tasks); (c) *modeling and querying languages* (possibly in a QBE fashion); (d) internal *operations of a pattern-base management system*, like similarity algorithms, indexing methods and synchronization (with respect to the changes of the underlying data); (e) support for the whole process of pattern management where people *extract, process and store patterns*.

7 Summary

The lack of database techniques supporting the management of patterns has been the main motivation behind the PaRMa'04 workshop. We believe that the papers presented along with the panel discussions have contributed towards this research direction.

We now summarize the main outcomes of the workshop:

- Although it appears too hard to obtain a generic model and language for all kinds of patterns, it is worthwhile to explore this possibility even for a broad subset of cases (e.g., data mining results). Approaches towards the logical and conceptual modeling, as well as the querying of patterns present a valid research challenge.
- The discovery of pattern similarity and pattern visualization appear to be a couple of approaches that are highly interesting. Both approaches aim at extracting useful knowledge at the pattern space through different means and provide interesting research topics, as well.
- There are several other research opportunities, with the building of systems and the exploration of their internal operations and data representation mechanisms being the most prominent one. Returning to our starting point, the provision of database-like support to the management of patterns is an open, novel and wide area of research.

Acknowledgments

On behalf of the Program Committee we would like to thank all the authors of submitted papers and all the referees for their careful work. The support of the European Commission through the IST/FET Programme and the PANDA Working Group (IST-2001-33058) is also gratefully acknowledged. Finally, we would like to express our gratitude to the members of the Organizing Committee of EDBT'04 for their support in organizing this workshop.

References

[1] I. Bartolini, P. Ciaccia, M. Patella, "A Framework for the Comparison of Complex Patterns", in [11].

[2] E. Bertino, B. Catania, A. Maddalena, "Towards a Language for Pattern Manipulation", in [11].

[3] P. Bouros, T. Dalamagas, T. Sellis, M. Terrovitis, "PatManQL: A language to manipulate patterns and data in hierarchical catalogs", in [11].

[4] R. Heilmann, D. Keim, C. Panse, J. Schneidewind, "Finding Spatial Patterns in Network Data", in [11].

[5] D. Keim, M. Wawryniuk, "Identifying Most Predictive Items", in [11].

[6] I. Ntoutsi, "The notion of similarity in data and pattern space", in [11].

[7] N. Pelekis, B. Theodoulidis, I. Kopanakis, "Fuzzy Miner A Fuzzy System for Solving Pattern Classification Problems", in [11].

[8] S. Rizzi et al, "Towards a Logical Model for Patterns", *Proc. of the 22nd Int'l Conference on Conceptual Modeling* (ER'03), Chicago, IL, USA, October 2003.

[9] S. Rizzi, "UML-Based Conceptual Modeling of Pattern-Bases", in [11].

[10] C. Robardet, R. Pensa, J. Besson, J. Ecois, "Using classification and visualization on pattern databases for gene expression data analysis", in [11].

[11] Y. Theodoridis, P. Vassiliadis (eds.), *Proc. of the Int'l Workshop on Pattern Representation and Management* (PaRMa'04), Crete, Greece, March 2004. http://ceur-ws.org/Vol-96/

Reminiscences on Influential Papers

Kenneth A. Ross, editor

See `http://www.acm.org/sigmod/record/author.html` for submission guidelines.

Kevin Chen-Chuan Chang, University of Illinois at Urbana-Champaign, `kcchang@cs.uiuc.edu`

[Philip A. Bernstein, Alon Y. Halevy, Rachel Pottinger: A Vision of Management of Complex Models. SIGMOD Record 29(4): 55–63 (2000).]

Opening a series of concrete works to follow, this vision paper identifies, motivates, and abstracts the problem of model management. It proposes to support "models" and their "mapping" as first-class constructs, with high-level algebraic operations to manipulate. In the winter of 2000, I was a starting faculty at UIUC, and this paper inspired me immensely at the time when I had to create a research agenda of my own. I have always been interested in information integration, on various topics like query translation and data mapping. The area was exciting to me, as it was full of "real-world" problems. However, it was also not hard to see that these problems seemed inherently messy and their solutions inherently heuristic. Probably because many problems remained unsolved, most research works were only able to address separate topics, without a clear context of an overall application.

This paper changed my view in two ways: First, it shows the relevance of studying related integration problems in the context of an overall system (in this case, a model-management system). Second, it shows the promise of deriving principled mechanisms even for messy problems: By raising the level of abstraction to generally capture the notion of models and their operations, this paper motivates well that a hard problem can be treated with a clean and elegant framework. While both of these seem natural, they were at least unusual for information integration, where problems were often thought to be too hard to seek principles and too early to situate in a system context. Although the paper was a vision paper to be concretely realized, it influenced me in attempting to pursue a system-driven research agenda and striving to seek principled solutions.

Jayant R. Haritsa, Indian Institute of Science, Bangalore, `haritsa@dsl.serc.iisc.ernet.in`

[H. T. Kung and John T. Robinson: On Optimistic Methods for Concurrency Control. ACM TODS, Vol. 6, No. 2, June 1981.]

I first read this paper as a raw graduate student, and was captivated by its refreshingly different espousal of a "Don't worry, be happy" approach towards transaction concurrency control, well before this philosophy entered popular parlance. In fact, it went diametrically against the conventional wisdom at the time that locking, with its "nattering nabobs of negativism" world-view (to borrow a phrase from William Safire), was the concurrency control protocol of choice for database systems. Specifically, the paper proposed a new protocol wherein a transaction's execution was divided into three phases: an "optimistic" (i.e. unrestricted) read phase with private updates, a validation phase to assess the transaction's conflicts, and, finally, a write phase where successful validators transferred their private writes to the public database.

Optimistic concurrency control (OCC) was originally mooted in the context of query-dominant systems, where conflicts are relatively rare. For this environment, the intuition was that OCC's RISC-like approach optimized the average case, as compared to the CISC-like approach of locking, where even read-only transactions suffered from lock maintenance overheads. Subsequently, it was realized that the optimistic strategy had another potent advantage — by lazily delaying conflict resolution to the very end, that is, the validation

phase, it provided great flexibility in dealing with transaction data conflicts, as compared to locking, where decisions had to perforce be made at conflict occurrence time. This flexibility could be gainfully used to support a global perspective of the conflict situation, rather than a localized and incremental view, even to the extent of dynamically deciding the transaction commit order. As a case in point, I was able to use this power of OCC in my own doctoral thesis on real-time databases, to ensure timely scheduling of transactions with firm completion deadlines.

This paper is a classic example of how the best research is contingent on thinking against the grain, and therefore has a timeless inspirational value over and beyond its outstanding technical contributions.

Sergey Melnik, Microsoft Research (USA), melnik@microsoft.com

[François Bancilhon and Nicolas Spyratos: Update Semantics of Relational Views. TODS 6(4): 557–575, 1981.]

Raising the level of abstraction is often the key to finding elegant solutions for hard and elusive problems. In their paper, Bancilhon and Spyratos provide an exemplary study of the view update problem put forth by Dayal and Bernstein in 1978. To get a handle on the "information not visible within the view," the authors treat views as functions over database states and express the view update problem in terms of view complements. Despite the word "relational" in the title, the developed formalization is agnostic about schema and view definition languages, i.e., applies to any contemporary data model, such as object-relational or XML.

The paper inspired me in a profound way. It prompted me to look into other established database problems, such as view integration, query composition, and view selection, and to think of how those can be characterized and generalized in a concise, language-independent fashion. It helped me deepen my perception of metadata management as a research area that combines formal approaches with engineering practice.

Guido Moerkotte, Univeristy of Mannheim, moerkotte@informatik.uni-mannheim.de

[Gérard Huet: Confluent Reductions: Abstract Properties and Applications to Term Rewriting Systems. JACM 27(4), 797–821, 1980.]

When I was a student I did not like to go to lectures. Therefore, I had plenty of time to read many papers in different areas of computer science. Among all the papers I read, one paper was really outstanding: the paper by Gérard Huet.

This paper exhibits plenty of attributes a good paper should have. Let me list some of them. Firstly, it is free of buzzwords. Secondly, it does not mention challenging new applications. Thirdly, the paper is very precise. And, last but not least, it is extremely well-written and exhibits a clarity that I never found again.

Unfortunately, the paper failed to influence me.

Michael Rabinovich, AT&T Research, misha@research.att.com

[D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy: Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. STOC 1997, 654–663.]

By the time this paper appeared I had been working on the WWW infrastructure for couple years already. At that time, client demand was outpacing server and networks capacity, and techniques to alleviate this

problem, such as caching, were actively pursued by researchers, myself included. One particular problem I was considering was locating cached objects in a distributed caching platform. An attractive approach to this problem is to use a globally-known hash function to map an arbitrary URL to one of the caches, so that everyone would immediately know which cache is responsible for which portion of the URL namespace. Unfortunately, with most of demand concentrated among a small number of URLs, this scheme can create serious load disbalance among the caches. Worse, an addition or deletion of even one cache can lead to drastic redistribution of the URL namespace among all the caches, forcing them to go through the warm-up phase again. Karger's et al. paper described a beautiful solution to this problem. Not only did their idea influence my thinking on the immediate problem at hand, but it became one of the standard techniques in my "toolkit" for my other research in the distributed systems area.

The impact of the consistent hashing idea extends beyond my own work, it has proved influential in the general distributed systems research. For example, it provided a foundation for the Chord peer-to-peer network, and through Chord, it has had a significant influence on the p2p research. It was also a starting point for Akamai, one of the few Web infrastructure companies that successfully survived the .com bust. I consider consistent hashing to be one of the most elegant ideas in the distributed systems area, and Karger's et al. paper to be a fine example of theoretical work with a significant practical impact.

Arnon Rosenthal, The MITRE Corporation, `arnie@mitre.org`

[Michael Siegel and Stuart E. Madnick: A Metadata Approach to Resolving Semantic Conflicts. VLDB 1991, 133–145.]

The paper contains two important ideas that have influenced both my research and several MITRE "grand visions" for data sharing. First, it suggested that it is important to capture receivers' interface requirements explicitly, rather than just harvest the queries they issue. The power of this simple idea is discussed in the last paragraph. Second, it provided context rules to describe attribute representations (e.g., simple assertions that the field FuelUnit describes the units for TankerContents, and inferred context: if CUSTOMER.ADDRESS.NATION= "USA" then the CUSTOMER.PRODUCT.PRICE has currency = "US dollar"). Using this sort of knowledge plus libraries of scalar transforms, their group built mediators that generated conversion programs. Over the next decade, I started creating mediators, and ended learning hard lessons about applying such technologies in large enterprises.

We learned that simple context mediation of attributes (when schemas match) was too small a unit of technology insertion. Administrators needed to manually map the context rules to the receiver's schema, an unacceptable burden. Siegel and Madnick's follow-on efforts (Goh et al., ACM TOOIS, 1999) provided a technical solution, but they captured knowledge in an exotic logic known only to a handful of researchers; our customers need a standards-based solution.

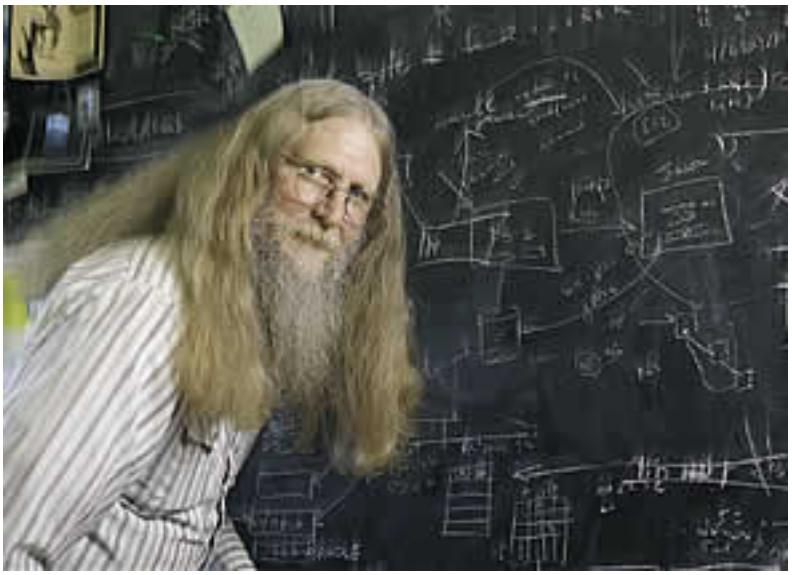
Despite these technology transition challenges, the basic idea of context mediation remains compelling. We used it to provide an alternative (simple context rules in OWL) versus current practice of documenting interface semantics in MS Office. We now believe that context mediation is a big win (over hand-coded transforms) if one has structural mappings and if one is mapping many sources to a receiver (or vice versa). We hope that in the near future, high-end integration systems (e.g., IBM's CLIO) will incorporate context mediation.

We also had successes, greatly simplifying several generations of data-sharing visions (large-scale vaporware) for our customers. We applied Siegel and Madnick's first insight (parallel descriptions for sources and receivers) for aspects such as: push versus pull, sending whole tables versus deltas, data quality requirements, data structure encodings, and protocol choice. This work (and also, mediators from MCC and ISI) contributed to the Department of Defense's dropping its vision of a monolithic standard, and managing instead in terms of diverse communities of interest (Rosenthal, et al., SIGMOD Record, December 2004).

Bruce Lindsay Speaks Out

on System R, Benchmarking, Life as an IBM Fellow, the Power of DBAs in the Old Days, Why Performance Still Matters, Heisenbugs, Why He Still Writes Code, Singing Pigs, and More

by Marianne Winslett



Bruce Lindsay photo by Tom Upton

Welcome to ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today we're in San Diego at the 2003 SIGMOD and PODS conferences. I have here with me Bruce Lindsay, a member of the research staff at IBM Almaden Research Center. Bruce is well-known for his work on relational databases, which has been very influential both inside and outside of IBM, starting with his work on the System R project. Bruce is an IBM Fellow and his PhD is from Berkeley. So Bruce, welcome!

Thank you.

Bruce, I've heard you quoted as saying that relational databases are the foundation of western civilization. Can you expand on this point?

I believe that their adoption in business, government, and education has enabled the progress we've made in productivity, and just made things generally better. I believe that if the relational databases were to stop right now, you would be stuck in San Diego until they got going again. Without these tools to manage the complex affairs of government, business, education, and science our progress would be really much slower.

When I taught introductory database courses years ago, I would always tell my students your famous remark that three things are important in the database world: performance, performance,

and perform ance. Some people would say that this is no longer true today. Would you suggest a different mantra for today?

Well, I think that the remarkable progress in processing and storage makes it possible to meet many requirements with rather modest systems. But faster performance at any level raises everybody's boat on the same rising tide. If the database can be made to run faster on a small system, then you don't have to buy as much hardware to get the job done. In fact, performance is a critical measure of what you're doing. It's like house cleaning: you just have to do it, because if you don't, you can make silly things happen.

Like what?

Like algorithms that take an order of magnitude too long. The mantra "performance, performance, performance" is still important to maintain, because that's where we compete with our competitors and that's where our customers are always happier to go faster, sooner, better.

I think there is another aspect of databases that we often tend to ignore: the utilities, the stuff around the edges. You don't think about it much, but without the utilities you can't really make anything happen. Generic utilities help you load data efficiently, do efficient backups, and manage the organization of your databases efficiently.

Do you see any research issues in the peripheral utilities, or is that more a matter of just good engineering to keep their performance up?

There are tremendously challenging issues in the area of utilities, especially in our 7 by 24 world today, where there's no downtime to do backups anymore. Making all of these operations work online and without disrupting the ongoing production work is indeed an engineering and a design challenge.

Is there anything that the research community should be working on too, or have we done our part there?

I think the research community has done a very good job of ignoring these issues.

I've seen papers on bulk loading.

I've seen some too, but they were not really talking about *online* loading, as I recall. The idea of adding, say, 10 GB of records to a table while continuing to process queries on that table is something that I've never seen much in academic literature.

So, PhD students who are looking for topics, you heard it here!

Is there anything you would like to tell us about TPC-C?

Don't do this at home.

Anything else, or is that a succinct summary of the whole thing?

Relational databases are the foundation of western civilization

Well, the benchmarking business is dirty work. The idea is to get the numbers by hook and by crook. And the good news is that about 40-70% of the stuff we do in performance tuning actually ends up helping end users. But much of what we're doing in the TPC-C realm these days is in a performance range that goes beyond what any user is doing. The customers wish they had that many transactions to run, but their businesses just aren't that big. So I think it's a mixed bag.

Of course, any benchmarking effort is used in two different ways. One is marketing and sales, but I think the rather more important and interesting aspect of standard benchmarks is to compare yourself against the competition. If you're the only one out there running a particular workload, you really have little way of knowing whether you're doing a good job or a bad job on it. So from my point of view as a researcher, a benchmark offers a measurement point to help me understand where I'm doing well and where I'm not doing so well.

Is there any particularly sneaky but still totally legal aspect of TPC-C tuning that you would like to mention?

Well, we do things that are very, what should I say? Intense. We get down to the level of worrying about the physical column order in the table so the reference columns are near each other, minimizing cache misses during fetching. This is feasible in the TPC-C benchmark because there are only five tables and only ten to fifteen columns in each table. In a more realistic application, where there are many more queries to be considered, the tables are typically much, much wider, in the 80 to 100 column range; and there are dozens if not thousands of tables. Then this kind of analysis is no longer practical.

Tell us about Heisenbugs.

Oh, Heisenbugs. One drunken night in Berkeley, Jim Gray and I were working away on an operating system project, the CAL-TSS system [an NSF-sponsored research project in the 1960s that developed a capability-based operating system for the CDC 6400]. We'd naturally had the kind of bug that the more you look at it, the less you can find it. Every time you turn on a measurement, the bug goes away and you can't see it. I'd been studying a little bit of physics at that time, and I was struck about what they were saying in atomic physics about the relationship between the measurements and the thing being measured. And so it struck me that we were somewhat in the same game: when you looked, the bug went away, because the measurement or the observation affected the phenomena you were trying to see. Hence the term “heisenbugs”.

I have heard that colloquium speakers at IBM Almaden used to be in terror of you, that you patrolled the seminar rooms like a lion, but that you've given up this practice in the last few years. Is that true? And if so, what prompted the change?

Well, I think that probably the reason that I'm not doing as much is that I’ve gotten lazy, or else I don't understand as well what they're talking about. But I do admit that I have little sympathy for shallow arguments.

Who will keep the speakers honest then if you're not down there?

I'm hoping somebody else will step up to that role.

At the time that System R was built, it was impossible to realize what an impact it would have on the world, for better and for worse. With perfect hindsight, what do you wish had been done differently in System R?

Well, there are a few little things that I think we made mistakes on. I think one of the biggest mistakes in the SQL language was the "select *" notion. The "select *" constructs were developed to provide ease of use, but they really have been an implementation nightmare. "Select *" means select all the columns, but in what order?! The relational model says nothing about what the order of the columns might or should be, so you're really getting down to a physical storage issue here. More problematic, when you create a view as "select * from table where predicate", what happens when I alter the table and add a column? What does the view mean now? Does it have more columns than before, or not? We have to figure out what to do about those views when their underlying tables are changed. In fact, we have to make the meaning of the views be unchanged by the update, as otherwise existing applications would break. So I think that this is one example of not quite thinking it through all the way.

Three things are important in the database world: performance, performance, and performance

In hindsight, what do you view as the key elements of System R's ultimate success?

Oh, I have a revisionist's history story on that. I think there are three elements of the System R prototype and its deployment in test environments that had an effect on the eventual adoption of relational data storage systems. The first one is the obvious one that everybody suggests: the invention of a nonprocedural query specification was a tremendous simplification that made it much easier to specify applications. No longer did you have to say which index to use and which join method to use to get the job done. This was a tremendous boon to application development.

I like to tell the story of what it was like to be a DBA in those days, because the application backlog was huge. It was a good deal to be a DBA, because you controlled what got done next. It was necessary for anybody that wanted anything to get done to bring you gifts, you know, bottles of things. And therefore it was a good deal to be a DBA, because of this application backlog.

I think the ability to more simply specify how or what it was you wanted retrieved from the database, instead of how to do it, was a tremendous advantage and has paid off enormously. But the real reasons that the relational system won have nothing to do with the relational model; they have to do with the fact that the early prototypes, both System R and Ingres, supported ad hoc queries and online data definition. You could try out a query right away, rather than typing it into your application, compiling the application, running it, and having a failure, usually a syntax error. Even if there were no syntax errors, you still had the advantage that you could look at the query answers and see if you had the right tuples. Ad hoc online query execution was something that was unheard of in the database community at that time, so this was a tremendous boon to application developers and to people who were browsing the data.

Online data definition was also a great boon to the development of applications in database systems. Previously, if you wanted to add a new table or add an index or add a column to a table (or *dataset*, as they were called in those days), you had to shut down the database system, invoke some compiler-like tools to define the new data entities, start up the database system, and try it

out. This could take hours. With online data definition, it's a matter of just typing the right mumbo jumbo and the system says, “Sure, you got it, try it!”

[Note: The reader may enjoy comparing Bruce’s answers to the previous questions with Pat Selinger’s answers in her interview in the December 2003 SIGMOD Record.]

When the non-relational people saw what was happening, couldn't they somehow retrofit some tools to work with their model, so they could do these things online too?

It goes deeper than tools. It goes right to the heart of the system in very fundamental ways, such as maintaining the coherence between the metadata that the system uses at run time to figure out where things are and which way's up and which way's downand the actual use of the system. So a difficult and rather successful effort in the System R project was to figure out ways to synchronize the modification of metadata without impacting or interrupting the ongoing work involving other data objects.

Bruce, some of my previous interviewees have said that the core areas of database technology are played out as a research area. You've spent your whole career working on core areas, so I thought you might like a chance to comment on this assessment.

I think we've come quite a long way in many areas---certainly in the areas of transaction management, recovery, concurrency control. I think we have those under control now, and we have a pretty good feeling for what the fundamentals are in those areas. For areas further out from the core, such as query processing---we continue to evolve the query processing algorithms. We find new indexing methods and consider them. There's still polishing to do in the area of index utilization. There are many more things we could do with indexes than we're doing now.

But I think the action is moving to the periphery of the database. The area of manageability and usability in the database has become enormously important. The complexity of the systems is probably not much greater than the complexity of the systems they replaced, but the cost of people that can deal with that complexity has doubled in the meantime. So our ability to reduce the human expertise needed to keep these systems running, fed, and happy is very important. The ability of DBAs to manage more easily, to have fewer knobs and fewer buttons but still get the job done, is very important.

I must not go overboard here. After all, I've heard people say to me, “I would like to have a database with no configuration parameters. That way I wouldn't have to hire a DBA and pay somebody to tell me how to tune them.” And that's certainly possible. Let’s compare a cockpit of a 747 with the cockpit of a Cessna. You say, “Gee, the 747 stinks. Too many knobs and dials in there! Nobody can operate this darn thing.” But, darn it, 747s can do things that Cessnas can't. And that's probably part of the reason we have so many knobs and dials. Because we apply relational technology to such a wide range of applications, from decision support to scientific analysis and transactions, we need to have mechanisms to tailor the installation to the needs of the particular applications.

Do you think a self-tuning system could do that tuning automatically?

I think we're making progress, but we're getting into that dangerous realm of AI and machine learning where I think the successes in our industry have been few and far between.

You have said that important issues and tensions remain between object oriented and Java bean applications and their use or misuse of relational persistent stores. Can you expand on this comment?

The difficulty with the object oriented applications is that these programming systems deal only with whole objects. Persistent whole objects may have a hundred or two hundred attributes. The use of that object in an application may involve only a few of its attributes. The current interfaces that we use, partly due to weakness in the object oriented application programming environments, don't seem to be able to focus the access to the persistent store towards the attributes that are of interest. We've actually had some customers with major application programs with the following problem: whenever one attribute of the object, Java bean, whatever it is, gets updated, they immediately update all of the attributes---to the value they had before, of course. Well, at least that works---we have the right data in the persistent store! But there's a cost to be paid in performance, performance, and performance.

What do you think they need to do to fix that problem?

I think it will require quite a lot of engineering to deal with partially materialized objects in the object oriented world, but they certainly could do a lot better job of realizing which parts of the object have been updated and mapping them back to their corresponding storage locations.

One of the biggest mistakes in the SQL language was the "Select *" notion

Let me see if I've got this right. You're an IBM Fellow; therefore the road is paved with gold for you at IBM. You can work on anything you want, you have your own pot of money to spend as you like and people bow down before you as you approach. Right?

I don't think it's quite that grand! There is this widespread misconception that we actually have a budget. We don't. Last year, for the first time, they gave us each $5,000 to spend as we wish. So big deal!

The ability to choose your own area of research, though, is quite a privilege. I've enjoyed being able to decide what I thought was important to work on and spending my time in those areas.

The idea of bowing down is totally silly. At IBM and elsewhere, I believe you have to interact with people on a basis of what they are and what's happening. I got to where I was by paying no attention to what anybody's position was but paying a lot of attention to what they knew and how good they were at explaining it to me.

Can you correct the flaw in the following chain of logic? The 56 IBM Fellows are the people who have made the largest technical contributions to IBM. Therefore they are likely to have unique insights into what IBM should do to be successful in the future. Therefore they should take on management roles to expand their spheres of influence. Therefore you should be a manager.

Oh, I think that making me a manager is like teaching a pig to sing: it doesn't work and it annoys the pig. I believe that the most foolish thing that IBM could do would be to encourage their senior technical staff to focus on management. My experience is that technical progress is made in our area only by teamwork, because the things that we attempt are too big for one person to do alone.

And my experience is that I interact best with the technical team when I'm part of the technical team and I'm doing the same work that they're doing, and whatever leadership or ideas I can provide are just icing on the cake. Technical people should do technical work. They should be at the bench fussing with the chemicals, or shoving the code around, because that's where we'll learn what's really going on and can make the biggest contributions.

What about the other Fellows in the database area who are in some part of management?

Well, I think that there are only two: Pat Selinger has been moving up in management because she's tremendously good at working with people. She can get people to do whatever she wants them to. I don't know how she does it. She did it for years with me, and it's a complete mystery. And Hamid Pirahesh is stepping into the shoes of managing the database research at San Jose Almaden Lab because somebody had to do it. But I believe that the idea of pushing technical people into management is really a very bad idea.

As I was preparing for the interviews this year, Jim Gray suggested that I ask Pat Selinger about how she managed difficult people like Jim and you. More generally, in your view, what is the role of a manager in managing people like you and Jim?

Stay out of the way, and keep bad things from happening to us, and calm down the people we anger.

Is it true that you thrive on the adrenaline that comes from writing code?

Oh, yeah. That's the real contribution, to actually build something. There's nothing more fun than finding the last bug---ha, ha---and developing an algorithm and getting it to work. That's really where the rubber meets the road. All the rest is just talk.

But doesn't code-writing narrow your sphere of influence?

Not at all, because the people that I need to have influence with to get my job done are the people that write code. Because what do I deliver at the end of the day? A working product. And I can't provide the leadership needed to motivate the people who have to work with me to build this product, or to get them to accept any of my ideas, unless I can demonstrate to them that I can do what they are doing.

What led you to join industry and to stay at IBM, as opposed to spending some time in an academic position, joining a start-up, or working for several companies?

My decision when I got my degree was whether to pursue the academic path or an industrial path. My concern with the academic path is that you are not really a researcher very much of the time. You are a teacher, and I do like teaching. Teaching takes a lot of time. But you are also an administrator, and you are also a funding agency. What I noticed at the university is that the faculty spend an inordinate amount of time not doing research and not doing teaching, Instead they are doing management tasks, funding tasks, and so on. I felt if I was going to be in research, I would like to be somewhere where the funding was assured and the research projects were chosen on their own merits.

Now why not change companies? Well, basically I think that that’s because I'm a coward. It took me years and years to get any credibility, because of the way I look. So I had to overcome this, and having gained credibility, I said, why should I start over? I really believe that if you are

going to work with people in a team and accomplish things as a group, you must have mutual respect. You just can't walk into a room and say, “My name is Bruce Lindsay, I am an IBM Fellow, you must respect me.” You have to earn the respect by working with the people for long enough. Word of mouth sometimes helps: “Oh, he's all right. He won't actually eat you,” that kind of interaction.

Did Jim Gray cut off his hair before or after he left IBM?

He actually cut his hair before he came to IBM. At Berkeley we were working together, and Jim Gray's hair was pretty long. Then he got a job at IBM and we started to notice that his hair was getting shorter. Not all of a sudden did it get short; it got shorter bit by bit, and by the time he went off to work at IBM Yorktown, his hair was the normal length for that era. We actually had a cartoon (drawn by Davd Redell) attached to a plan in the office that depicted Jim Gray with shorter and shorter hair.

If you were just finishing your PhD in computer science today, what would you do?

Oh, I'd try to get a job.

An industrial one?

I think an industrial job, yes. You asked earlier why I didn't join a start-up. I believe some of that work is very exciting, but also I believe people need to have a life. Eighty hours a week were not appealing to me.

Making me a manager is like teaching a pig to sing: it doesn’t work and it annoys the pig

Now that multi-disciplinary research topics have become extremely common, are single area people becoming obsolete?

I think you do have to have some ability and understanding of a diverse number of fields. I am a little bit disappointed that some of the database people, new ones that I run into, really don't know very much about current directions in microprocessor architecture. You may say that microprocessor architecture is not relevant. I think it is. But more and more we really have to go outside the computer science area. We need to have expertise in physics or chemistry or life sciences to understand the problems that are important to work on and to bring our computer science skills to bear on solving them. After all, a solution of a problem in computer science, such as a better compiler, is not a solution to anybody's problem. The problem is only solved when a program that was written using that compiler solves a molecule or sends you your Visa bill.

Besides trying to move the company, does an IBM Fellow have increased responsibilities in the area of mentoring?

Oh, absolutely. It’s actually everybody's job to be a mentor, but Fellows should be good at it because they have demonstrated that they have the confidence to move forward and to do bold things. And that's basically what you're trying to teach somebody when you're mentoring them: stand up for yourself, speak out, and move forward.

If you magically had enough extra time to do one additional thing at work that you're not doing now, what would it be?

I think I would work on indexing a little harder.

What aspect of indexing?

Oh, there are many aspects of indexing that I think need improvement. I think we can do a better job in the searching. I think there are exciting things to do in multi-dimensional indexing. I think there are a lot more advanced ways that we can use database indexes for indexing on columns that contain sequences or XML data, and for partial indexing. With partial indexing, we would index only some of the rows, based on some predicate. For example, we might not index the NULL values, or we might index only those salaries greater than $100K.

If you could change one thing about yourself as a computer science researcher, what would it be?

Be taller?

But what advantage would that give you?

I could look Mike Stonebraker in the eye.

Anything else?

If I could be better at understanding and proving theorems, I would be very happy.

What kind of topics would you prove your theorems about?

Well, I've been working on an algorithm for peer-to-peer replication, and I can't find a proof that's correct.

Thank you very much for talking with me today.

Building Data Mining Solutions with OLE DB for DM and XML for Analysis

Zhaohui Tang, Jamie Maclennan, Peter Pyungchul Kim

Microsoft SQL Server Data Mining
One, Microsoft Way
Redmond, WA 98007
{ZhaoTang, JamieMac, PeterKim}@microsoft.com

ABSTRACT

A data mining component is included in Microsoft SQL Server 2000 and SQL Server 2005, one of the most popular DBMSs. This gives a push for data mining technologies to move from a niche towards the mainstream. Apart from a few algorithms, the main contribution of SQL Server Data Mining is the implementation of OLE DB for Data Mining. OLE DB for Data mining is an industrial standard led by Microsoft and supported by a number of ISVs. It leverages two existing relational technologies: SQL and OLE DB. It defines a SQL language for data mining based on a relational concept. More recently, Microsoft, Hyperion, SAS and a few other BI vendors formed the XML for Analysis Council. XML for Analysis covers both OLAP and Data Mining. The goal is to allow consumer applications to query various BI packages from different platforms. This paper gives an overview of OLE DB for Data Mining and XML for Analysis. It also shows how to build data mining application using these APIs.

Categories and Subject Descriptors

Data Mining Languages and Industrial Standard

General Terms

Standardization, Languages

Keywords

Data mining, database, SQL Server, OLE DB for Data Mining, XML for Analysis, Web Service.

1. Introduction

Data Mining receives more and more attention these days. Data mining, as we use the term, is the exploration and analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns and rules. These patterns and rules will help corporations to improve their marketing, sales, and customer support operations through better understanding of their customers. Through the years, corporations have accumulated very large databases from applications such as ERP, CRM or other operational systems. People believe there is untapped value hidden inside these data; to get these patterns out requires data mining techniques.

SQL Server 2000 has introduced data mining features for the first time. Two scalable data mining algorithms are included: Microsoft Decision Trees and Microsoft Clustering. Both algorithms are developed and patented by Microsoft Research.

The Analysis Services of SQL Server 2000 and SQL Server 2005 have two components: OLAP and data mining. Both data mining and OLAP are important techniques for data analysis, but the related technologies are different, with data mining providing automatic or semi automatic pattern discovery. Data mining combines techniques developed in artificial intelligence, database and statistics, while OLAP is mainly based on SQL plus certain aggregation techniques. The term often employed in OLAP is multi-dimensional database, or a so called data cube. For example, a sales cube can be built on top of the sales table that has a number of dimensions such as Product, Region, and Time. Each cell in the cube gives the aggregated sales value for a particular product, region and time period.

Data mining and OLAP are complementary as they are both analytical tools. For example, in the customer dimension of a sales cube, there are lots of members, so it is very difficult to find customers' buying patterns. Data mining techniques can be applied to analyze this dimension to find out what are the clusters among the customers based on customer member properties and measures. SQL Server Analysis Services has advanced features that bridge data mining and OLAP, however, we will not address details of these features in this article.

2. OLE DB for Data Mining

The data mining industry today is highly fragmented, making it difficult for application software vendors and corporate developers to integrate different knowledge-discovery tools. We can consider the current data mining market similar to the database market before SQL was introduced. Every data mining vendor has its own data mining package, which does not communicate with other products. For example, a customer is interested in a decision tree algorithm from Vendor A and has built the data mining application based on Vendor A's package. Later on, the customer finds the time series algorithm from vendor B is also very attractive for prediction tasks. He now faces a difficult situation as product A and B has no common interface and he has to restart the whole project from the beginning.

Most data mining products are horizontal packages and are difficult to integrate with user applications such as customer care, CRM, ERP. With the help of the OLE DB for DM Specification, any data mining algorithms can be accessed through OLE automation, which can be easily embedded into any consumer applications.

Another problem of most commercial data mining products is that data extraction from a relational database to an intermediate storage format is necessary. Data porting and transformation are very expensive operations. Why can't we do data mining directly on relational databases where most data are stored?

To solve these problems, Microsoft initiated the OLE DB for Data Mining (DM) Specification with more than a dozen independent software vendors (ISVs) in business intelligence. Its goal is to provide an industry standard for data mining so that different data mining algorithms from various data mining ISVs can easily plug into consumer applications. Those software packages that provide data mining algorithms are called Data Mining Providers, while those applications that use data mining features are called Data Mining Consumers. OLE DB for DM specifies the common interface between Data Mining Consumers and Data Mining Providers. Figure 1 shows the basic architecture of OLE DB for Data Mining. It allows consumer applications to communicate

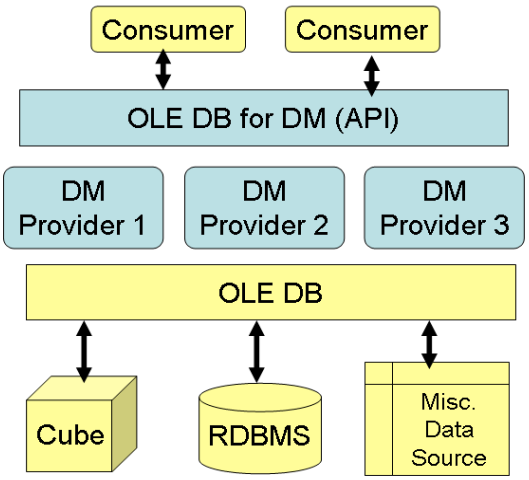


Figure 1. Architecture of OLE DB for Data Mining

with different data mining algorithm providers through the same API (SQL style).

2.1 Data Mining Model

A *data mining model* is a new concept introduced in OLE DB for DM. A data mining model can be considered as a relational table in the sense that it has a list of columns with different data types. Some of these columns are input columns while others are predictable columns. A data mining model is a container. However, a data mining model is different from a relational table as it doesn't store raw data, rather it stores the patterns that data mining algorithms have discovered in some source data. A mining model also has to specify the data mining algorithm with which it is associated and (optionally) a list of parameters. To create a data mining model, OLE DB for DM adapts the table creation syntax of SQL. The following example creates a mining model to predict credit risk level based on customer demographic information using Microsoft Decision Trees algorithm:

```
CREATE MINING MODEL CreditRisk
(
 CustomerId     long     key,
 Profession     text     discrete,
 Income         text     discrete,
 Age            long     continuous,
 RiskLevel      text     discrete  predict,
)
USING [Microsoft Decision Tree]
```

2.2 Training a Mining Model

When a data mining model is created, it is an empty container. During the training stage, the data mining algorithm analyzes the input cases and populates the patterns it has discovered to the mining model. According to OLE DB for DM Specification, training data can be from any tabular data source as long as there is a proper OLE DB driver. It does not require users to export data from a relational source to any special intermediate storage format. This largely simplifies the process of data mining. To be consistent with SQL, OLE DB for DM adopts the syntax of data insertion query. The following sample trains the CreditRisk mining model with the data stored in the customers table of a SQL Server database.

```
INSERT INTO CreditRisk
(
 CustomerId,Profession,Income, Age,RiskLevel
)
OPENROWSET('sqloledb', 'sa'; 'mypass'; '' ,
'SELECT CustomerID, Profession, Income,
        Age, Risk
 FROM Customers'
)
```

The Openrowset command can access remote data from an OLE DB data source. SQL Server 2000 ships OLE DB drivers for SQL Server, Access and Oracle. Data does not have to be loaded ahead of time. This is called *in-place mining*. The training process may take some time as during this stage, the data mining algorithm goes through all the input cases and does some complicated calculations. After training, the data mining algorithms find patterns, which are persisted inside the data mining model. Users can browse the mining model to look at the discovered patterns, or use the trained mining model for prediction tasks.

2.3 Prediction

Prediction, also known as scoring, is an important data mining task. It requires two elements: a trained data mining model and a set of new cases. The result of prediction is a new recordset that contains values for predictable columns as well as other input columns. The overall process is very similar to relational join. Instead of joining two tables, prediction joins a data mining model with an input table. Thus we introduce a new concept called a *prediction join*. The following example shows the syntax of a prediction join:

```
SELECT
  Customers.ID,
  CreditRisk.RiskLevel,
  PredictProbability(CreditRisk.RiskLevel)
FROM CreditRisk PREDICTION JOIN
        Customers
ON   CreditRisk.Profession =
     Customers.Profession AND
     CreditRisk.Income =
     Customers.Income AND
     CreditRisk.Age =
     Customers.Age
```

The query returns a record set with three columns: Customer ID, RiskLevel and Probability. OLE DB for DM predefines a set of prediction functions; most of them return additional statistics for the prediction.

Usually prediction can be done on the fly. It is also possible to do prediction on single case instead of a set of new cases. We call these prediction queries singleton queries.

OLE DB for DM has also defined a list of prediction functions that can be included in the select clause of the prediction statement. These functions will return the probability of the predicted value, histogram information about other possible values and related probabilities, top counts, cluster id, etc.

2.4 Schema Rowsets

The schema information specified in OLE DB is based on the assumption that providers support the concepts of a catalog and a schema. Schema information can be retrieved in predefined schema rowsets. Schema rowsets are global tables for meta data. In OLE DB for Data Mining, we have predefined a list of schema rowsets. These schema rowsets help applications to dynamically discover the available data mining algorithms and their parameters, existing mining models and their contents. For example, after training a mining model using decision tree algorithms, the content schema rowset contains the tree nodes information. Consumer applications can display the tree graphically based on the content schema rowset.

3. Data Mining in SQL Server

In this section, we look at the data mining component of SQL Server 2000.

3.1 Components Architecture

Microsoft has implemented an OLE DB provider for Data Mining based on the OLE DB for DM Specification. The provider includes two data mining algorithms developed by Microsoft Research. The SQL Server data mining provider is part of Analysis Services 2000 (previously called OLAP Services in SQL Server 7.0). Similar to Microsoft OLAP Services, the data mining component in SQL Server 2000 is mainly targetted to DBAs. There is no sophisticated GUI component for data mining, and Microsoft is working closely with ISV partners to build these general consumer tools. However, there are a few data mining GUI components for data mining in the Analysis Services. These components include a model creation wizard, model editor, model content browser, and DTS task for prediction. Figure 2 shows the major components in Analysis Services 2000.

3.2 Data Mining Algorithms

We now describe the data mining algorithms included with the SQL Server data mining provider: Microsoft Decision Trees and Microsoft Clustering. Decision trees are widely used for classification tasks. Unlike other classification algorithms such as nearest neighbor, neural networks, regression-based statistical techniques, decision trees can handle high dimensional data and the rules found can be more easily understood.

In order to achieve highly scalable classification over a large database, we implemented the execution module proposed in [2]. The execution module batches execution of multiple queries for the classification client in a single scan of data to compute statistics, and appropriately stages data from server to client file system and to client main memory. It also has an optimization facility to tradeoff the cost of scanning data at the server versus use of in-memory operations depending on the available client memory size.

The Microsoft clustering algorithm is a scalable implementation of the Expectation-Maximization (EM) algorithm[1]. Unlike distance-based algorithms such as K-Means, EM constructs proper statistical models of the underlying data source and naturally generalizes to cluster databases containing both discrete-valued and continuous-valued data. The scalable method is based on a decomposition of the basic statistics the algorithm needs: identifying regions of the data that are compressible and regions that must be maintained in memory. The approach operates within the confines of a limited main memory buffer and requires at most a single database scan. Data resolution

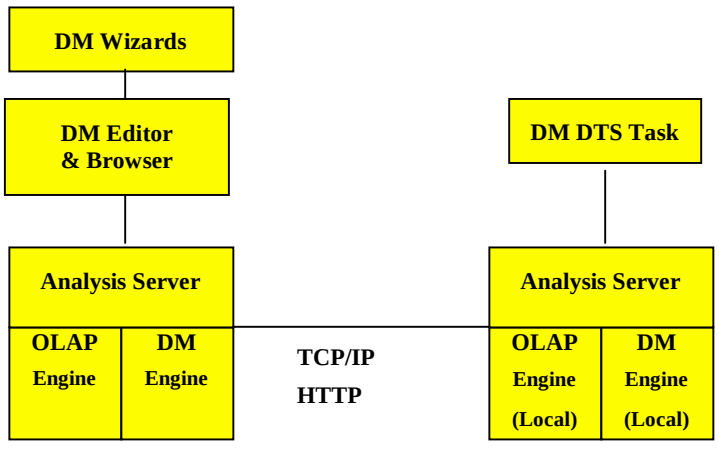


Figure 2 – SQL Server Analysis Services 2000

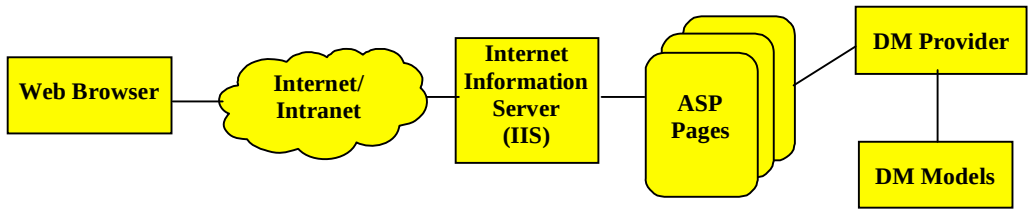


Figure 3 – Building data mining applications

is preserved to the extent possible based upon the size of the main memory buffer and the fit of the current clustering model to the data.

When there are too many attributes involved in training (e.g., more than 255 attributes by default), we apply a feature selection method to filter out less interesting attributes. The interestingness of an attribute is calculated based on the entropy of the attribute [3].

In SQL Server 2005, several more data mining algorithms are added, including time series, association, naïve bayes, sequence clustering, and neural network. It also provides an algorithm plug-in API, which allows 3rd parties to integrate their algorithms into SQL Server.

4. Building Data Mining Application using OLE DB for Data Mining

One of the biggest advantages of Microsoft Data Mining is that it is based on OLE DB for DM specification. It is very easy to use; any database developer can develop applications using data mining features. The data mining language is very similar to SQL. Microsoft data mining provider is open as it is a OLE DB component. Algorithms from other ISVs can be plugged into the same platform. Data mining services can be invoked from any consumer application through a DSO (Decision Support Object) or ADO object.

For example, a bank is developing its loan application software. It would like to add data mining to evaluate the loan risk for each applicant. The application is an n-tier intranet application. A loan assistant inputs customer information through a web form and then by clicking the submit button, the associated loan risk indication will be displayed through an Active Server Page. The architecture of the application is displayed in Figure 3:

Before building the application, the first thing to do is to create a data mining model and train the model. There are a few different ways to do this task. The easiest way is to use the mining model creation wizard of Analysis Manager. The wizard will generate data mining creation and training queries and send these queries to the Microsoft data mining provider through the OLE DB for DM interface. The other way is to write some VB or C++ code to connect to the data mining provider through the Active Data Objects (ADO) or Decision Support Objects (DSO) APIs, and then issue the text queries to the provider like a database developer does with database queries. The following example shows how to connect to the data mining provider through an ADO object in the ASP page.

```
Set con = server.CreateObject("adodb.connection")
con.open "provider=msolap; Data
Source=myserver;initial catalog=Data Mining
Database"
```

The msolap provider will then initialize the Microsoft data mining provider.

To do prediction, it should first create a prediction query in the ASP page. In our example, the loan assistant only inputs information about one loan candidate at a time, so the prediction join is on singleton cases. The query is shown as the following:

```
QueryText = "Select t.CustomerId,
CreditRisk.RiskLevel " _
    + "From CreditRisk natural prediction join (" _
    + "Select 100 as CustomerId, 'Engineer' as -
Profession, 50000 as Income, 30 as Age) as t"
```

To execute the prediction query, it is the same as doing a database query through ADO:

```
Set rs = con.Execute (QueryText)
```

The prediction will return a record set, which has two columns: CustomerId and predicted credit risk level.

Analysis Services 2000 has extended its DSO model to support data mining. DSO is the only way to create and manage models that will persist on the server. Models created through an ADO command object will exist only for the duration of a session – and are called "session mining models." Additionally, using DSO objects allows the creation and setting of security roles and integration with the repository. However, DSO requires more coding than ADO, plus access is restricted to server administrators.

Sometimes consumer applications need to discover the capabilities of a server, the existing models on the server, or even the learned content of a mining model. Model content represents the patterns the data mining algorithm found in the dataset. The mining content for a tree model represents the tree graph. Developers can get this information through schema rowsets that describe the state of the server. The following code shows how to

connect to the content schema rowset. The schema rowsets are defined in the OLE DB for DM specification.

```
const DMSCHEMA_MINING_MODEL_CONTENT =
"{3add8a76-d8b9-11d2-8d2a-00e029154fde}"
set rs = con.OpenSchema(adSchemaProviderSpecific, ,
DMSCHEMA_MINING_MODEL_CONTENT)
```

5. XML for Analysis

The programming methods described above provide an easy way to access data mining functionality from inside applications. However, these methods require certain components to exist on the client machine in order to function, namely the OLEDB client for ADO programming, or the DSO libraries for DSO programming. In circumstances where it is not convenient or possible to install the client components, developers can use XML for Analysis to communicate with their mining models.

XML for Analysis (XML/A) is a SOAP-based XML API standardizing the interaction between clients and analytical data providers. It specifies how to construct SOAP packets that can be sent to an XML/A server to discover metadata and to execute queries. The format of the result is a SOAP packet containing a rowset encoded in XML. This allows connection and interaction from any client platform without any specific client components to communicate to the server, which simplifies application deployment and permits cross-platform development.

XML/A specifies only two commands to interact with the server: *Discover* and *Execute*. The former is used to pull metadata describing the capabilities and the state of the server, and the latter is used to execute queries against the server objects.

The Discover command has the following syntax:

Discover (

[in] *RequestType* As EnumString,

[in] *Restrictions* As Restrictions,

[in] *Properties* As Properties,

[out] *Result* As Rowset)

The RequestType parameter indicates the schema that is being requested – the schemas that are supported by the provider can be accessed by first using the DISCOVER_SCHEMA_ROWSETS request type. Restrictions is an array of OLEDB-type restrictions used to limit the data returned from the server – the list of acceptable restrictions is also available from the same DISCOVER_SCHEMA_ROWSET request. Properties is a collection of properties used to control various aspects of the discover method, such as the return type. The list of supported properties can be accessed through the DISCOVER_PROPERTIES request type. Required request types and properties are specified in the XML for Analysis 1.1 specification. Finally, the Result is the tabular result of the call returned in XML format.

The following example specifies a discover call to a XML/A server requesting a list of mining models in a data base. Note that the response is restricted to those models in the "Foodmart 2000" database and the return format is specified as "Tabular."

```
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
">
 <SOAP-ENV:Body>
  <Discover xmlns="urn:schemas-microsoft-com:xml-analysis"
  SOAP-ENV:encodingStyle
="http://schemas.xmlsoap.org/soap/encoding/">
   <RequestType>DMSCHEMA_MINING_MODELS
</RequestType>
   <Restrictions>
    <RestrictionList>
     <CATALOG_NAME>
      FoodMart 2000
     </CATALOG_NAME>
    </RestrictionList>
   </Restrictions>
   <Properties>
    <PropertyList>
     <DataSourceInfo>
      Provider=MSOLAP;Data Source=local;
     </DataSourceInfo>
     <Catalog>
      Foodmart 2000
     </Catalog>
     <Format>
      Tabular
     </Format>
    </PropertyList>
   </Properties>
  </Discover>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The following is a truncated sample response from the discover

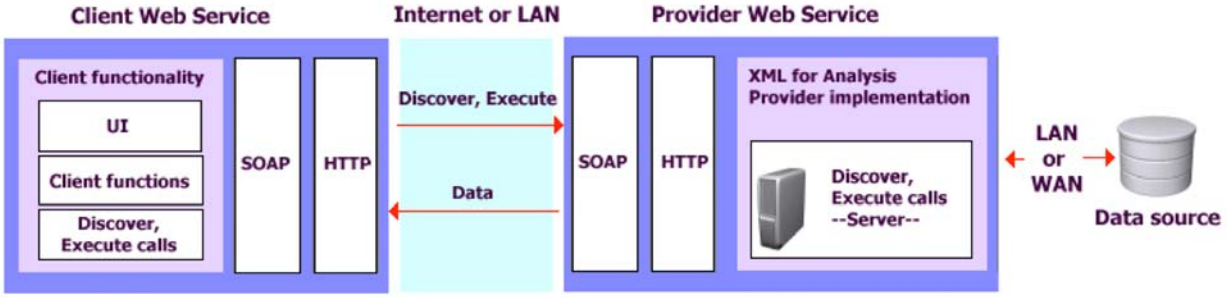


Figure 4: Architecture of XML for Analysis

call.

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV=
"http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle
="http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <DiscoverResponse     xmlns="urn:schemas-microsoft-com:xml-
analysis">
   <return>
    <root>
     <xsd:schema xmlns:xsd=
"http://www.w3.org/2001/XMLSchema">
      <!-- The XML schema definition of the result comes here -->
      ...
     </xsd:schema>
     <row>
      <CATALOG_NAME>FoodMart2000
</CATALOG_NAME>
      <MODEL_NAME>Sales</MODEL_NAME>
      ...
     </row>
     <row>
      <CATALOG_NAME>FoodMart2000
</CATALOG_NAME>
      <MODEL_NAME>Warehouse</MODEL_NAME>
      ...
     </row>
     ...
    </root>
   </return>
  </DiscoverResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The result of an XML/A discover request is a simple XML structure that can be interpreted on any platform using off-the-shelf XML tools.

The Execute method is very similar to the Discover method with this syntax:

Execute (

[in] *Command* As Command,

[in] *Properties* As Properties,

[out] *Result* As Resultset)

The Command parameter specifies the query to be executed, along with any query parameters. The syntax of the query is that described in the OLEDB for Data Mining specification. The Properties parameter is identical to that of the Discover method. And, of course, the Result is the result as specified in the properties parameter.

The following example shows an Execute call of the same query we described previously using ADO.

```
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
"> <SOAP-ENV:Body>
 <Execute xmlns="urn:schemas-microsoft-com:xml-analysis"
  SOAP-ENV:encodingStyle
="http://schemas.xmlsoap.org/soap/encoding/">
  <Command>
   <Statement>
Select t.CustomerId, CreditRisk.RiskLevel From CreditRisk
natural prediction join (Select 100 as CustomerId, 'Engineer' as
Profession, 50000 as Income, 30 as Age) as t
   </Statement>
  <Command>
  <Properties>
   <PropertyList>    <DataSourceInfo>Provider=MSOLAP;Data
Source=local;
    </DataSourceInfo>
    <Catalog>Foodmart 2000</Catalog>
    <Format>Tabular</Format>
   </PropertyList>
  </Properties>
 </Execute>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The resultant rowset will be formatted as in the discover call. With XML/A, the effort to develop data mining web services, especially data mining prediction, is minimal.

XML/A becomes the native communication protocol in SQL Server Analysis Services 2005.

6. Conclusion

With Analysis Services of SQL Server 2000, OLE DB for Data Mining, and XML for Analysis, ordinary database developers can create and train data mining models and embed these advanced features into consumer applications. By adding an easy and flexible programming model, data mining can now become a mass market analytical technique.

7. References

[1] Paul Bradley, Usama Fayyad, Cory Reina, Scaling EM (Expectation Maximization) Clustering to Large Databases, Microsoft Tech. Report MSR-TR-98-35, Microsoft, 1998.

[2] Surajit Chaudhuri, Usama Fayyad, Jeff Bernhardt, Scalable Classification over SQL Databases. ICDE 1999, pp. 470-479.

[3] Huan Liu, Hiroshi Motoda (ed.), Feature Extraction, Construction and Selection: A Data Mining Perspective, Kluwer Academic Publishers, 1998.

[4] Microsoft, OLE DB for Data Mining Specifications, July 2000, www.microsoft. com/data/oledb/dm.

[5] XML/A.org, XML for Analysis Specification v1.1, November 2002, http://xmla.org/docs_pub.asp

Tools for Composite Web Services: A Short Overview**

Richard Hull
Bell Labs Research
Lucent Technologies
hull@lucent.com

Jianwen Su*
Department of Computer Science
UC Santa Barbara
su@cs.ucsb.edu

ABSTRACT

Web services technologies enable flexible and dynamic interoperation of autonomous software and information systems. A central challenge is the development of modeling techniques and tools for eanbling the (semi-)automatic composition and analysis of these services, taking into account their semantic and behavioral properties. This paper presents an overview of the fundamental assumptions and concepts underlying current work on service composition, and provides a sampling of key results in the area. It also provides a brief tour of several composition models including semantic web services, the "Roman" model, and the Mealy/conversation model.

1. INTRODUCTION

The web services paradigm promises to enable rich, flexible, and dynamic interoperation of highly distributed and heterogeneous web-hosted services. Substantial progress has already been made towards this goal (e.g., emerging standards such as SOAP, WSDL, BPEL) and industrial technology (e.g., IBM's WebSphere Toolkit, Sun's Open Net Environment and JiniTM Network technology, Microsoft's .Net and Novell's One Net initiatives, HP's e-speak, BEA's WebLogic Integration). Several research efforts are already underway that build on or take advantage of the paradigm, including the DAML-S/OWL-S program [Coa03, MSZ01, Grü03], and automata-based models for web services [BFHS03, HBCS03, BCG+03]. But there is still a long way to go, especially given the ostensible long-term goal of enabling the *automated discovery*, *composition*, *enactment*, and *monitoring* of collections of web services working to achieve a specified objective. A fundamental challenge concerning the design and analysis of composite web services is to develop necessary techniques and tools to handle the novel aspects of the web services paradigm.

The challenge raises a variety of questions, several of which are relevant for the database research community. Some of the questions are: What is the right way to model web services and their compositions? What is the right way to query them in order to support automated composition and analysis algorithms? And how can the data management aspects of composite web services be incorporated into current web services standards? This paper attempts to provide the groundwork needed to address these questions, by describing emerging frameworks for studying composite services, and identifying emerging tools and techniques for both automated design and analysis of composite web services.

The focus of this short survey is on teh foundations of composition and related issues. From this perspective, it is worthwhile to understand the overall process of designing composite services. Fig. 1 shows the key elements in the typical design process. It is not surprising that the design process is similar to planning studied in articificial intelligence (e.g., [RN95]). In this case, activities that may contribute to accomplish the overall goal need be discovered and orchestrated. The orchestration has to be constrained by the flow of available information, which is generally agreed to be in the form of *messages*. Note that monitoring is closely related to orchestration; specific models of the latter may determine how and to some degree what to monitor. Interestingly, the design process embodies elements from all four task groups listed above.

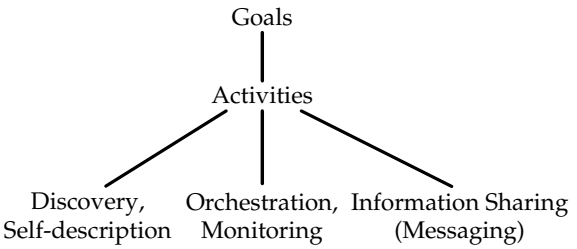


Figure 1: Anatomy of Web Service Composition

Fig. 2 illustrates three dimensions that "measure" Web service description/composition models. The *component service complexity* dimension indicates the information capacity of the languages and model in representing a service. In this dimension, OWL-S is low since it describes only the input and output. On the other hand, WSDL captures richer structural information with XML Schema and uses messages for input and outputs, and other models including BPEL, CSP [Hoa78] and π-Calculus [Mil99], the Mealy model [BFHS03], the Roman model [BCG+03], and BPML also model service states and message or activity sequences. Clearly, the ability of assembling individual services together in this "cluster" of models make them high along the *glue language complexity* dimension. The third dimension is the ability to describe "semantics". OWL-S can describe the properties on the input and output of an operation, and also specify how the service interacts with an

*Supported in part by NSF grants IIS-0101134.
**Database Principles Column.
Column editor: Leonid Libkin, Department of Computer Science, University of Toronto, Toronto, Ontario M5S 3H5, Canada. E-mail: libkin@cs.toronto.edu.

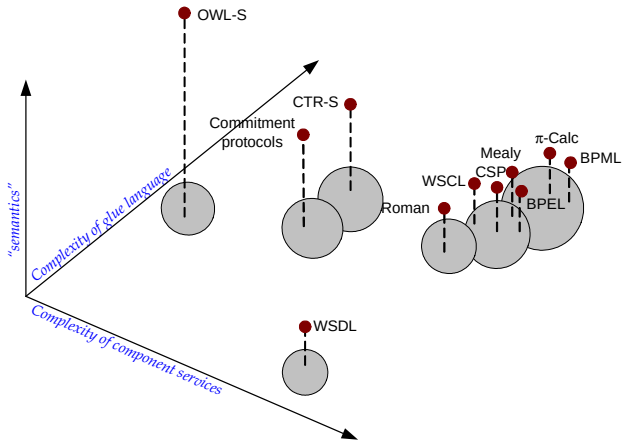


Figure 2: Anatomy of Web Service Composition

abstract model of the "real world", which distinguishes it from the other models.

This paper is organized as follows. Section 2 gives a short overview of several standards related to Web services and composition. Section 3 is intended to provide some high level of key aspects web service composition and a discussion on models that have been studied in the context. Section 4 briefly summarizes techniques and approaches for analyzing Web services. Section 5 concludes the paper.

2. WEB SERVICES AND STANDARDS

A good starting point for understanding the web services paradigm is to consider the stated goals, as found in the literature and the standards communities. The basic motivation of standards such as SOAP and WSDL is to allow a high degree of flexibility in combining web services to create more complex ones, often in a dynamic fashion. The current dream behind UDDI is to enable both manual and automated discovery of web services, and to facilitate the construction of composite web services. Building on these, the BPEL standard provides the basis for manually specifying composite web services using a procedural language that coordinates the activities of other web services.

Much more ambitious goals are espoused by the OWL-S coalition [Coa03] and more broadly the semantic web services community (e.g., [Com05]). These goals are to provide machine-readable descriptions of web services, which will enable automated discovery, negotiation with, composition, enactment, and monitoring of web services. OWL-S is an ontology language for describing web services, in terms of their inputs, outputs, preconditions and (possibly conditional) effects, and of their process model. Importantly, OWL-S provides a formal mechanism for modeling the notion of the state of the "real world", and describing how atomic web services impact that state over time. (This is described in more detail in Subsection 3.1 below.) OWL-S also provides a *grounding*, which provides mechanisms for mapping an OWL-S specification into a WSDL specification (e.g., see [PKPS02]).

A kind of middle ground is also emerging, which provides abstract "signatures" of web services that are richer than WSDL but retain a declarative flavor. Most popular here is the use of automata-based descriptions of permitted sequencing patterns of the web services, with a focus on either activities performed [BCG+03] or messages passed [HBCS03].

The underlying structure for the web services paradigm will most likely be guided by already established standards and practices. Some of the current standards are illustrated by the layered structure shown in Figure 3. Briefly, web services interact by passing XML data, with types specified using XML Schema. SOAP can be used as the communication protocol, and the i/o signatures for web services are given by WSDL. All of these can be defined before binding web services to each other. Behavioral descriptions of web services can be defined using higher level standards such as BPEL, WSCDL, BPML, DAML-S, etc.

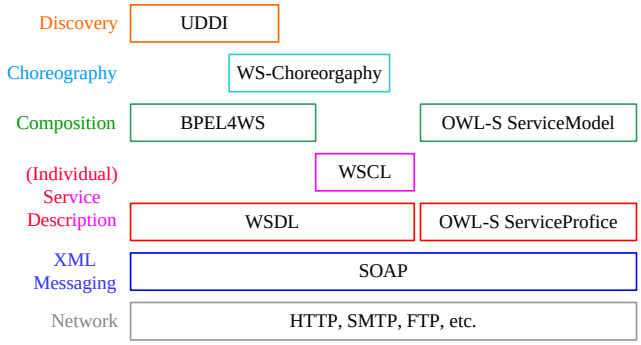


Figure 3: Web Service Standards Stack

In Figure 3, XML messaging and Network layers provide the foundation for interoperations or interactions between services. The starting point for service descriptions is the W3Cs Web Service Description Language (WSDL) [WSDL]. WSDL describes a Web service via its set of visible operations. They can be thought of as message endpoints, or the set of messages that it can send and receive. WSDL specifies on the one hand "reactive" operations in which a message is received by the service. If the reactive operation is declared as "one-way", then it does not return a response, otherwise it is a "request-response" operation, and the return type is also declared. It also describes proactive operations that send out messages from the service. "Notification" operations send out messages without waiting for a response, while "solicit-response" operations block waiting for a response, with the response type being specified with the operation. The receive and response types of the operations are mapped onto concrete XML Schema types to be used in messages. WSDL can thus be seen as an extension of traditional input/output signatures in programming languages and distributed computing to a peer-to-peer setting. A service is viewed both as a server (via its reactive operations) a client (via its proactive operations).

WSDL model services that are essentially stateless. (WSDL2.0 incorporates a limited notion of state, since it enables specification of certain message patterns that a service should satisfy.) While it is clear that the complete *internal* state of a running service is not expected to be known, it is also commonly accepted that *some* execution states need be visible either due to the fact that they are *observable* or the necessity (e.g., to interact correctly with other services). The Web Service Conversation Language (WSCL) [WSCL02] proposal is an interesting approach in defining the overall input and output message sequences for one service using essentially a finite state automaton (FSA) over

the alphabet of message types. The FSA is called a *conversation*.

Note that WSCL complements extremely well with WSDL. A WSDL description along with a WSCL conversation of a service provide a rather rich semantics about the service while keeping the internal implementation/execution encapsulated. For this reason, they are combined into a layer named "individual service description" in the stack in Fig. 3.

While WSDL and WSCL can define Web services, clearly there is a need for languages in the corresponding level of abstraction to compose services. BPEL [BPEL] was developed as a language both for programming Web services and for specification of Web services (e.g., in legacy systems). For example, [HBCS03] identified two topologies for service composition: "peer-to-peer", and "hub-and-spoke" with a mediator. Mediators play the role of coordinating the activities of other web services. A primary goal of BPEL is provide a language for specifying the behavior of mediator services, rather than for general-purpose programming.

In contrast to specifying individual services that BPEL provides, WS-Choreography (WSCDL) [WSCDL04] attempts to specify globally how different component services should behave. WSCDL emphasizes on the "choreography" aspects: the roles of the participating services, information that being passed between services, and channels that enable the information flow.

At the top of the standards stock is the UDDI (Verion 3 was recently adopted as an OASIS standard). UDDI allows services to be registered with a *registry*. Services in a registry can be searched with querying abilities provided by the standard.

3. FOUNDATIONS FOR SERVICE COMPOSITION: A SAMPLING

Web service *composition* addresses the situation when a client request cannot be satisfied by a single pre-existing service, but can be satisfied by suitably combining some available, pre-existing services. Reference [Haa04, ACKM04] identifies two aspects of compostion: *composition synthesis* is concerned with synthesizing a specification of how to coordinate the component services to fulfill the client request; and *orchestration*, is concerned with how to actually achieve the coordination among services, by executing the specification produced by the composition synthesis and by suitably supervising and monitoring that execution. The focus here is on composition synthesis and related issues.

The theoretical study of (automatic) composition synthesis for web services is still in its infancy. The models underlying this work on automatic composition have roots in AI, logic, situation calculii, transition systems, and automata theory, and thus rely on a variety of philosophical bases. The discussion here is not intended to be comprehensive, but rather highlights some of the most important foundations and results that might be most interesting to the database community.

The initial results on automatic composition considered here are grouped around three different models for web services, which we identify here as (a) *OWL-S* (e.g., [Coa03]), which includes a rich model of atomic services and how they interact with an abstraction of the "real world", (b) *Roman* (e.g., [BCG$^+$03]), which uses a very abstract notion of atomic service in a finite-state automata framework for describing process flows, and (c) (*message-based*) *Mealy machine* (e.g., [BFHS03]), which focuses on (message-based) "behavioral signatures" of services, and again using a finite-state automata framework for process flow.

The work on the different models has centered around three different aspects of composition. To describe these, we first establish some vocabulary. When a family of web services interact, the overall topology may be, speaking informally, *mediator-based* or *peer-to-peer*. By mediator-based we mean that there is one web service, called the *mediator*, which has the specialized role of controlling the operation and interaction of the other services; the other services are called *component* services. In a peer-to-peer framework, each participating service is called a *component* service.

It is also useful to distinguish between compositions that are intended for *single* or *multiple* use. In the former case, a composition algorithm is invoked each time that a client identifies a desired goal service, similar to typical scenarios on AI planning. In the latter case, the goal established for the composition algorithm is to produce a (composite) service that can be run multiple times, in support of the same or different clients.

The three families of initial results are now described; more details are given below.

Synthesis of mediator from atomic component services. The first category of results, which has been used successfully [NM02] with OWL-S, is focused on building a workflow schema (e.g., flowchart, Petri net, composite OWL-S service, ...) that invokes atomic services in order to achieve a specified goal within specified constraints. This workflow schema would typically be enforced by a mediator, and in practice might be specified using BPEL. This work tends to focus on the single-use case.

Selection of multi-step components and synthesis of mediator. In this work, initiated with the Roman model, the component services are generally non-atomic, and have process flow specified using a transition system or finite-state automata. Unlike the OWL-S work, however, the atomic services inside the component services are very abstract. As described below, seminal results here focus on the construction (if one exists) of a specialized kind of multi-use mediator that achieves a desired goal behavior.

Synthesis of component services in peer-to-peer setting. The results here focus on message-passing Mealy machines. In this context, a *composition schema* is defined as a template that component web services can be plugged into. The goal behavior is specified as a family of permitted message exchange sequences, or *conversations*, that should be realized by the system. Key results here characterize when a conversation language can be realized, and synthesize component services in that case. As detailed below, these results can be used to help with practical composition, and they provide an approach to map "global" choreography constraints onto "local" constraints concerning the message sequencing behaviors of the component services.

In Subsection 3.1 below we describe the OWL-S model and results in more detail. Special emphasis is placed on the model, and a natural bridge from the AI foundations that OWL-S relies on to a formal basis that essentially builds on a relational database framework. In Subsection 3.2 we

describe the Roman model and results, and in Subsection 3.3 we describe the model and results on message passing and Mealy machines. Finally, Subsection 3.4 presents a brief overview of recent work on web service models and results that combine the fundamental aspects of the above models.

3.1 OWL-S: Model and composition

We now consider salient aspects of the OWL-S model and some of the key composition results obtained. In our presentation we do not follow exactly the original formulation [Coa03], but instead adopt the approach recently introduced by work on the First-order Logic Ontology for Web Services (FLOWS) [BGHM04]. FLOWS provides the basis for the first-order logic component of the recently released Semantic Web Services Ontology (SWSO) [Com05]. We use the FLOWS framework here, as it is somewhat closer to a relational database formulation than the underpinnings originally used by OWL-S.

A key contribution of OWL-S is the explicit modeling of how web services in OWL-S interact with the "real world". The basic building block of the OWL-S process model is the notion of "process"; this includes both atomic and composite processes. OWL-S processes are specified to have inputs, outputs, pre-conditions, and conditional effects (IOPEs). The IOPEs of two example OWL-S atomic processes taken from a stock broker domain are informally sketched now.

```
select_stock
 input: stock_name, quantity
 output: price, reservation_id
 pre-condition: the quantity of stock is
                available for sale
 conditional effects:
   if true, then modify world state to
   reflect the fact that this stock
   is being held for sale

purchase_stock
 input: reservation_id, billable_account
 output: confirmation_id
 pre-condition: reservation_id is valid
                and has not expired
 conditional effects:
   if enough money in billable_account then
      transfer of $$ to stock owner and
      transfer of stock to buyer
   if not enough money, then
   make reservation_id void
```

A client would use `select_stock` in order to check the availability of a given quantity of some stock, and if it is available, then reserve that for purchase. Speaking intuitively, successful execution of the service would result a commitment that this stock will be held for purchase (e.g., for 10 minutes). After a successful reservation of that sort, the client might invoke `purchase_stock` to actually make the purchase, using some bank account to pay for it. `purchase_stock` will execute as long as the reservation is valid, but the outcome of this execution depends on whether there are sufficient funds in the account. A composite OWL-S service might be created by combining these two atomic processes using the sequence construct; in this case the IOPE of this composite process can be inferred from the IOPEs of the atomic processes.

To provide a formal basis for this interaction with the real world, OWL-S takes the approach of the Situation Calculus [Rei01], a logic-based formalism which explicitly models the fact that over time different "situations" or world states will arise. To briefly illustrate this here we follow FLOWS, and use the Process Specification Language (PSL) [sg, GM03], a recent ISO standard which among other things incorporates core aspects of the Situation Calculi. PSL is a first-order logic ontology for describing the core elements of processes. PSL is layered, with PSL-Core at the base and many extensions. PSL-Outercore is a natural family of extensions above PSL-Core, useful for modeling processes. FLOWS is a family of extensions on top of PSL-OuterCore.

PSL-OuterCore provides first-order predicates that can be used to describe the basic building blocks of processes and there execution. To give the flavor, we mention a few highlights. Fundamental is the class of *activity*; an activity can be viewed as a process template, and there may be zero or more *occurrences* of an activity (corresponding to instances of a process or enactments of a workflow). The binary predicate `occurrence_of`(occ, a) is used to specify that occ is an occurrence of activity a. There are *complex* activities, and the binary `subactivity` predicate is used to specify sub-activities of an activity. There are also *atomic* activities, such that, intuitively, the execution of an occurrence of an atomic activity is "atomic" in the typical database sense. A notion of time points in a discrete linear ordering is included; occurrences have a begin time and an end time.

In a typical usage of PSL (FLOWS), an application domain is created by combining the PSL-OuterCore (FLOWS-Core) axioms with domain-specific predicates and sentences to form a (first order logic) theory. The models of this theory can be thought of as a tree or forest, whose nodes correspond to occurrences of (atomic) activities, and with edges from one occurrence to another if they follow each other with no intervening occurrence. A path in this tree (forest) can be viewed as one possible sequence of steps in the execution of the overall system.

Speaking loosely, each sentence in an application domain theory can be viewed as a *constraint* or restriction on the models (in the sense of mathematical logic) that satisfy the theory. In particular, and following the spirit of Golog [Rei01] and similar languages, even process model constructs such as `while` or `if-then-else` correspond formally to constraints rather than procedural elements. A mapping of OWL-S to PSL is provided in [Grü03].

We now get back to our stock purchase example, and provide a database-oriented cast on PSL and OWL-S. The PSL- and domain-specific predicates associated with our example can be viewed essentially as relations in some (potentially vast) relational database. Some of the domain-specific relations might be "accessible" by just one service, others accessible by several services, and still others might be accessible by all services. In the example, we might have the following domain-specific relations

```
available_stock(stock_name, qty, price)
reserved_stock(stock_name, qty, price, res_id)
purchase_reservations(res_id, start_time)
billable_accounts(acct_id, current_balance)
```

Using the above relations it is now relatively straightfoward to specify the behaviors of the atomic processes (which can be viewed as atomic activities in the parlance of PSL). For example, if `select_stock` is called with inputs s, q, then the pre-condition can be specified as a test against the current contents of relation `available_stock`, to see if

there is a tuple (s, q', p) with $q' \geq q$. The effect (which in this case will always occur if the pre-condition holds) might be to replace that tuple with $(s, q' - q, p)$, create a "new" reservation_id r, and insert (s, q, p, r, t) into `reserved_stock`, where t is a time-stamp. All of these operations should be considered as an atomic database transaction. (A refinement would be to enable the case where the shares of stock fulfilling a single request would have multiple prices.)

Now, there is one step remaining before we have the full picture on how the behavior of the OWL-S atomic processes can be formally specified in PSL. Two general approaches have arisen in the context of the situation calculii (see [Pin94] for a detailed discussion). Under one approach, relations such as `available_stock` are transformed by adding a new field, which holds a time value. Under a second approach, taken by PSL, this relation would be transformed into a function, say, `f_available_stock` that takes three arguments. Two PSL predicates, `holds` and `prior` are used to associate (intuitively) truth values to terms created using this function. For example, if the predicate `holds(f_available_stock(s,q,p), occ)` is true in a model, this corresponds to the intuition that the term `holds(f_available_stock(s, q, p)` is true "about the world" immediately after the occurrence `occ` was executed, or in terms of the relational perspective, that tuple (`s`, `q`, `p`) was in the relation `available_stock` at that time. The predicate `prior` works analogously, referring to the time immediately prior to the start of an occurrence.

OWL-S provides a variety of features beyond its rich notion of IOPEs. We mention one here, which is a family of constructs for building composite processes from atomic ones. These constructs are inspired by GOLOG and are reminiscent of flowchart-style constructs, but in OWL-S they are viewed as declarative constraints in the specification of a web service, rather than procedural imperatives.

We now briefly describe two of the key results concerning automatic composition for OWL-S and OWL-S-like services. The first is developed in [NM02], where salient aspects of the OWL-S model are mapped first into a Situation Calculus, and from there into the Petri net domain. In this composition result, it is assumed that all relevant aspects of the application domain are represented using a finite number of propositional fluents, and the family of permitted input and output argument values is finite. (In a relational database setting, this can be achieved by restricting attention to a finite domain.) The goal of the (single-use) composition is then specified in terms of an overall effect to be achieved (on all of the relevant propositional fluents) starting from some initial state (which in essence includes relevant input argument values). Another element of the result is the translation of OWL-S atomic processes into Petri net fragemnts; these are used to assemble a Petri net that corresponds to all possible sequencing of the processes. (For simplicity, it is assumed that no atomic service is invoked twice in these sequences.) The overall complexity of determining the existence of a composition is EXPSPACE. Of course, various heuristics can be applied to make this tractable. Also, the technique can be generalized to support composition of composite processes, as well as atomic onces.

Another approach to composition, also developed in the context of OWL-S, is developed in [MS02]. This work proposes a two-tier framework, based on *generic programs* and *customization via constraints*. Beginning with a family of

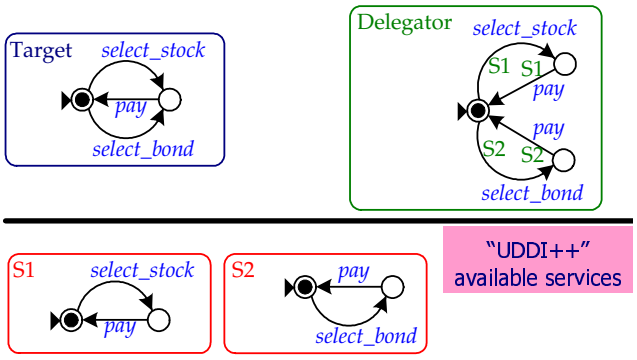


Figure 4: Illustration of composition with Roman model

atomic OWL-S services, a generic program can be specified using the GOLOG language. Note that this generic program may not be completely specified. As a second phase, additional constraints can be written to capture customizations needed by a given client. (It is typical to express these using Horn clauses). If the overall family of constraints is satisfiable, then there is a composition, which can be equated with a branch of the execution tree of the generic program that satisfies certain properties. Reference [MS02] describes how a ConGolog interpreter can be used to find such branches.

3.2 Roman Model perspective: Actions with Automata-based Process Model

We now consider a family of results involving automated composition of multi-step services. This work was launched by the seminal work [BCG+03, BCDG+04a], which is of interest for at least two reasons: (a) the paper develops a novel, general framework for studying composition of human-machine style web services, and (b) the theoretical techniques developed in the paper can be generalized to a much broader context.

The paper starts with a very abstract model of web services, based on an abstract notion of *activities*. Basically, there is a (finite) alphabet of activity names, but no internal structure is modeled (no input, output, or interaction with the world). To specify the internal process flow of a web service, [BCG+03, BCDG+04a] starts with transition systems. In the most general case, these are potentially infinite trees, where each branch corresponds to a permitted sequencing of executions of the activities. For the theoretical results they restrict attention to finitely specifiable transition systems; in particular on systems that can be specified as deterministic finite-state automata, where the activities act as the input alphabet (i.e., where the edges are labeled by activities). It is convenient to refer to this as the "Roman" model.

References [BCG+03, BCDG+04a] focus on human-machine style web services. As a simple illustration of how services work, consider the service labeled `Target` in Figure 4. When this service is being used by a client (could be human or automated) it performs the following sequence of steps:

(1) Give the client a set S of activities to choose from (possibly including the special "activity" `terminate`).

(2) Wait for the client to choose exactly one element of S

(3) Execute the chosen activity and return to step (1), or terminate if that was chosen.

In each iteration, the set S is chosen according to the current state of the service – in a state p the set S includes each activity that labels an out-going edge from p, and includes "terminate" if p is a final state.

To illustrate, when the `Target` is launched it gives the client a choice of {`select_stock`, `select_bond`, `terminate`}. Supposing the client chooses `select_stock`, in the next iteration the service gives the set {`pay`}. Execution continues until the client chooses `terminate`.

We can now describe the basic composition problem studied with the Roman model. Consider now the two services `S1` and `S2` shown in Figure 4. Suppose that these are the only pre-existing services available. The question is whether it is possible to use some or all of those to "build" a service that acts the way `Target` does. One way to accomplish this is with the "service" shown as `Delegator` in Figure 4. The operation of this is similar to the operation of the other Roman services, except that with each transition of `Delegator` it is also assumed that one or more of the pre-existing services performs a transition. In particular, the pre-existing service must perform the same activity as the `Delegator`. Intuitively, one can think of `Delegator` as not executing the services, but rather, as delegating the execution to the pre-existing services. Furthermore, in a valid execution of `Delegator`, each of the pre-existing services must perform a valid (and terminating) execution.

The central result of [BCG$^+$03] is that the existence of a delegator can be determined in EXPTIME, and that there is a constructive approach for building such delegators. Importantly, the form or skeleton of the delegator may be different than the form or skeleton of the target service (e.g., in the example the delegator has more states than the target service).

This result can be proved by using a transformation of the problem into Propositional Dynamic Logic (PDL), a well-known logic for reasoning about programs [KT90]. As detailed in [BCDG$^+$04a], there is a natural, polynomial-size translation of a Roman composition problem into deterministic PDL. The result then follows because satisfiability for deterministic PDL is EXPTIME, and a constructive technique is available. Further, [BCDG$^+$04b] describes how a description logic reasoner has been adapted to perform this test and construction.

We briefly mention two extensions of the results in [BCG$^+$03]. Reference [BGL$^+$04] extends the basic model by considering (pre-existing) services that can themselves delegate to other services. In the model there, if an edge in a service $S1$ has label $^{>>}a$ this indicates that $S1$ can perform activity a, and if an edge in $S2$ has label $a^{>>}$ this indicates that $S2$ is able to delegate activity a to some other service. The delegation feature adds a level of non-determinism, but checking the existence of, and constructing, a (generalized) delegator is still possible in EXPTIME.

Consider now the composition problem illustrated in Figure 5. There is no delegator that can simulate the target service in this case. However, as discussed in [GHS04], there is a "1 look-ahead" delegator, that is, a delegator which can make the correct delegations, as long as the delegator is given information not only about the immediate choice of the client, but also the choice that the client will make in the next move. Reference [GHS04] shows that there is a non-collapsing hierarchy of k look-ahead delegation problems, and there are examples requiring "infinite" or arbitrarily

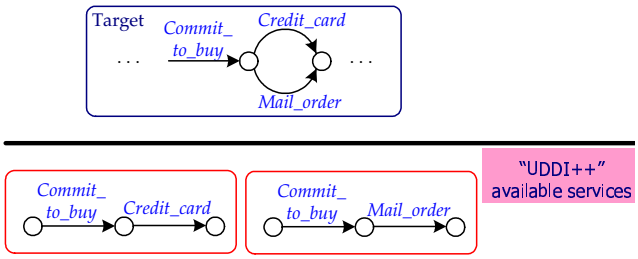


Figure 5: A composition problem with a 1 look-ahead solution

long look-ahead. Also, the problem of checking existence of a k look-ahead delegator can be transformed into a problem of checking existence of a conventional (0 look-ahead) delegator, and has EXPTIME complexity.

3.3 Conversations and Mealy services

The OWL-S and Roman models focus on *what* a service or composition does, either in terms of the input/output of services (activities) and their impact on the world, or the sequencing of the activities in a service. At the operational level, neither addresses the issue of *how* the component services in composition should interact with each other. The notion of "conversations" was naturally formulated in addressing this need [HNK02, HNL02, WSCL02]. Preliminary theoretical work on conversations was reported in [BFHS03, FBS03]. In this subsection, we highlight key technical notions and results. Much of this work assumes a peer-to-peer framework.

To begin with, we assume the existence of an infinite set of *service names*. These will act essentially as place-holders for *service spec(ification)s*, which can be viewed as completely or partially specified implementations for the service names. We also assume an infinite set of *message classes*, where each class mc has a service name as *source* and a service name as *target*. A *composition schema* is a a pair (P, M) where P is a finite set of service names and M is a finite set of message classes. whose sources and targets are in P. Figure 6 shows a composition schema for a "stock analysis service" (SAS) [FBS04b]. The SAS composition schema uses three service names: *Investor*, *Stock Broker*, and *Research Department*, and uses the 9 message classes indicated in the figure. This composition schema models abstractly a domain in which the *Investor* service can request reports on different stocks from the *Research Department*. The requests are controlled by the *Stock Broker* service, which handles issues such as permissions and billing.

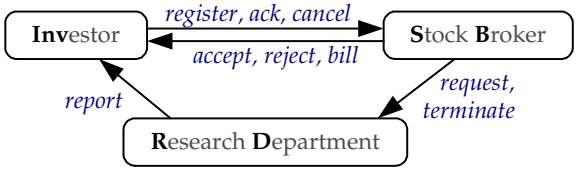


Figure 6: A Composition Schema

A *composite service* for composition schema (P, M) is an association of actual web services to each service name in P, where the actual web services are capable of sending/receiving messages coming from the relevant message classes in M. Given a composite service, a basic approach for the underlying operational model is that each service has a FIFO queue for all incoming messages (possibly from dif-

ferent services). This correcponds abstractly to the message oriented model in SOA [Haa04, ACKM04]. Queuing of messages provides a model asynchronous communciation, which is realistic in the web setting.

Given a composite service S over composition schema (P, M), a *conversation* of S is the (global) sequence of messages sent during one successful execution of S. Several standards and research works are concerned with properties of conversation languages. WS-Choreography [WSCDL04], and references [HNK02, HNL02, WSCDL04] focus on "global" constraints on conversations. In contrast, the standards WSCL [WSCL02] and WSDL2.0 [WSDL04] focus on "local" constraints on conversations, in the sense that they focus on constraints about message patterns of the component services of a composite service.

Reference [BFHS03] initiated the formal study of the relationship between such global and local constraints. A *conversation protocol* for a composition schema (P, M) is a finite state machine over the set of message classes in M. Figure 7 shows a conversation protocol for the SAS service. In this example, the *register* message from INV can include a list of stock symbols. Upon acceptence of a message of this type, each symbol will result in a seperate *request* from the *Stock Broker* service to the *Research Department* service (corresponding to the *request-report-ack* loop in the protocol).

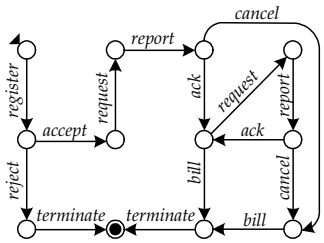


Figure 7: A Conversation Protocol for SAS

Three natural questions are (Q1) Given a conversation protocol over composition schema (P, M), can it be realized by some composite service S over (P, M)? (Q2) If so, what can be inferred about the individual services in S? (Q3) If the individual services in a composition have a message-passing behavior corresponding to a regular language, then is the resulting conversation also characterized by a regular language (i.e., by a conversation protocol)?

In [BFHS03] these questions are studied using the notion of *Mealy services*. These are service specs based on finite state automata whose transitions correspond to sending a message m (denoted as $!m$) or receiving a message m (denoted as $?m$). Fig. 8 shows a Mealy service for INV. A Mealy composite service is a composite service over Mealy services.

We note that a Mealy service can be viewed in three different ways: (a) as an implementation of a service name; (b) as a "behavioral signature" that abstractly describes some properties of a service behavior; and (c) as a constraint on the local messaging behavior that a service should satisfy.

With regards to question (Q3), in [BFHS03], it was shown that the conversation languages of some Mealy composite services are not regular (see Fig. 9), nor even context-free in the language hierarchy, but they are always context-sentitive. The key factor here is the use of *unbounded* queues. This is not surprising, since finite state automata

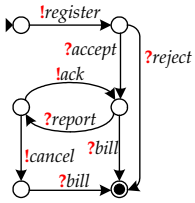


Figure 8: A Mealy Service for Investor

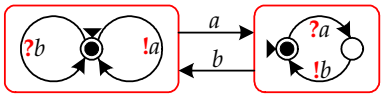


Figure 9: A Non-regular Conversation Language

with queues can simulate Turing machines [BZ83]. When all queues are bounded, the conversation languages are always regular.

Turning to question (Q1), a finer analysis reveals two important aspects of conversations languages: (a) What individual Mealy service sees vs. the (global) conversation, and (b) When two Mealy services are both ready to send, it isn't possible to force a particular ordering. These two factors are formulated into two operators in [BFHS03]:

- *Projecting* a conversation to the message alphabet of a single service name and *joining* message sequences of Mealy services into a conversation, and
- *Preponing* (i.e., swapping) messages from different senders (in a local perspective).

It turns out that Mealy conversation languages are closed under (1) projection-join and (2) preponing.

The closure results indicate that if a conversation protocol is not closed under either operator, then it cannot be realized by any composite service. Intituively, this suggests that during design it may not be a good candidate as a global constraint on the composition during design. On the other hand, it is interesting to know if the projection-join and prepone closure of a regular language is *realizable* as the conversation language of some Mealy composite service. The answer turns out to be positive [BFHS03], and, relevant to question (Q2), a construction is provided.

It is not clear whether one can always determine if a regular language is closed under projection-join and preponing, though a sufficient condition was given in [FBS03].

We conclude this discussion by describing two broad ways in which the above results can be applied in practical settings. One is to help in the process of practical compositions. Suppose that one seeks to create a (peer-to-peer) composition, based on a pre-determined composition schema, and satisfying a given conversation protocol. Suppose further that a family of component Mealy services is synthesized to satisfy this protocol. This family of Mealy services can be used to select a family of candidate (real-world) web services that satisfy the "local" constraints specified by the Mealy services. Then perform further analysis of the candidate services, on aspects that are not captured by the Mealy services alone.

The second practical application is to view a conversation language as a global constraint that should be satisfied by a family of web services, essentially in the spirit of the WS-Choreography standard. The results above can be used to determine whether this global constraint can be realized at

all. It can also be used to infer local constraints, that specify properties of the behavioral signatures of each component service.

3.4 Selected Additional Developments

We close our sampling of composition models and results with a brief overview of two very recent efforts, which hold the promise of providing a unifying framework for this area.

The first, already touched upon in Subsection 3.1, is the work on FLOWS [BGHM04, Com05], which provides a first-order logic ontology for web services. We provide here a little more detail on FLOWS.

FLOWS includes objects of type *service*. These can be related to several types of object, including non-functional properties (e.g., name, author, URL), and also to its *activity*. This is a PSL activity, and may have specialized kinds of subactivities. This includes (*FLOWS*) *atomic processes*, which are based on OWL-S atomic processes with inputs, outputs, pre-conditions and conditional effects. Some of these atomic processes, as in OWL-S, are *domain-specific*, in that they interact with the "real world" as represented in the application domain, and others are *service-specific*, in that they focus on the standardized mechanics of services, e.g., message-passing and channel management (see below).

The flow of information *between services* can occur in essentially two ways: (a) via message passing and (b) via shared access to the same "real world" fluent (e.g., an inventory database, a reservations database). With regards to message passing, FLOWS models *messages* as (conceptual) objects that are created, read, and (possibly) destroyed by web services. A message life-span has a non-zero duration. Messages have types, which indicate the kind of information that they can transmit. FLOWS also includes a flexible *channel* construct.

To represent the acquisition and dissemination of information *inside a Web service*, FLOWS follows the spirit of the ontology for "knowledge-producing actions" developed in [SL03]; this also formed the basis for the situation calculus semantics of OWL-S inputs and effects [NM02].

FLOWS is very open-ended concerning the process or data flow between the atomic processes inside a services. This is intentional, as there are several models for the internal processing in the standards and literature (e.g., BPEL, OWL-S Process Model, Roman model, Mealy model) and many other alternatives besides (e.g., rooted in Petri nets, in process algebras, in workflow models, in telecommunications). The FLOWS work is still quite preliminary, but holds the promise of providing a unifying foundation for the study and comparison of these many variations on the notion of web service.

A second very recent work is reported in [BCD+05b, BCD+05a]. That work develops Colombo, a formal model for web services that combines

(a) A world state, representing the "real world", viewed as a database instance over a relational database schema

(b) Atomic processes in the spirit of OWL-S,

(c) Message passing, including a simple notion of ports and links, as found in web services standards (e.g., WSDL, BPEL)

(d) An automata-based model of the internal behavior of web services, where the individual transitions correspond to atomic processes, message writes, and message reads.

The first three elements parallel in several aspects the core elements of FLOWS. Colombo also includes

(e) a "local store" for each web service, used manage the data read/written with messages and input/output by atomic processes; and

(f) a simple form of integrity constraints on the world state

Using the Colombo model, [BCD+05b] develops a framework for posing composition problems, that closely parallels the way composition will have to be done using standards-based web services. Specifically, the basic composition problem studied is how to build a mediator that simulates the behavior of a target web service, where the mediator can only use message passing to get the pre-existing web services to perform atomic activities (which in turn impact the "real world"). Under certain restrictions, [BCD+05b] demonstrates the decidability of the existence of a mediator and develops a method for constructing them. This result is based on (a) a technique for reducing potentially infinite domains of data values into a finite set of symbolic data values, and (b) in a generalization of [BCDG+04a], a mapping of the composition problem into PDL. The results reported in [BCD+05b] rely on a number of restrictions; a broad open problem concerns how these restrictions can be relaxed while still retaining decidability.

4. ANALYSIS COMPOSITE SERVICES

The need for analysis is particularly acute for composite services, especially if they are to be created from pre-existing services using automatic algorithms. The ultimate goal is to ensure that the eventual execution of a composite service produces the desired behavior. Ideally, one would be able to statically verify properties (e.g., in temporal logic) for composite services. There have been various attempts at developing such static analysis methods for web services and workflow systems.

Based on workflows represented in concurrent transaction logic, [DKRR98] studied the problem of whether a workflow specification satisfies some given constraints similar to the Event Algebra of [Sin95]. Algorithms were given for such analysis. An alternative approach is developed in [vdA99], which maps conventional workflows to Petri nets, and then applies standard results to analyze properties such as termination and reachability for workflows. Similar results have been obtained for OWL-S compositions [NM02].

There are two recent projects that use model checking techniques to verify BPEL composite web services. In [FUMK03], mediated composite services specified in BPEL were verified against the design specified using Message Sequence Chart and Finite State Process notations, with a focus on the control flow logic. In [FBS04a], both conversation protocols and Mealy services were extended to *guarded automata* which incorpoate (i) XML messages and (ii) XPath expressions as the basis for verifying temporal properties of the conversations of composite web services. The following shows an example of a transition in a protocol for the SAS service:

```
t14 {
```

```
  s8 -> s12 : bill,
  Guard {
    $request//stockID =
    $reginfo//stockID [position() = last()]
    =>
    $bill[//orderID := $reginfo//orderID]
  }
},
```

Roughly, the transition is defined from state "s8" to "s12" on message "bill". The part of guard prior to "=>" defines an additional equality condition with both side XPath expressions. The part after "=>" specified an assignment.

The extended model makes it possible to verify designs at a more detailed level and to check properties about message content. A framework is presented where BPEL specifications of web services are translated to an intermediate representation, followed by the translation of the intermediate representation to the verification language Promela, input language of the model checker SPIN [Hol03]. Since the SPIN model checker is a finite-state verification tool, the tool can only achieve partial verification by fixing the sizes of the input queues in the translation. Sufficient conditions for complete verification are also obtained.

An important step in statically analyzing Web services defined in BPEL (or other languages) is to translate them into formalisms that are suitable for analysis. Such formalisms include finite state machines [Nak02, FUMK03], extended Mealy machines [FBS04a], process algebra [KvB03]. Effectively, these translations provide formal semantics to BPEL. Most of the translations mentioned above focus only on control flow structures, with an exception of [FBS04a] (where XML message types, local data, and XPath expressions are also taken into consideration. More recently, an extensive translation of BPEL into a version of Petri nets was developed [Mar04], which deals with scoping rules, variable assignments, correlation sets, among other things.

The presence of databases makes the static analysis extremely hard but not entirely impossible. In [DSV04], a rule based language was developed for specifying interactive Web services. The rules may access associated (relational) databases. Web service properties are defined using a linear temporal first-order logic. Based on the Abstract State Machine techniques [Spi03], it was shown that some properties are verifiable for a subclass of Web services.

Service oriented architecture may "fundamentally change the way software is made and used" [Blo02]. The current development platforms may no longer be suitable. While this is a rare chance to revisit important software development environment, there is no doubt that analysis tools (static or dynamic), testing and debugging tools, monitoring tools are becoming critically importance.

5. CONCLUSIONS

The web services paradigm raises a vast array of questions, including many that will be of interest to, and can benefit from, the database research community. First is the question of finding appropriate models and abstractions for representing Web services and their "behaviors", which are suitable to the web services paradigm, and can support efficient querying and manipulation as needed by web service composition and analysis algorithms. More broadly, to what extent can the problem of automated composition be re-cast to be a problem in writing and answering one or several queries against behavioral descriptions of services? Another aspect concerns the application of XML constraint-checking techniques to perform compile-time or run-time checking of Web service specifications (e.g., in WSDL and BPEL, or in emerging behavioral specification languages).

A second category of questions is how to bring data manipulation more clearly into the web services paradigm and their associated standards. The standards and most research at present are focused primarily on process model and I/O signatures, but not on the data flow and the manipulation of the data as it passes through this flow. Is there value in associating integrity constraints with web service I/O signatures? What is an appropriate way to model the data transformations occurring in a web service, which will enable reasoning about the behavior of data being passed or written to databases by a composite web service? Are there specialized models of Web service composition that will be more suitable for applications that are targeted primarily at data processing? A starting point here might be [AVFY98, DSV04].

Acknowledgements: The authors wish to thank their collaborators for many fruitful discussions and inspirations, including Michael Benedikt, Daniela Berardi, Tevfik Bultan, Diego Calvanese, Vassilis Christophides, Guiseppe De Giacomo, Xiang Fu, Cagdas Gerede, Michael Gruninger, Oscar Ibarra, Grigorios Karvounarakis, Maurizio Lenzerini, Massimo Marcella, and Sheila McIlraith.

6. REFERENCES

G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services. Concepts, Architectures and Applications.* Springer-Verlag, 2004.

S. Abiteboul, V. Vianu, B. Fordham, and Y. Yesha. Relational transducers for electronic commerce. In *Proc. ACM Symp. on Principles of Database Systems*, 1998.

D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic composition of web services in Colombo. In *Proc. of 13th Itallian Symp. on Advanced Database Systems*, June 2005.

D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic Composition of Transition-based Semantic Web Services with Messaging. Technical report, University of Rome, "La Sapienza", Roma, Italy, April, 2005.

D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Mecella. Automatic Services Composition based on Behavioral Descriptions. *International Journal of Cooperative Information Systems (IJCIS)*, 2004. To appear.

D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Mecella. $\mathcal{ESC}$: A Tool for Automatic Composition of e-Services based on Logics of Programs. In *Proc. Workshop on Technologies for E-Services*, 2004.

D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Mecella. Automatic composition of e-services that export their behavior. In *Proc. 1st Int. Conf. on Service Oriented Computing (ICSOC)*, volume 2910 of *LNCS*, pages 43–58, 2003.

T. Bultan, X. Fu, R. Hull, and J. Su. Conversation specification: A new approach to design and analysis of e-service composition. In *Proc. Int. World Wide Web Conf. (WWW)*, May 2003.

D. Berardi, M. Gruninger, R. Hull, and S. McIlraith. Towards a first-order ontology for web services. In *W3C Workshop on Constraints and Capabilities for Web Services*, October 2004.

D. Berardi, G. De Giacomo, M. Lenzerini, M. Mecella, and D. Calvanese. Synthesis of underspecified composite e-services based on automated reasoning. In *Proc. Second International*

Conference on Service-Oriented Computing, pages 105–114, 2004.

J. Bloomberg. The seven principles of service-oriented development. *XML & Web Services*, 3(5):32–33, 2002.

Business Process Execution Language for Web Services (BPEL), Version 1.1. `http://www.ibm.com/developerworks/library/ws-bpel`, May 2003.

D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, 1983.

OWL Services Coalition. OWL-S: Semantic markup for web services, November 2003.

SWSL Committee. Semantic web service ontology (swso), 2005. Available in http://www.daml.org/services/swsl/report/.

H. Davulcu, M. Kifer, C. R. Ramakrishnan, and I. V. Ramakrishnan. Logic based modeling and analysis of workflows. In *Proc. ACM Symp. on Principles of Database Systems*, pages 25–33, 1998.

A. Deutsch, L. Sui, and V. Vianu. Specification and verification of data-driven web services. In *Proc. ACM Symp. on Principles of Database Systems*, 2004.

X. Fu, T. Bultan, and J. Su. Conversation protocols: A formalism for specification and verification of reactive electronic services. In *Proc. Int. Conf. on Implementation and Application of Automata (CIAA)*, 2003.

X. Fu, T. Bultan, and J. Su. Analysis of interacting BPEL web services. In *Proc. Int. World Wide Web Conf. (WWW)*, May 2004.

X. Fu, T. Bultan, and J. Su. Model checking XML manipulating software. In *Proc. Int. Symposium on Software Testing and Analysis (ISSTA)*, July 2004.

H. Foster, S. Uchitel, J. Magee, and J. Kramer. Model-based verification of web service compositions. In *Proc. the 18th IEEE Int. Conf. on Automated Software Engineering Conference (ASE 2003)*, 2003.

C. E. Gerede, R. Hull, and J. Su. Automated composition of e-services with lookahead. In *Proc. Intl. Conf. on Services-Oriented Computing*, 2004.

M. Grüninger and C. Menzel. Process specification language: Principles and applications. *AI Magazine*, 24:63–74, 2003.

M. Grüninger. Applications of PSL to semantic web services. In *Proceedings of SWDB'03, The first International Workshop on Semantic Web and Databases*, 2003.

H. Haas. Architecture and future of web services: from SOAP to semantic web services. `http://www.w3.org/2004/Talks/0520-hh-ws/`, May 2004.

R. Hull, M. Benedikt, V. Christophides, and J. Su. E-services: A look behind the curtain. In *Proc. ACM Symp. on Principles of Database Systems*, 2003.

J. E. Hanson, P. Nandi, and S. Kumaran. Conversation support for business process integration. In *Proc. 6th Int. Enterprise Distributed Object Computing (EDOC)*, Ecole Polytechnic, Switzerland, 2002.

J. E. Hanson, P. Nandi, and D. W. Levine. Conversation-enabled web services for agents and e-business. In *Proc. International Conference on Internet Computing (IC)*, pages 791–796. CSREA Press, 2002.

C. A. R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666–677, 1978.

G. J. Holzmann. *The SPIN Model Checker: Primer and Reference Manual*. Addison-Wesley, Boston, Massachusetts, 2003.

Dexter Kozen and Jerzy Tiuryn. Logics of programs. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science — Formal Models and Semantics*, pages 789–840. Elsiever, 1990.

M. Koshkina and F. van Breugel. Verication of business processes for web services. Technical Report CS-2003-11, Department of Computer Science, York University, 2003.

A. Martens. Analysis and re-engineering of web services. In *Proc. 6th Int. Conf. on Enterprise Information Systems*, 2004.

R. Milner. *Communicating and Mobile Systems: The π-calculus*. Cambridge University Press, 1999.

S. McIlraith and T. Son. Adapting Golog for composition of semantic web services. In *Proc. of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002)*, pages 482–493, April 2002.

S. A. McIlraith, T. C. Son, and H. Zeng. Semantic web services. In *IEEE Intelligent Systems*, March/April 2001.

S. Nakajiama. Verification of web services with model checking techniques. In *Proc. First Int. Symposium on Cyber Worlds*. IEEE Computer Society Press, 2002.

S. Narayanan and S. McIlraith. Simulation, verification and automated composition of web services. In *Proc. Int. World Wide Web Conf. (WWW)*, 2002.

J.A. Pinto. *Temporal Reasoning in the Situation Calculus*. PhD thesis, University of Toronto, 1994.

M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara. Importing the semantic web in UDDI. In *Proc. Workshop on Web Services, E-business and Semantic Web (at CAISE conference)*, 2002.

R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, Cambridge, MA, 2001.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.

PSL standards group. Psl home page. `http://ats.nist.gov/psl/`.

M. Singh. Semantical considerations on workflows: An algebra for intertask dependencies. In *Proc. Workshop on Database Programming Languages (DBPL)*, 1995.

R. B. Scherl and H. J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144:1–39, 2003.

M. Spielmann. Verification of relational transducers for electronic commerce. *Journal of Computer and System Sciences*, 66(1):40–65, 2003.

W. M. P. van der Aalst. On the automatic generation of workflow processes based on product structures. *Computer in Industry*, 39(2):97–111, 1999.

Web Services Choreography Description Language Version 1.0 (W3C Working Draft). `http://www.w3.org/TR/2004/WD-ws-cdl-10-20041217/`, December 2004.

Web Services Conversation Language (WSCL) 1.0. `http://www.w3.org/TR/2002/NOTE-wscl10-20020314/`, March 2002.

Web Services Description Language (WSDL) 1.1. `http://www.w3.org/TR/wsdl`, March 2001.

Web Services Description Language (WSDL) Version 2.0 Part 2: Predefined Extensions (W3C Working Draft). `http://www.w3.org/TR/2004/WD-wsdl20-extensions-20040803/`, August 2004.