SIGMOD Officers, Committees, and Awardees

Chair	Vice-Chair	Secretary/Treasurer
Yannis Ioannidis	Christian S. Jensen	Alexandros Labrinidis
University of Athens	Department of Computer Science	Department of Computer Science
Department of Informatics	Aalborg University	University of Pittsburgh
Panepistimioupolis, Informatics Bldg	Selma Lagerlöfs Vej 300	Pittsburgh, PA 15260-9161
157 84 Ilissia, Athens	DK-9220 Aalborg Øst	USA
HELLAS	DENMARK	
+30 210 727 5224	+45 99 40 89 00	+1 412 624 8843
<yannis AT di.uoa.gr>	<csj AT cs.aau.dk >	<labrinid AT cs.pitt.edu>

SIGMOD Executive Committee:
Curtis Dyreson, Christian S. Jensen, Yannis Ioannidis, Alexandros Labrinidis, Jan Paredaens, Lisa Singh, Raghu Ramakrishnan, and Jeffrey Xu Yu.

Advisory Board: Raghu Ramakrishnan (Chair), Yahoo! Research, <First8CharsOfLastName AT yahoo-inc.com>, Rakesh Agrawal, Phil Bernstein, Peter Buneman, David DeWitt, Hector Garcia-Molina, Masaru Kitsuregawa, Jiawei Han, Alberto Laender, Tamer Özsu, Krithi Ramamritham, Hans-Jörg Schek, Rick Snodgrass, and Gerhard Weikum.

Information Director:
Jeffrey Xu Yu, The Chinese University of Hong Kong, <yu AT se.cuhk.edu.hk>

Associate Information Directors:
Marcelo Arenas, Denilson Barbosa, Ugur Cetintemel, Manfred Jeusfeld, Alexandros Labrinidis, Dongwon Lee, Michael Ley, Rachel Pottinger, Altigran Soares da Silva, and Jun Yang.

SIGMOD Record Editor:
Alexandros Labrinidis, University of Pittsburgh, <labrinid AT cs.pitt.edu>

SIGMOD Record Associate Editors:
Magdalena Balazinska, Denilson Barbosa, Ugur Çetintemel, Brian Cooper, Cesar Galindo-Legaria, Leonid Libkin, and Marianne Winslett.

SIGMOD DiSC Editor:
Curtis Dyreson, Washington State University, <cdyreson AT eecs.wsu.edu>

SIGMOD Anthology Editor:
Curtis Dyreson, Washington State University, <cdyreson AT eecs.wsu.edu>

SIGMOD Conference Coordinators:
Lisa Singh, Georgetown University, <singh AT cs.georgetown.edu>

PODS Executive: Jan Paredaens (Chair), University of Antwerp, <jan.paredaens AT ua.ac.be>, Georg Gottlob, Phokion G. Kolaitis, Maurizio Lenzerini, Leonid Libkin, and Jianwen Su.

Sister Society Liaisons:
Raghu Ramakhrishnan (SIGKDD), Yannis Ioannidis (EDBT Endowment).

Awards Committee: David Maier (Chair), Portland State University, <maier AT cs.pdx.edu>, Rakesh Agrawal, Peter Buneman, Laura Haas, and Gerhard Weikum.

[Last updated on October 3, 2009]

SIGMOD Officers, Committees, and Awardees (continued)

SIGMOD Edgar F. Codd Innovations Award

For innovative and highly significant contributions of enduring value to the development, understanding, or use of database systems and databases. Until 2003, this award was known as the "SIGMOD Innovations Award." In 2004, SIGMOD, with the unanimous approval of ACM Council, decided to rename the award to honor Dr. E.F. (Ted) Codd (1923 - 2003) who invented the relational data model and was responsible for the significant development of the database field as a scientific discipline. Recipients of the award are the following:

Michael Stonebraker (1992)	Jim Gray (1993)	Philip Bernstein (1994)
David DeWitt (1995)	C. Mohan (1996)	David Maier (1997)
Serge Abiteboul (1998)	Hector Garcia-Molina (1999)	Rakesh Agrawal (2000)
Rudolf Bayer (2001)	Patricia Selinger (2002)	Don Chamberlin (2003)
Ronald Fagin (2004)	Michael Carey (2005)	Jeffrey D. Ullman (2006)
Jennifer Widom (2007)	Moshe Y. Vardi (2008)	Masaru Kitsuregawa (2009)

SIGMOD Contributions Award

For significant contributions to the field of database systems through research funding, education, and professional services. Recipients of the award are the following:

Maria Zemankova (1992)	Gio Wiederhold (1995)	Yahiko Kambayashi (1995)
Jeffrey Ullman (1996)	Avi Silberschatz (1997)	Won Kim (1998)
Raghu Ramakrishnan (1999)	Michael Carey (2000)	Laura Haas (2000)
Daniel Rosenkrantz (2001)	Richard Snodgrass (2002)	Michael Ley (2003)
Surajit Chaudhuri (2004)	Hongjun Lu (2005)	Tamer Özsu (2006)
Hans-Jörg Schek (2007)	Klaus R. Dittrich (2008)	Beng Chin Ooi (2009)

SIGMOD Jim Gray Doctoral Dissertation Award

SIGMOD has established the annual SIGMOD Jim Gray Doctoral Dissertation Award to *recognize excellent research by doctoral candidates in the database field.* This award, which was previously known as the SIGMOD Doctoral Dissertation Award, was renamed in 2008 with the unanimous approval of ACM Council in honor of Dr. Jim Gray. Recipients of the award are the following:

- **2006** *Winner*: Gerome Miklau, University of Washington
  Runners-up: Marcelo Arenas, Univ. of Toronto; Yanlei Diao, Univ. of California at Berkeley.
- **2007** *Winner*: Boon Thau Loo, University of California at Berkeley
  Honorable Mentions: Xifeng Yan, UIUC; Martin Theobald, Saarland University
- **2008** *Winner*: Ariel Fuxman, University of Toronto
  Honorable Mentions: Cong Yu, University of Michigan; Nilesh Dalvi, University of Washington.
- **2009** *Winner*: Daniel Abadi (advisor: Samuel Madden), MIT
  Honorable Mentions: Bee-Chung Chen (advisor: Raghu Ramakrishnan), University of Wisconsin at Madison; Ashwin Machanavajjhala (advisor: Johannes Gehrke), Cornell University.

A complete listing of all SIGMOD Awards is available at: **http://www.sigmod.org/awards/**

[Last updated on October 3, 2009]

Editor's Notes

Welcome to the June 2009 issue of SIGMOD Record. We begin the issue with an article about the 2009 SIGMOD/PODS award winners.

The first regular article of this issue, by Mazon and Trujillo, is proposing a new framework for the multidimensional modeling of data warehouses.

The **Database Principles Column** (edited by Leonid Libkin) features one overview article, on machine models for query processing, by N. Schweikardt, who advocates the introduction of new models to handle modern software and database technologies.

We continue with an article in the **Surveys Column** (edited by Cesar Galindo-Legaria), entitled "XML: Some Papers in a Haystack" by Moro, Braganholo, Dorneles, Duarte, Galante, and Mello. The article categorizes the plethora of XML research topics and summarizes some of the most representative papers in each category.

We continue with an article in the **Systems and Prototypes Column** (edited by Magdalena Balazinska), authored by Chatziantoniou and Tzortzakakis, that presents DataMingler, a GUI for ASSET Queries, which are a declarative alternative to MapReduce.

The **Distinguished Profiles in Data Management Column** (edited by Marianne Winslett) features an interview of Peter Buneman, who is a professor in the School of Informatics at the University of Edinburgh. Peter is an ACM Fellow, a trustee of the VLDB Endowment, and a Fellow of the Royal Society of Edinburgh. Read Peter's interview to find out (among many other things) about the Integration of Databases and Programming Languages, Curated Databases, British Plumbing, the Value of Talking to Users, and When to Ignore the Literature.

We continue with five articles in the **Event Reports Column** (edited by Brian Cooper). First is the *Report on the 10th International Workshop on Web Information and Data Management (WIDM 2008)*, written by Chen and Polyzotis. Second is the *Report on the Workshop on Operating Systems Support for Next Generation Large Scale NVRAM (NVRAMOS 2009)*, written by Lee, Kang, Won and Choi. Third is the *Report on the Workshop on Theory and Practice of Provenance (TaPP 2009)*, written by James Cheney.

We close the issue with multiple **Calls for Papers/Submissions** associated with the 2010 SIGMOD/PODS Conference (SIGMOD Research, PODS, SIGMOD Demos, Industrial Track, Panels, and Tutorials) along with a call for papers for a special issue of the IEEE Privacy and Security magazine.

Alexandros Labrinidis
September 2009

2009 SIGMOD/PODS Award Winners

SIGMOD Edgar F. Codd Innovations Award

For innovative and highly significant contributions of enduring value to the development, understanding, or use of database systems and databases. Until 2003, this award was known as the "SIGMOD Innovations Award." In 2004, SIGMOD, with the unanimous approval of ACM Council, decided to rename the award to honor Dr. E.F. (Ted) Codd (1923 - 2003) who invented the relational data model and was responsible for the significant development of the database field as a scientific discipline.

2009 SIGMOD Edgar F. Codd Innovations Award Winner:

Masaru Kitsuregawa

Masaru Kitsuregawa is the recipient of the 2009 SIGMOD Edgar F. Codd Innovations Award for *contributions to high-performance database technology*. Kitsuregawa made major contributions to the development of hash-join algorithms, which significantly improved the performance of join operations in relational database systems. That work has influenced related research in areas such as query execution, plan optimization and dynamic query-workload balancing, as well as the development of commercial database products. He implemented the hash-based approach on a variety of platforms, including the Functional Disk System and multi-node PC clusters, demonstrating its substantial advantages through detailed evaluations. He has also applied hash-based strategies to parallel association mining and showed its effectiveness there. His contributions in the hardware area include a high-speed sorting system with a sophisticated memory management algorithm. That work was eventually commercialized in collaboration with colleagues, and won the Datamation sort benchmark in 2000.

Details

Professor Kitsuregawa has contributed extensively to the area of high-performance database systems, particularly involving hash-based methods. The work began in the early 1980's in the context of the GRACE relational-database machine. He is particularly known for his work on hash-based join algorithms, which is still widely cited. By the late 1980's and early 1990's, others had built on that work to develop various hybrid versions of hash join, and most database conference of that time had sessions devoted to the topic. His own refinements include dynamic destaging and bucket tuning. At the time, most commercial relational-database products used only looping and sort-based joins. Nearly all current system include hash-based join implementations. He also contributed hash-based approaches to aggregation operations.

He went on to implement the Functional Disk System (FDS), a parallel, hash-based relational system with a shared-memory architecture. He demonstrated that efficient parallel execution of relational operations was possible with hash-based methods, showing substantial performance improvement on the Wisconsin Benchmark. He also developed database-engine software for a shared-nothing architecture on a 100-node PC cluster, which was evaluated against other systems on the TPC-D Benchmark in the late 1990s. He is among the first to apply hash-based approaches to parallel data mining.

SIGMOD Contributions Award

For significant contributions to the field of database systems through research funding, education, and professional services.

2009 SIGMOD Contributions Award Winner:

Beng Chin Ooi

Beng Chin Ooi is the recipient of the 2009 SIGMOD Contributions Award for *his sustained and selfless contributions to the database community in promoting and pursuing high standards of database research at both the international and regional level.* Beng Chin Ooi has been consistently involved in activities that push the frontier of database research. He has taken leadership roles on technical programs for ICDE, SSD, SIGMOD and VLDB. He has helped ensure a high standard for database journal publications as an editor for The VLDB Journal, IEEE Transactions on Knowledge and Data Engineering, International Journal of Geographical Information Science, and Geoinformatica. Currently, he is the editor-in-chief of IEEE Transactions on Knowledge and Data Engineering, and an editor for the Distributed and Parallel Databases journal. He also serves on the SIGMOD Doctoral Dissertation Award committee and on the board of the VLDB Endowment. Ooi has contributed greatly to promoting database research, especially in the Asia-Pacific region, by building an eminent database group at the National University of Singapore that supports high quality research throughout the region through direct collaborations and participation in regional conferences.

Details

Beng Chin Ooi has been continuously involved in leadership roles that help create strong technical programs for the community. He served as the co-PC chair for the Third International Symposium on Large Spatial Databases (SSD) in 1993. He also served as the vice PC Chair for the International Conference on Data Engineering (ICDE) in 2000, 2004 and 2006. More recently, Ooi was involved in running two major database conferences, serving as PC chair for the ACM SIGMOD International Conference on Management of Data in Beijing in 2007 and as the Core DB PC chair in the Very Large Database Conference (VLDB) in Auckland in 2008. Ooi has consistently worked to maintain the high quality of these conferences.

Ooi has also been involved in ensuring high standards of database journal publications. He was an editor for The VLDB Journal, IEEE Transactions on Knowledge and Data Engineering (TKDE), International Journal of Geographical Information Science and Geoinformatca. He has recently been appointed editor-in-chief of TKDE and continues to serve as an editor for the Distributed and Parallel Databases journal.

Ooi is currently a board member of the VLDB Endowment and is involved in the liaison and fostering of cooperation between the Endowment and database researchers in the Asia-Pacific region. Since 2005 he has served as a member the ACM SIGMOD Dissertation Award committee. Since 2008 he has served as co-chair of this committee.

Ooi is an active proponent for database research, especially in the Asia-Pacific region. He has been engaging researchers from other countries in the region in collaborative projects with a view to enhancing the standard of work being done. This outreach has resulted in fruitful collaborations with leading universities in the region, including Fudan University, Tsinghua

University and Renmin University of China. Ooi has served on the program committees of many regional conferences. In 2005, he was co-PC chair of the International Conference on Database Systems for Advanced Applications (DASFAA) and he acted as the general chair for the same conference in 2006. He successfully advocated Singapore as the venue for VLDB 2010. His research group has taken an active part in regional conferences such as DASFAA, COMAD, ADC, APWeb, WAIM and NDBC, both as organizers and participants.

SIGMOD Test of Time Award

The ACM SIGMOD Test of Time Award recognizes the best paper from the SIGMOD proceedings 10 years prior (i.e., for 2009 the 1999 proceedings were consulted), based on the criterion of identifying the paper that has had the most impact (research, products, methodology) over the intervening decade. This paper is chosen by the SIGMOD Awards Committee.

2009 SIGMOD Test of Time Award

Approximate Computation of Multidimensional Aggregates of Sparse Data Using Wavelets
Jeffrey Scott Vitter (Texas A&M University) and **Min Wang (IBM Research)**

This influential paper showed that aggregates over sparse, high-dimensional arrays can be approximated with wavelets to give a compact data-cube representation that supports queries at interactive speeds. Prior histogram-based methods had prohibitive I/O costs for massive data sets of high dimensionality. The method is more accurate than random sampling and supports progressive refinement of answers when additional accuracy is desired. The paper inspired significant follow-on work by others, in the OLAP domain and also more broadly in approximate query processing, selectivity estimation, indexing of images and time series, and data-stream processing.

Abstract of the 1999 SIGMOD paper:

Computing multidimensional aggregates in high dimensions is a performance bottleneck for many OLAP applications. Obtaining the exact answer to an aggregation query can be prohibitively expensive in terms of time and/or storage space in a data warehouse environment. It is advantageous to have fast, approximate answers to OLAP aggregation queries. In this paper, we present a novel method that provides approximate answers to high-dimensional OLAP aggregation queries in massive sparse data sets in a time-efficient and space-efficient manner. We construct a compact data cube, which is an approximate and space-efficient representation of the underlying multidimensional array, based upon a multiresolution wavelet decomposition. In the on-line phase, each aggregation query can generally be answered using the compact data cube in one I/O or a small number of I/Os, depending upon the desired accuracy. We present two I/O-efficient algorithms to construct the compact data cube for the important case of sparse high-dimensional arrays, which often arise in practice. The traditional histogram methods are infeasible for the massive high-dimensional data sets in OLAP applications. Previously developed wavelet techniques are efficient only for dense data. Our on-line query processing algorithm is very fast and capable of refining answers as the user demands more accuracy. Experiments on real data show that our method provides significantly more accurate results for typical OLAP aggregation queries than other efficient approximation techniques such as random sampling.

SIGMOD Best Paper Award

2009 SIGMOD Best Paper Award

Generating Example Data for Dataflow Programs

Christopher Olston, Shubham Chopra, Utkarsh Srivastava (Yahoo! Research)

While developing data-centric programs, users often run (portions of) their programs over real data, to see how they behave and what the output looks like. Doing so makes it easier to formulate, understand and compose programs correctly, compared with examination of program logic alone. For large input data sets, these experimental runs can be time-consuming and inefficient. Unfortunately, sampling the input data does not always work well, because selective operations such as filter and join can lead to empty results over sampled inputs, and unless certain indexes are present there is no way to generate biased samples efficiently. Consequently new methods are needed for generating example input data for data-centric programs.

We focus on an important category of data-centric programs, *dataflow programs*, which are best illustrated by displaying the series of intermediate data tables that occur between each pair of operations. We introduce and study the problem of generating example intermediate data for dataflow programs, in a manner that illustrates the semantics of the operators while keeping the example data small. We identify two major obstacles that impede naive approaches, namely (1) highly selective operators and (2) noninvertible operators, and offer techniques for dealing with these obstacles. Our techniques perform well on real dataflow programs used at Yahoo! for web analytics.

http://doi.acm.org/10.1145/1559845.1559873

2009 SIGMOD Best Paper Award Runner-up

An Architecture for Recycling Intermediates in a Column-store

Milena G. Ivanova, Martin L. Kersten, Niels J. Nes, Romulo A.P. Goncalves (CWI)

Automatically recycling (intermediate) results is a grand challenge for state-of-the-art databases to improve both query response time and throughput. Tuples are loaded and streamed through a tuple-at-a-time processing pipeline avoiding materialization of intermediates as much as possible. This limits the opportunities for reuse of overlapping computations to DBA-defined materialized views and function/result cache tuning. In contrast, the operator-at-a-time execution paradigm produces fully materialized results in each step of the query plan. To avoid resource contention, these intermediates are evicted as soon as possible.

In this paper we study an architecture that harvests the by-products of the operator-at-a-time paradigm in a column store system using a lightweight mechanism, the *recycler*. The key challenge then becomes selection of the policies to admit intermediates to the resource pool, their retention period, and the eviction strategy when facing resource limitations. The proposed recycling architecture has been implemented in an open-source system. An experimental analysis against the TPC-H ad-hoc decision support benchmark and a complex, real-world application (SkyServer) demonstrates its effectiveness in terms of self-organizing behavior and its significant performance gains. The results indicate the potentials of recycling intermediates and charters a route for further development of database kernels.

http://doi.acm.org/10.1145/1559845.1559879

SIGMOD Jim Gray Doctoral Dissertation Award

SIGMOD has established the annual SIGMOD Jim Gray Doctoral Dissertation Award to recognize excellent research by doctoral candidates in the database field. This award, which was previously known as the SIGMOD Doctoral Dissertation Award, was renamed in 2008 with the unanimous approval of ACM Council in honor of Dr. Jim Gray.

2009 SIGMOD Jim Gray Doctoral Dissertation Award - Winner

- Daniel Abadi (advisor: Samuel Madden), MIT
  Dissertation: *Query Execution in Column-Oriented Database Systems*.

This dissertation addresses all the key aspects of implementing an efficient database system based on a column-store architecture including storage management, compression strategies, query execution techniques, and alternative strategies for materializing result rows. The thesis is remarkable for its breadth, depth, system implementation and impact, and is an excellent treatise on the design and implementation of database systems employing a column-oriented storage model. The work is significant in bringing sound database design principles into practice and offering efficient solutions to urgent real problems.

2009 SIGMOD Jim Gray Doctoral Dissertation Award - Honorable Mentions:

- Bee-Chung Chen (advisor: Raghu Ramakrishnan), University of Wisconsin at Madison
  Dissertation: *Cube-space Data Mining*

- Ashwin Machanavajjhala (advisor: Johannes Gehrke), Cornell University.
  Dissertation: *Defining and Enforcing Privacy in Data Sharing*

The annual SIGMOD Jim Gray Dissertation Award recognizes excellent research by doctoral candidates in the database field. For the 2009 award a dissertation needed to be completed and accepted by the candidate's department between September 1, 2007 and December 15, 2008. This year 14 outstanding dissertations were nominated by their departments. The dissertations were evaluated by the SIGMOD Jim Gray Dissertation Awards Committee (Beng Chin Ooi (co-chair), Johannes Gehkre (co-chair---excused due to COI), Alfons Kemper, Hank Korth, Alberto Laender, Gerome Miklau, Timos Sellis, Kyu-Young Whang) for technical depth and significance of the research contribution, potential impact on theory and practice, and quality of the presentation. All submissions were of extremely high quality and we congratulate all the nominated students and their departments for their excellent work and contributions to our field.

2009 SIGMOD Undergraduate Awards

- Daniel Scott Brotherston (University of Waterloo) -- *Comparing and visualizing query optimizer search spaces*
- David M. Lewis (University of Maryland, Baltimore County) -- *An evaluative comparison of similarity coefficients for binary valued data*
- Sang-Phil Lim (Sung Kyun Kwan University) -- *FASTer FTL for enterprise-class flash memory SSDs*
- Manasi Vartak (WPI) -- *Recommendation based query relaxation via mapping functions and space partitioning*
- Shashank Yaduvanshi (IIT Bombay) -- *An architecture for regulatory compliant database management*

A complete listing of all SIGMOD Awards is available at: **http://www.sigmod.org/awards/**

2009 PODS Best Paper Award

Size and Treewidth Bounds for Conjunctive Queries

Georg Gottlob, Stephanie Tien Lee (Oxford University), and Gregory J. Valiant (University of California, Berkeley)

This paper provides new worst-case bounds for the size and treewith of the result *Q(D)* of a conjunctive query *Q* to a database *D*. We derive bounds for the result size |*Q(D)*| in terms of structural properties of *Q*, both in the absence and in the presence of keys and functional dependencies. These bounds are based on a novel "coloring" of the query variables that associates a *coloring number C(Q)* to each query *Q*. Using this coloring number, we derive tight bounds for the size of *Q(D)* in case (i) no functional dependencies or keys are specified, and (ii) simple (one-attribute) keys are given. These results generalize recent size-bounds for join queries obtained by Atserias, Grohe, and Marx (FOCS 2008). An extension of our coloring technique also gives a lower bound for |*Q(D)*| in the general setting of a query with arbitrary functional dependencies. Our new coloring scheme also allows us to precisely characterize (both in the absence of keys and with simple keys) the treewidth-preserving queries--the queries for which the output treewidth is bounded by a function of the input treewidth. Finally we characterize the queries that preserve the sparsity of the input in the general setting with arbitrary functional dependencies.

http://doi.acm.org/10.1145/1559795.1559804

2009 PODS Best Student Paper Award

XPath Evaluation in Linear Time with Polynomial Combined Complexity

Pawel Parys (Warsaw University)

We consider a fragment of XPath 1.0, where attribute and text values may be compared. We show that for any unary query in this fragment, the set of nodes that satisfy the query can be calculated in time linear in the document size and polynomial in the size of the query. The previous algorithm for this fragment also had linear data complexity but exponential complexity in the query size.

http://doi.acm.org/10.1145/1559795.1559805

ACM PODS Alberto O. Mendelzon Test-of-Time Award

In 2007, the PODS Executive Committee decided to establish a Test-of-Time Award, named after the late Alberto O. Mendelzon, in recognition of his scientific legacy, and his service and dedication to the database community. Mendelzon was an international leader in database theory, whose pioneering and fundamental work has inspired and influenced both database theoreticians and practitioners, and continues to be applied in a variety of advanced settings. He served the database community in many ways; in particular, he served as the General Chair of the PODS conference, and was instrumental in bringing together the PODS and SIGMOD conferences. He also was an outstanding educator, who guided the research of numerous doctoral students and postdoctoral fellows. The Award is to be awarded each year to a paper or a small number of papers published in the PODS proceedings ten years prior, that had the most impact (in terms of research, methodology, or transfer to practice) over the intervening decade. The decision was approved by SIGMOD and the ACM. The funds for the Award were contributed by IBM Toronto.

The PODS Executive Chair has appointed us to serve as the Award Committee for 2009. After careful consideration, we have decided to select the following paper as the award winner for 2009:

Georg Gottlob, Nicola Leone and Francesco Scarcello.
Hypertree Decompositions and Tractable Queries

The paper deals with a central problem in database research, namely finding classes of conjunctive queries for which problems, such as the evaluation of Boolean queries and query containment, are in polynomial time. This problem has attracted a lot of attention since the pioneering work of Yanakakis on acyclic queries. The paper shows that the earlier notion of bounded query width (introduced by Chekuri and Rajaraman in ICDT 97) is NP-hard, introduces the notion of bounded hypertree width, then shows that this notion properly generalizes earlier notions of acyclicity, that constant hypertree width is efficiently recognizable, and that Boolean queries with constant hypertree width can be efficiently evaluated. The results of the paper are applicable to both conjunctive query evaluation and to constraint satisfaction. The paper is extensively cited in the literature, and had an impact on subsequent research on these two problems. Hence, the committee has found it to be worthy of the Award.

Catriel Beeri (Chair), Phokion G. Kolaitis, Christos H. Papadimitriou
The Alberto O. Mendelzon Test-of-Time Award Committee for 2009

Georg Gottlob is a Professor of Computing Science at Oxford University, UK, and an Adjunct Professor of Computer Science at the Vienna University of Technology (TU Wien). His research interests are in database theory (in particular, query languages), Web information processing, AI, and computational logic. Gottlob got his Ph.D. degree in Computer Science from TU Vienna, Austria in 1979 and 1981, respectively. Before he moved to Oxford in 2006, he was a Professor of Computer Science at TU (since 1988). Before that, he was affiliated with the Italian National Research Council in Genoa, Italy. He also was a Research Associate and lecturer at the Politecnico di Milano, Stanford University and held visiting positions at Paris VII and at Berkeley. Georg has received the Wittgenstein Award from the Austrian National Science Fund. He is an ACM and an ECCAI Fellow, and a member of the Austrian Academy of Sciences, the German National Academy of Sciences Leopoldina, and the European Academy of Sciences Academia Europaea in London. He chaired the Program Committees of IJCAI 2003 and ACM PODS 2000. He has co-founded the Lixto software company (www.lixto.com) which offers software and services for Web data extraction.

Nicola Leone is full professor of Computer Science, Head of the Department of Mathematics, and Director of the PHD School ISIMR at University of Calabria. He has previously held the professor position for Database Systems at Vienna University of Technology in Austria from 1995 to 2000. Before that, he was with the Italian National Research Council in Cosenza, Italy. His research interests include Database Theory, Artificial Intelligence, and Computational Logics. He is the leader of the team which developed DLV -- the state-of-the-art Disjunctive-Datalog system. He is the author of over 200 scientific papers. He was the program chair of a number of conferences including LPNMR-99, JELIA-02, AGP-03, and LPNMR-05. He was the invited speaker of LPAR'06, RR'07, LPNMR'07, and ICLP'08, and he is the President of the Steering Committee of LPNMR. He has recently co-founded three spin-off companies, namely, Exeura, Artemat, and DLVSYSTEM, working on Databases, AI applications, and Knowledge Management.

Francesco Scarcello is a professor of computer science at the University of Calabria. His research interests include computational complexity, database theory, constraint satisfaction, graph theory, game theory, knowledge representation, and non-monotonic reasoning. He has extensively published in all these areas in leading conferences and journals, including the Journal of the ACM, Journal of Computer and System Sciences, Information and Computation, and Artificial Intelligence. He holds a Ph.D. in Computer Science from the University of Calabria. He was recipient of two grants from the Italian National Research Council (CNR) until 1999, when he became an assistant professor at the University of Calabria. He holds his current position of associate professor since 2001. Also, during these years, he has been visiting the Department of Information Systems at the Vienna University of Technology. He serves on program committees and as a reviewer for many international conferences and journals. He is co-recipient of the 2008 IJCAI-JAIR Best Paper Prize, awarded to an outstanding paper published in the Journal of Artificial Intelligence Research in the preceding five calendar years.

A complete listing of all PODS Awards is available at: **http://www.sigmod.org/pods/**

A Hybrid Model Driven Development Framework for the Multidimensional Modeling of Data Warehouses*

Jose-Norberto Mazón
Lucentia Research Group
Dept. of Software and Computing Systems
University of Alicante, Spain
jnmazon@dlsi.ua.es

Juan Trujillo
Lucentia Research Group
Dept. of Software and Computing Systems
University of Alicante, Spain
jtrujillo@dlsi.ua.es

ABSTRACT

Developing a multidimensional (MD) model of a data warehouse (DW) is a highly complex, prone to fail, and time consuming task, due to the fact that (i) the information needs of decision makers and the available operational data sources that will populate the DW must both be considered in a conceptual MD model, and (ii) complex mappings must be performed to obtain an implementation of this conceptual MD model. However, no significant effort has been made to take these issues into account in a systematic, well structured and comprehensive development process. To overcome the lack of such a process, a framework based on the Model Driven Architecture (MDA) is proposed for the development of a hybrid MD model at the conceptual level and for the automatic derivation of its logical representation. Also, a running example is shown throughout this paper.

1. INTRODUCTION

Data warehouse (DW) systems provide a multidimensional (MD) view of huge amounts of historical data from operational sources, thus supplying useful information for decision makers to improve a business process in an organization. The MD paradigm structures information into facts and dimensions. A fact contains the interesting measures (fact attributes) of a business process (sales, deliveries, etc.), whereas a dimension represents the context for analyzing a fact (product, customer, time, etc.) by means of hierarchically organized dimension attributes. MD modeling requires specialized design techniques that resemble the traditional database design methods [16]. First, a conceptual design phase is performed whose output is an implementation-independent and expressive MD model for the DW. A logical design phase then aims to obtain a technology-dependent model from the previously defined conceptual MD model. This logical model is the basis for the implementation of the DW. Therefore, there are two cornerstones in MD modeling: the development of a conceptual MD model and the derivation of its corresponding logical representation.

On one hand, with regard to conceptual design, current approaches usually embrace one of the following perspectives: (i) *data-driven*, in which the conceptual MD model is based on a detailed analysis of the data sources; while information requirements are only considered later when the implemented MD model is queried, and (ii) *requirement-driven*, in which the conceptual MD model design is based on the information needs of decision makers, thus considering the available data sources later when the DW is populated. Nevertheless, although data-driven approaches simplify the process of properly populating the DW, user needs and expectations may not be satisfied by the MD model designed because these approaches do not provide enough mechanisms through which to analyze and understand the decision making process supported by the DW. Moreover, data-driven approaches do not provide mechanisms to highlight the important parts of the data sources, thus wasting resources by specifying unneeded information structures in the MD model. Conversely, requirement-driven approaches involve decision makers in the development of the MD model, but lack mechanisms through which to formally match the data sources with information requirements in early stages of the development, thus making it highly complex to populate the DW in a proper manner, as the correspondence between the elements of the MD model and their counterparts in the data sources may not be obvious. In order to overcome these drawbacks, we argue that the most promising solution consists of formally considering both the data sources and the requirements in a hybrid approach.

On the other hand, once a conceptual MD model is designed, another important issue must be faced: obtaining a logical representation of the conceptual MD model. This is a tedious and repetitive task which requires a high degree of automatization if failures in the implementation of the DW are to be avoided.

Bearing these considerations in mind, two tasks must be considered in the development of the DW if the previously-stated problems are to be overcome: (i) the formal reconciliation of the operational data sources and the information needs of decision makers at the conceptual level, and (ii) the formal specification of transformations which will allow designers to obtain the most suitable logical representation of

*Work supported by the ESPIA (TIN2007-67078) project from the Spanish Ministry of Education and Science, and and by the QUASIMODO project (PAC08-0157-0668) from the Castilla-La Mancha Ministry of Education and Science. Jose-Norberto Mazón is funded by the Spanish Ministry of Education and Science under a FPU grant (AP2005-1360).

the developed conceptual MD model in an automatic manner. Several previous works have separately covered these tasks [7, 8, 9, 10, 11, 12]. In this paper, the main contribution is to propose a Model Driven Development (MDD) process to elegantly combine these previous works into an integrated framework. MDD is useful to this aim because it offers mechanisms to both manage the integration of models and to define the transformations between them. Other works have, in fact, taken advantage of using MDD in database development (such as [20]). However, to the best of our knowledge, our work is the first to tackle MD modeling problems and overcome them by using MDD.

The remainder of this paper is structured as follows. Sect. 2 presents current approaches towards the MD modeling of DWs, stressing how our work contributes to the state-of-the-art. Sect. 3 describes our overall MDD approach towards MD modeling. A running example is used to clarify theoretical concepts. Our conclusions are presented in Sect. 4.

2. RELATED WORK

Research in DW modeling has been tackled from several points of view. For example, the design of ETL processes [19] or the development of formal models for data analysis [18]. The focus of this paper is on research that addresses MD modeling of the DW repository.

Current approaches towards conceptual MD modeling focus on providing mechanisms with which to specify an implementation-independent MD model. These approaches can be classified into data-driven [4] or requirement-driven approaches [15]. Data-driven approaches are based on analyzing the operational data sources, while requirements are considered later when the data is about to be analyzed. Requirement-driven approaches focus on the information needs of decision makers, and data sources are only taken into account later when the data is loaded into the DW.

Surprisingly, few approaches advocate the consideration of both data sources and information requirements in the early stages of development [3]. Importantly, these approaches rely on a manual analysis of data sources to reconcile them with the requirements, which could be unfeasible in real-life situations, where the operational data sources are huge. Therefore, an automatization of this reconciliation process is faced up to as a key issue in MD modeling [12].

Moreover, approaches for conceptual MD modeling either do not give mechanisms with which to derive the logical representation [1] or they only describe a set of informal guidelines through which to derive a logical representation with a low level of automatization [4]. Other works [7] have pointed out the need of having a set of transformation rules for deriving logical schemas. However, until now, mechanisms with which to define formal transformations to automatically derive every possible logical representation from the conceptual MD model are not provided, and obtaining logical models thus becomes a time-consuming, tedious and prone to fail task for designers.

To overcome these problems, a hybrid approach for MD modeling is described in this paper. This approach (i) combines both data-driven and requirement-driven strategies in an integrated fashion (i.e. reconciling data sources and decision makers' information needs at the conceptual level) in such a way that the DW meets decision makers' needs and simultaneously agrees with data sources, and (ii) provides a repository of formal transformations through which

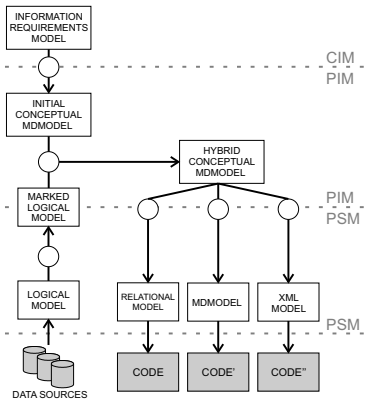


Figure 1: Hybrid MDA-based framework for DWs

to include the knowledge about how to automatically obtain a suitable logical representation from a conceptual MD model, thus ameliorating the tedious and prone to fail task of designers who can save time and effort in obtaining the implementation of the DW.

3. HYBRID MULTIDIMENSIONAL MODEL DRIVEN DEVELOPMENT

Several parts of our MD modeling approach have been separately defined in our previous work [7, 8, 9, 10, 11, 12]. Now, in this paper, thanks to the integration mechanisms provided by MDD, each piece is fitted into an overall framework (see Fig. 1). The novelty of this framework is that it considers a hybrid viewpoint for MD modeling in a systematic, well structured and comprehensive way, whilst a set of formal transformations are simultaneously established in order to obtain the final implementation of the MD model in an automatic manner. Concretely, our framework is based on the Model Driven Architecture (MDA) [13] proposed by the OMG as a standard with which to carry out the MDD. As is shown in Fig. 1, a conceptual MD model of the DW (Platform Independent Model, PIM) is developed from an information requirements model (Computation Independent Model, CIM) obtained from decision makers [9]. This initial PIM must then be reconciled with the data sources [12, 10], thus obtaining a hybrid PIM. Each element of these data sources must be previously marked with its MD counterparts. Moreover, several logical models can be derived from this hybrid PIM as Platform Specific Models (PSMs), by taking into account different deployment platforms (relational, multidimensional, etc.). The relations between models are implemented by using the Query/View/Transformations (QVT) language [13]. These relations are shown in Fig. 1 as circles. Finally, several transformations have been defined in order to obtain the code for the implementation of the MD model according to each PSM. These transformations have been established by means of the Model to Text Transformation (Mof2Text) language [13].

A plugin that supports our approach has been developed with the *Eclipse Modeling Framework*[1]. Our running example has been implemented by using this plugin.

[1] `http://www.eclipse.org/modeling/emf/`

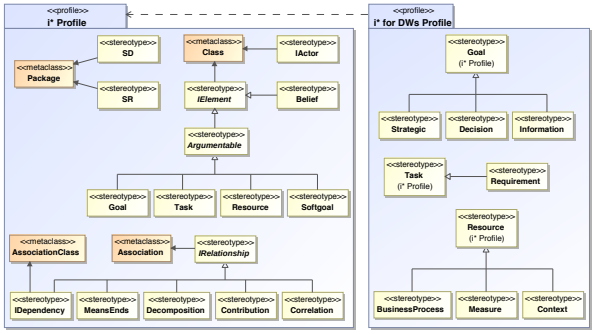


Figure 2: UML profiles for i* modeling in DWs

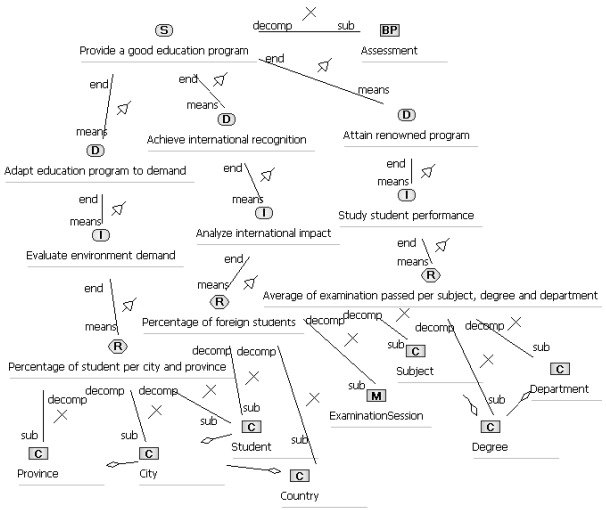


Figure 3: CIM for our running example

3.1 Multidimensional CIM

An explicit requirement analysis stage is needed in order to model the information requirements of decision makers and to derive a suitable conceptual MD model which meets the real needs of decision makers, thus increasing the success of a DW project. Decision makers who use DWs often ignore how to suitably describe information requirements, since they are concerned rather with the goals which the DW helps to fulfil. Therefore, a requirement analysis phase for DWs must ideally start by discovering decision makers' goals. The information requirements can be discovered more easily from these goals.

Goals related to the DW can be specified on three levels [9]: *Strategic goals*, which are main objectives of the business process: e.g. *"increase sales"*. *Decision goals* aim at taking the appropriate actions to fulfil a strategic goal, e.g. *"open new stores"*. Finally, *information goals* are related to the information required by a decision goal to be achieved; e.g. *"analyze customer purchases"*. Once these goals are defined, information requirements can be directly obtained from the information goals. The various MD elements, such as *facts* or *dimensions*, will be discovered from these information requirements in order to specify the corresponding conceptual MD model of the DW.

In order to model goals and information requirements in a CIM, the UML (Unified Modeling Language) [13] has been used together with the i^* modeling framework [21]. Specifically, we have developed [9] (i) a UML profile for i^*, in order to integrate it within our MDA framework; and (ii) a UML profile which adapts i^* to the DW domain. Both profiles are described in Fig. 2.

The i^* modeling framework provides mechanisms with which to represent the various DW actors, their dependencies, and with which to structure the business goals that the organization wishes to achieve. Decision makers' goals are defined by using the *Strategic*, *Decision*, and *Information* stereotypes. Information requirements (*Requirement*) are derived from information goals and are represented as stereotyped tasks. Furthermore, the requirement analysis for DWs necessitates the addition of certain MD concepts (in the sense of [3]). Therefore, the following concepts are added to the CIM as stereotyped resources: business processes related to the goals of decision makers (*BusinessProcess* stereotype), relevant measures related to decision makers' information requirements (*Measure*), and the contexts needed to analyze these measures (*Context*). Additionally, foreseen relations between the contexts of analysis are modeled. For instance, the city and the country contexts are related because cities can be aggregated in countries. In order to model these relationships, we have used the (shared) aggregation relationship of UML.

An example of how to use our approach is illustrated through a running example based on the strategic educational plan of the University of Alicante[2]. This plan determines goals and actions which must be undertaken in order to provide a high-quality education program. This running example specifically focuses on developing a DW that will support decision making in the *"assessment"* process. This process is related to one main actor, the *"education manager"*, via the strategic goal *"provide a good education program"*. Three different decision goals are derived from this strategic goal: *"adapt education program to demand"*, *"achieve international recognition"*, and *"attain renowned program"*. The following information goals have been obtained from each of these decision goals : *"evaluate environment demand"*, *"analyze international impact"*, and *"study student performance"*. The derived information requirements are as follows: *"percentage of students per city and province"*, *"percentage of foreign students"*, *"average of examination passed per subject, degree and department"*. Furthermore, the necessary measures and contexts of analysis are associated with the information requirements. The sole measure is *"examination session"*, and the elements that represent the context of analysis are *"student"*, *"city"*, *"province"*, *"country"*, *"subject"*, *"degree"*, and *"department"*. Several of these contexts of analysis are related to each other in order to aggregate data conveniently. Each of these elements is defined in a CIM according to the UML profile for i^* (see Fig. 3).

In summary, if a CIM is to be properly defined, then several steps must be followed: (i) discovering the actors (i.e. the decision makers), (ii) discovering their goals according to

[2] `http://www.ua.es`

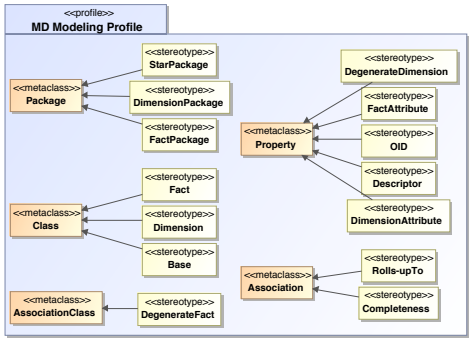


Figure 4: UML profile for MD modeling of DWs

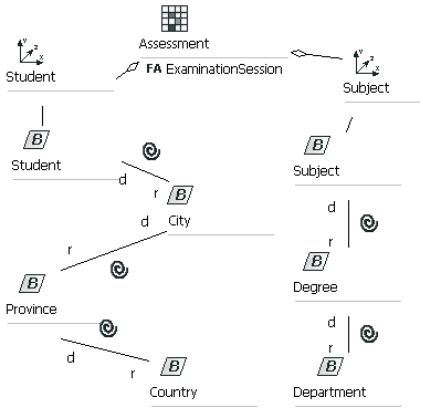


Figure 5: Initial PIM for our running example

the classification previously described, (iii) deriving information requirements from information goals, and (iv) obtaining the MD concepts related to the information requirements.

3.2 Initial Multidimensional PIM

Once goals and information requirements are specified in a CIM, a conceptual MD model that supports them must be derived in a PIM independently of any database technology. Therefore, within our MDA approach several QVT transformation rules are applied to obtain this initial PIM, thus assuring traceability between goals and information requirements in the CIM and MD elements in the PIM.

The definition of the PIM is based on our UML profile for conceptual MD modeling presented in [7]. This profile contains the stereotypes and constraints necessary to elegantly represent main MD properties at the conceptual level by means of a UML class diagram (see Fig. 4).

To automatically obtain a PIM from the CIM we have developed a set of QVT relations including four main rules [9]: (i) a hierarchy of goals together with a business process is transformed into a fact, (ii) measures, which are linked to an information requirement, are transformed into fact attributes within the corresponding fact, (iii) contexts for analyzing a fact are transformed into dimensions, and (iv) initial hierarchies can be discovered from the relations between these contexts of analysis.

By applying this QVT transformation to the CIM of our running example, the initial PIM (shown in Fig. 5) is obtained: a *Fact* class *Assessment* is created with a *FactAttribute* property *ExaminationSession*; two *Dimension* classes, and their hierarchy levels (*Base* classes), are also created according to the contexts of analysis defined in the CIM.

3.3 Hybrid Multidimensional PIM

As has previously been described, the initial PIM is directly derived from the CIM, thus ensuring that the DW will be useful in fulfilling decision makers' goals. However, this initial PIM is defined without taking the operational data sources into account, and it may not agree with these sources because decision makers may have a limited view of them. Due to this fact, the initial PIM might not be faithful (it may not be properly populated from data sources) nor complete (it may not capture the analysis potential provided by the data sources). Therefore, if these flaws are to be avoided, then this initial PIM must be checked against the available data sources.

Interestingly, several MD normal forms have been developed [6] to reason, in a rigorous manner, about several desirable properties of a conceptual MD model derived from operational data sources (among others, faithfulness and completeness). Hence, we have developed a set of QVT relations based on these MD normal forms [12] to ensure that an initial PIM is faithful and complete with respect to the source databases, thus obtaining a hybrid PIM.

The approach through which we obtain a hybrid PIM consists of two main phases. The first is based on considering the design of the MD model as a software modernization task [10]. The aim of this task is to link MD concepts to elements of the data sources, thus facilitating the subsequent phase [17]. This first phase starts by using data reverse engineering mechanisms in order to obtain a logical representation of data sources. The motivation is that operational data sources are truly legacy systems, therefore the documentation is not generally available, it cannot be obtained, or it is too complex to be easily understandable through a manual analysis [2]. This first phase concludes with the application of a set of QVT rules to identify MD concepts (fact, dimension, measure and so on) in the logical representation of data sources in order to obtain a marked logical model.

The second phase consists of reconciling the initial PIM with the marked logical model of data sources by using a set of QVT relations based on MD normal forms [12], thus obtaining a hybrid PIM. These relations focus on detecting the functional dependencies (FDs) implied by the initial PIM and by the source databases. Specifically, *faithfulness* is checked by ensuring that the FDs implied by the initial PIM are a subset of those observed in the source databases (otherwise, some source data cannot be represented under the MD model), while *completeness* is enforced by ensuring that FDs among dimension levels contained in the source databases are represented as *Rolls-upTo* associations in the PIM and that FDs among sets of measures contained in the source databases are represented via derivation formulas in the PIM (otherwise analysis potential is lost in the MD model). Furthermore, MD normal forms ensure that each measure is assigned to a fact at the "right" level of detail (without redundancies). The set of developed QVT relations

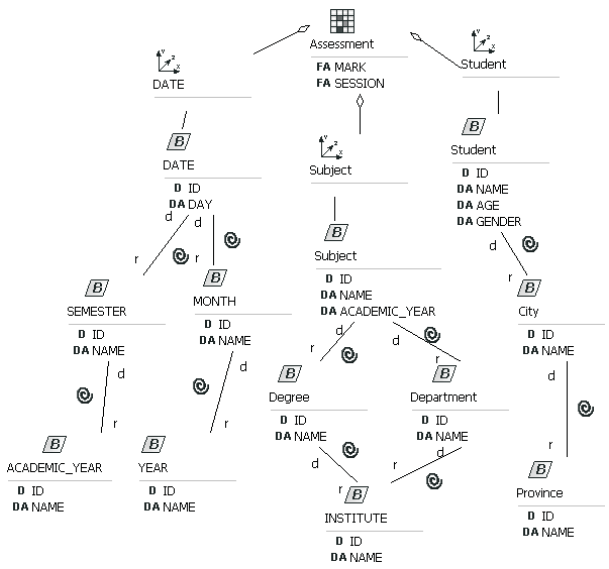


Figure 6: Hybrid PIM for our running example

based on the MD normal forms enforces these properties by removing, adding or modifying elements in the initial PIM, thus obtaining the hybrid PIM.

For example, to ensure faithfulness, *ExaminationSession*, *Country* and the *Rolls-upTo* association between *Degree* and *Department* are removed from the PIM of our running example (see Fig. 6) because they have no corresponding counterparts in the data source model. On the other hand, to ensure completeness, the hybrid PIM of Fig. 6 shows new useful elements which are part of the data sources but which do not appear in the initial PIM (e.g. the *Dimension* class *Date* or the *Base* class *Institute* which is the right way to relate *Degree* and *Department* to each other).

The great benefit of this hybrid PIM lies in that it faithfully represents the data sources and completely captures their analysis potential whilst decision makers' information requirements are simultaneously fulfilled.

3.4 Multidimensional PSM

A PSM represents the model of the same system specified by the PIM, but it also captures how that system makes use of one specific platform or technology. In MD modeling, platform-specific means that the PSM is specially designed for a kind of database technology: relational technology (relational database to store multidimensional data), multidimensional technology (structures the data directly in multidimensional structures) or XML technology (semistructured database to store multidimensional data).

In our approach, each PSM is modeled by using CWM (Common Warehouse Metamodel) [13]. Basically, CWM is a metamodel definition for interchanging DW specifications between different platforms and tools. CWM provides a set of metamodels that are comprehensive enough to be able to model an entire DW including data sources, ETL processes, MD modeling, relational implementation of a DW, and so on. These metamodels are intended to be generic, external representations of metadata in order to ensure their interchange among different platforms and tools. Specifically, the Resource layer of CWM contain several metamodels with which to represent in a PSM the structure of data according to different technologies: (i) *relational metamodel* to represent every aspect of relational databases (tables, columns, primary keys, etc.); (ii) *multidimensional metamodel* to represent commonly used multidimensional data structures; or (iii) *XML metamodel* to represent common metadata describing XML data resources.

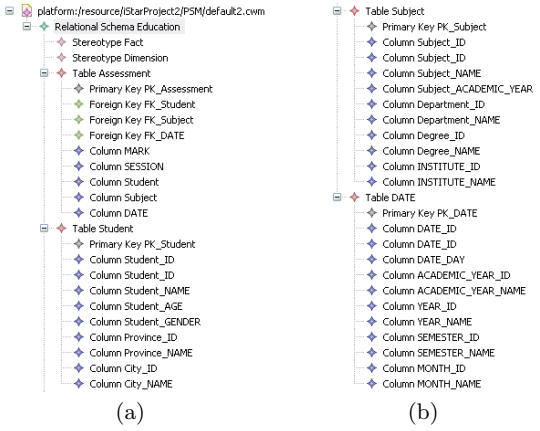


Figure 7: PSM for our running example

Following our running example, the PSM corresponds to the most common logical representation of MD models: the relational *star schema* [5]. This schema consists of a central fact table with a composite key which is joined to several dimension tables, each with a single primary key (see Fig. 7).

A set of QVT relations has been defined to establish formal and automatic transformations between the conceptual MD model and the star schema, i.e. between our hybrid PIM and the PSM. For the sake of understanding, these relations are informally described (the corresponding formal QVT relations are shown in [11]):

- Each dimension in the PIM is transformed into a table in the PSM. The descriptor of the dimension is the primary key of the new table.
- Once the dimensions have been transformed, each fact is transformed into a table. This new table has several columns which are foreign keys to every dimension table previously created. Moreover, the primary key of this fact table is a compound of every column that is a foreign key to a dimension table.
- Each dimension attribute belonging to a hierarchy level is transformed into a column in the PSM. These columns belong to the dimension table that is related to the hierarchy level.
- The fact attributes that belong to the fact class are translated into columns of the respective fact table.
- Degenerate dimensions belonging to a fact class are transformed into columns of the fact table. Moreover, degenerate dimensions take part of the primary key of the fact table.

- Degenerate facts are presented when there is a many-to-many relationship between fact and dimension. They are represented as bridge tables in the PSM.

It is worth noting that, although our PSM is assumed to be a star schema, other transformations can be specified in order to obtain different kinds of PSMs, e.g. a PSM directly based on multidimensional technology (as shown in [8]).

Finally, a set of Mof2Text transformations have been developed to obtain the corresponding code for each PSM. For instance, a relational PSM would derive SQL code. However, since the CWM metamodels are close to their respective technologies, deriving code is a straightforward task which is not dealt with in this paper.

4. CONCLUSIONS AND FUTURE WORK

A DW is an integrated collection of historical data which supports management's decisions. According to this definition, in the development of a MD model for the DW, it is not only important to take into account the information needs of decision makers (requirement-driven approaches), but also the existing data sources that will populate the DW (data-driven approaches). Therefore, formal mechanisms are needed to integrate these two points of view in a hybrid approach. Furthermore, the MD modeling of the DW resembles the traditional database design methods [16] because it must be structured into a variety of steps during which a conceptual design phase is performed, whose results are transformed into a logical data model as the basis for schema implementation. This manner of proceeding claims for the automation of these transformations.

In this paper, a framework for dealing with these issues has been presented in order to obtain a major benefit: the systematic, well structured and comprehensive development of a hybrid MD model at the conceptual level and the automatic derivation of its logical representation. Specifically, our hybrid MD modeling framework aligns with MDA by (i) specifying the information requirements in a CIM, (ii) describing how to derive an initial PIM for the MD modeling from the CIM, (iii) reconciling this PIM with the information provided by the available data sources which will populate the DW, thus obtaining a hybrid PIM, (iv) using CWM to build PSMs tailored to various database technologies, and (v) formally establishing QVT relations between these models and Mof2Text transformations in order to obtain the corresponding code. Therefore, our framework allows designers to decrease the inherent complexity of DW development, thus saving time and effort.

Our immediate future work will focus on considering complex MD structures such as heterogeneous dimension hierarchies [14] in the PIM and how they are translated into a PSM, thus avoiding inaccurate queries.

5. REFERENCES

[1] A. Abelló, J. Samos, and F. Saltor. YAM^2: a multidimensional conceptual model extending UML. *Inf. Syst.*, 31(6):541–567, 2006.

[2] R. Alhajj. Extracting the extended entity-relationship model from a legacy relational database. *Inf. Syst.*, 28(6):597–618, 2003.

[3] P. Giorgini, S. Rizzi, and M. Garzetti. GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decis. Support Syst.*, 45(1):4-21, 2008.

[4] M. Golfarelli, D. Maio, and S. Rizzi. The Dimensional Fact Model: A conceptual model for data warehouses. *Int. J. Cooperative Inf. Syst.*, 7(2-3):215–247, 1998.

[5] R. Kimball and M. Ross. *The Data Warehouse Toolkit*. Wiley & Sons, 2002.

[6] J. Lechtenbörger and G. Vossen. Multidimensional normal forms for data warehouse design. *Inf. Syst.*, 28(5):415–434, 2003.

[7] S. Luján-Mora, J. Trujillo, and I.-Y. Song. A UML profile for multidimensional modeling in data warehouses. *Data Knowl. Eng.*, 59(3):725–769, 2006.

[8] J.-N. Mazón, J. Pardillo, and J. Trujillo. Applying transformations to model driven data warehouses. In *DaWaK*, volume 4081 of *Lecture Notes in Computer Science*, pages 13–22. Springer, 2006.

[9] J.-N. Mazón, J. Pardillo, and J. Trujillo. A model-driven goal-oriented requirement engineering approach for data warehouses. In *ER Workshops*, volume 4802 of *Lecture Notes in Computer Science*, pages 255–264. Springer, 2007.

[10] J.-N. Mazón, and J. Trujillo. A Model Driven Modernization Approach for Automatically Deriving Multidimensional Models in Data Warehouses. In *ER*, volume 4801 of *Lecture Notes in Computer Science*, pages 56–71. Springer, 2007.

[11] J.-N. Mazón and J. Trujillo. An MDA approach for the development of data warehouses. *Decis. Support Syst.*, 45(1):41-58, 2008.

[12] J.-N. Mazón, J. Trujillo, and J. Lechtenbörger. Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms. *Data Knowl. Eng.*, 63(3):699–725, 2007.

[13] Object Management Group. `http://www.omg.org`.

[14] E. Pourabbas, and M. Rafanelli. Hierarchies and Relative Operators in the OLAP Environment. *SIGMOD Record 29(1)*: 32-37 (2000)

[15] N. Prakash, Y. Singh, and A. Gosain. Informational scenarios for data warehouse requirements elicitation. In *ER*, Volume 3288 of *Lecture Notes in Computer Science*, pages 205–216, Springer 2004.

[16] S. Rizzi, A. Abelló, J. Lechtenbörger, and J. Trujillo. Research in data warehouse modeling and design: dead or alive? In *DOLAP* 2006, pages 3–10.

[17] I.-Y. Song, R. Khare, and B. Dai. SAMSTAR: a semi-automated lexical method for generating star schemas from an entity-relationship diagram. In *DOLAP* 2007, pages 9–16.

[18] N. Spyratos. A functional model for data analysis. In *FQAS* 2006, pages 51-64.

[19] P. Vassiliadis, A. Simitsis, P. Georgantas, M. Terrovitis, S. Skiadopoulos. A generic and customizable framework for the design of ETL scenarios. *Inf. Syst. 30(7)*: 492-525 (2005)

[20] B. Vela, E. Fernández-Medina, E. Marcos, and M. Piattini. Model driven development of secure XML databases. *SIGMOD Record 35(3)*: 22-27 (2006)

[21] E. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *RE'97*, pages 226-235.

Machine Models for Query Processing

Nicole Schweikardt

Institut für Informatik, Goethe-Universität Frankfurt am Main
schweika@informatik.uni-frankfurt.de

1 Introduction

The massive data sets that have to be processed in many application areas are often far too large to fit completely into a computer's internal memory. When evaluating queries on such large data sets, the resulting communication between fast internal memory and slower external memory turns out to be a major performance bottleneck.

Modern software and database technologies use clever heuristics to minimize the costs produced by external memory accesses. There has been a wealth of research on query processing and optimization along these lines (cf. e.g. [31, 16, 40, 27]), and it seems that the current technologies scale up to current user expectations.

Our theoretical understanding of the problems involved, however, is not quite as developed. Most results concerning the computational complexity of query languages are formulated in terms of classical complexity classes such as PTIME or PSPACE. Two examples of such results are that the combined complexity of evaluating relational algebra queries is PSPACE-complete [38, 37] and that the combined complexity of evaluating acyclic conjunctive queries belongs to PTIME [42] and is in fact LogCFL-complete [15]. The classes considered in computational complexity theory are usually based on Turing machines or random access machines, i.e., on machine models that do not take into account the existence of multiple storage media of varying sizes and access characteristics. Since the performance bottleneck for communicating between such storage media is completely ignored in classical complexity classes, "classical" results on the complexity of query evaluation only give a very rough measure of the complexity of query evaluation on a real computer.

In recent years, a number of machine models have been developed that take into account the existence of multiple storage media of varying sizes and access characteristics. These models are particularly useful for studying the complexity of query evaluation on massive data sets. The present paper gives an overview of such machine models. The two "extreme" models are the (very powerful) *external memory model*, presented in Section 2, and the (severely restricted) *mud model* (a model for massive, unordered, distributed computations), presented in Section 7. Further models considered in this survey are the *read/write streams* (a model for sequential external memory processing, Section 3), the *finite cursor machines* (a model for relational database query processing, Section 4), the *mpms-automata* (a model for processing indexed XML files, Section 5), and the *data stream model* (a model for processing data on-the-fly, Section 6).

As running examples throughout this article, we will take a closer look at the problem of *sorting* a given set of data items and at the problem of deciding whether two sets of data items are *disjoint*. Note that these two problems are of fundamental importance for query processing: Efficient query evaluation often relies on intermediate sorting steps; and already the easiest kind of *join* (i.e., the join of two *unary* relations) corresponds to computing the intersection of two sets.

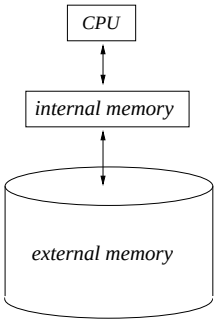
2 The External Memory Model

In the classical *random access machine* (RAM) model, the input consists of N data items, each of which can be represented by a bitstring of length $O(\log N)$. An unbounded number of data items can be stored in memory. Access to any item in memory as well as arithmetics and bitwise operations on data items can be performed in constant time.

The basic *external memory model* (cf., e.g., [27]) can be viewed as a refinement of the RAM model where memory is divided into internal memory, capable of storing up to M data items, and external memory of unbounded size. Data present in internal memory can be accessed quickly (as in the RAM model). Data present in external memory can only be accessed by an operation called *Input/Output communication* (I/O, for short) that moves a contiguous block of B data items between internal and external memory. Initially, the N input items are stored in external memory. One typically assumes that $M < N$ and $1 \leqslant B \leqslant M/2$. An illustration of the model is given in Figure 1.

The primary measure of performance in the external memory model is the number of I/O operations performed. Further relevant measures are (as in the RAM model) the total number of computation steps and the

Database Principles Column. Column editor: Leonid Libkin, School of Informatics, University of Edinburgh, Edinburgh, EH8 9AB, UK. E-mail: libkin@inf.ed.ac.uk.



$N =$ problem size (# data items)

$M =$ internal memory size (# data items)

$B =$ block transfer size (# data items)

$M < N, \quad 1 \leqslant B \leqslant M/2.$

Figure 1: The basic external memory model.

total amount of memory that is used.

This model is the most widely used model for designing and analyzing efficient external memory algorithms (cf., [27, 40]). On the one hand, it allows to design algorithms that explicitly control the communication between internal and external memory, so that algorithms that are "good" with respect to the model also perform well in practice. On the other hand, it is simple enough to allow for a high-level description of an algorithm and a rigorous mathematical analysis of its performance.

Obviously, *scanning*, i.e., reading the entire input in the order it is stored in external memory, takes

$$\mathit{scan}(N) \;=\; \Theta(N/B) \quad \text{I/O operations.}$$

A careful implementation of the *merge sort* algorithm shows that *sorting* can be performed using

$$\mathit{sort}(N) \;=\; \Theta\left(\tfrac{N}{B} \cdot \log_{M/B} \tfrac{N}{B}\right) \quad \text{I/O operations.}$$

Aggarwal and Vitter [2] proved a matching lower bound on the worst case complexity of sorting, provided that the machine model is restricted in such a way that the input data items are *indivisible* (i.e., they cannot be split up into their representations as bitstrings) and that at any point in time, the external memory consists of a permutation of the input items.

The *Parallel Disk Model*, introduced by Vitter and Shriver [41], is a generalization of the basic external memory model. In this model, the external memory is partitioned into D independent disks, such that during each I/O operation, each of the D disks can simultaneously transfer a block of B contiguous data items into internal memory (one typically assumes that $1 \leqslant D{\cdot}B \leqslant M/2$). Furthermore, instead of a single CPU there are P identical processors that work in parallel and that are connected by a network. Each of these processors has access to internal memory of size M/P. Furthermore, if $D \geqslant P$, then each processor owns D/P of the D disks, and if $D < P$, then each disk is shared by P/D processors.

It is obvious that *scanning* can be done by performing $\Theta(N/DB)$ I/O operations. Concering *sorting*, an elaborate construction by Nodine and Vitter shows the following:

Theorem 2.1 ([29]) *Sorting can be performed in the Parallel Disk Model using* $O\left(\frac{N}{DB} \cdot \log_{M/B} \frac{N}{B}\right)$ *I/O operations.*

The algorithm treats data items as indivisible units, and at any point in time during the execution of the algorithm, the content of external memory consists of a permuation of the input. For this restricted version of the Parallel Disk Model, a matching lower bound for sorting was obtained by Aggarwal and Vitter in [2].

Suggestions for further reading: An overview of external memory algorithms for various computation problems is given in [39]; more details can be found in the survey [40] and the books [1, 27].

An important future task: Concerning the external memory model and its extensions, a number of lower bound results are known, among them the lower bound for *sorting* mentioned above. A common feature of these lower bounds is that they rely on the assumption that the input data items are *indivisible* and that at any point in time the external memory consists, in some sense, of a permutation of the input items. It is a challenging future task to develop methods for proving lower bounds for the external memory model or the Parallel Disk Model without relying on such an indivisibility assumption.

3 Read/Write Streams

The external memory model considered in the previous section distinguishes between accesses to internal memory and accesses to external memory. Current technology for external storage systems (disks and tapes), however, presents us with a situation where *sequential scans* are strictly preferable over *random accesses* to external memory: A random access to a hard disk requires to move the read/write head to a certain position of the disk, and this is a comparably slow mechanical operation. During the time required for a single such *random* access, a considerable amount of data stored *in sequence*, starting at the current position of the disk's read/write head could have been processed. Modern software and database technologies use clever heuristics to minimize the number of accesses and to prefer streaming over random accesses to external memory.

The present section concentrates on a machine model for external memory processing which was introduced in [21, 22] and which has available

- *internal memory* that can be accessed very fast, but that is limited in size, and, additionally,
- *external memory* that can store huge amounts of data, which can easily be accessed (for reading as well as for writing) in a sequential way, but for which *random accesses* are expensive.

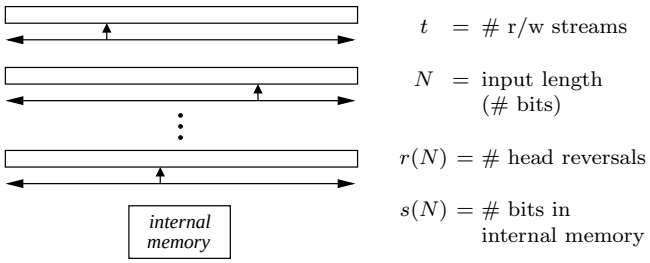


Figure 2: The model of read/write streams.

Formally, this model is based on a standard multi-tape Turing machine. Some of the tapes of the machine, among them the input tape, represent external memory (each of these tapes can be viewed as representing an external memory device such as, e.g., a hard disk). They are unrestricted in size, but access to these tapes is restricted by allowing only a certain number $r(N) - 1$ of reversals of the head directions (where N denotes the size of the input, measured in terms of the number of bits necessary for representing the entire input). This may be seen as a way of (a) restricting the number of sequential scans and (b) restricting random accesses to these tapes (because each random access can be simulated by moving the head to the desired position on a tape, and this involves at most two head reversals). The remaining tapes of the Turing machine represent internal memory. Access to these internal memory tapes is unrestricted, but their size is bounded by a parameter $s(N)$. Such Turing machines are called (r, s, t)-*bounded*, if t is the total number of external memory tapes of the machine.

It sometimes is convenient to adopt Beame, Jayram, and Rudra's [7] informal view of this as being a computation model where, in addition to some internal memory, a constant number of *read/write streams* (r/w, for short) are available; each such r/w stream corresponds to an external memory tape of the Turing machine. An illustration of the model is given in Figure 2. The resources of interest are

(1) the number $r(N)$ of scans of (or, head reversals on) the r/w streams (where N denotes the number of bits necessary for representing the entire input),

(2) the size $s(N)$ of internal memory, and

(3) the total number t of r/w streams that are available.

In the remainder of this article, we call this the *computational model of r/w streams*, and we sometimes informally say that there is an (r, s, t)-*bounded algorithm on r/w streams* for a specific problem, iff this problem can be solved by an (r, s, t)-bounded Turing machine.

When considering problems that produce an output other than just the answer "yes" or "no", we assume that there is an additional write-only output stream available.

Remark 3.1 Note that this model allows *forward* scans as well as *backward* scans of the external memory tapes. Even more, it allows the r/w heads to reverse direction in the middle of a tape. While it is realistic to assume that in current "real-world" hard disks, sequential *forward* scans can be performed very efficiently, this is not the case for *backward* scans. Thus, when aiming at designing efficient algorithms for r/w streams, one should make sure that the algorithms essentially use only forward scans of the r/w streams. On the other hand, when aiming at *lower bounds*, i.e., showing that some problem can *not* be solved by any (r, s, t)-bounded algorithm on r/w streams, allowing for forward scans as well as backward scans makes lower bound results only stronger.

Having in mind the motivation that (r, s, t)-bounded algorithms on r/w streams serve as a computation model for external memory processing, we are mainly interested in cases where the size $s(N)$ of the internal memory is considerably smaller than the input size N, and the number $r(N)$ of sequential scans of (or, correspondingly, the number of random accesses to) external memory is, preferably, as small as possible.

Note that a priori there is no restriction on the running time or the size of the external memory of an (r, s, t)-bounded Turing machine. However, an easy induction on $r(N)$ shows that implicitly these two parameters are bounded in terms of the number of head reversals, the internal memory space, and the input size: Every run of an (r, s, t)-bounded Turing machine on an input of length N consists of at most $N \cdot 2^{O(r(N)\cdot(t+s(N)))}$ computation steps (cf., [20]).

There are substantial differences between the computational power of the model where only *one* r/w stream is available and the model where $t \geqslant 2$ r/w streams are available. Both scenarios will be considered in the next two subsections.

3.1 One r/w stream

Let us first consider the problem *short-sorting*, which is the restriction of the sorting problem to inputs that consist of a list of an arbitrary number m of bitstrings $x_1, \ldots, x_m$, each of which has length at most $2 \log m$. Note that the total length of the input is $N = \Theta(m \log m)$.

It can be easily seen that, for any function s with $s(N) \geqslant 4 \log N$, the problem *short-sorting* can be solved by an $(r, s, 1)$-bounded algorithm on one r/w stream with $r(N) = O(N/s(N))$ sequential scans and internal memory of size $s(N)$. A matching lower bound was proved in [21], leading to the following result.

Theorem 3.2 ([21]) *Let r, s be functions such that $s(N) \geqslant 4 \log N$. The problem short-sorting can (respectively, cannot) be solved by an $(r, s, 1)$-bounded algorithm on one r/w stream if $r(N) \cdot s(N)$ is of size $\Omega(N)$ (respectively, of size $o(N)$).*

The basic idea for proving a lower bound on resources necessary for solving a problem in the computation model with one r/w stream is to divide the r/w stream into two parts. Obviously, during an $(r, s, 1)$-bounded computation on an input of size N, the border between the two parts of the r/w stream can be crossed at most $r(N)$ times. Each time we cross this border, we can only "transport" the amount of information that is currently stored in internal memory, and this consists of at most $O(s(N))$ bits. Consequently, during an entire $(r, s, 1)$-bounded computation, only $O(r(N) \cdot s(N))$ bits of information can be communicated between the two parts of the r/w stream. Therefore, lower bounds known from *communication complexity* (cf., [26]) almost immediately imply corresponding lower bounds for $(r, s, 1)$-bounded algorithms on one r/w stream.

Using this, one for example obtains that the *set disjointness* problem cannot be solved by an $(r, s, 1)$-bounded algorithm on one r/w stream if $r(N) \cdot s(N)$ is of size $o(N)$. Here, the *set disjointness* problem receives as input two lists of bitstrings $x_1, \ldots, x_m$ and $y_1, \ldots, y_m$, and the task is to decide whether $\{x_1, \ldots, x_m\} \cap \{y_1, \ldots, y_m\} = \emptyset$.

By using this lower bound, one immediately obtains that evaluating *relational algebra queries* is hard for algorithms on one r/w stream: Note that if A and B are unary relations, then $A \bowtie B = A \cap B$. Therefore, already the basic task of checking whether the join of two relations A and B is empty cannot be performed if $r(N) \cdot s(N)$ is of size $o(N)$ (where N denotes the number of bits used for storing the two input relations A and B).

In a similar way one also obtains lower bounds for evaluating queries against XML data. For example, two unary relations A and B can easily be encoded by an XML document, such that the query which asks if the join of A and B is empty can be formalized in the language *XQuery*. This immediately leads to the result stating that there exists an *XQuery* query Q such that, for all functions r, s with $r(N) \cdot s(N) = o(N)$, there is no $(r, s, 1)$-bounded algorithm on one r/w stream, which receives an XML document D of length N as input and checks whether the result of Q on D is empty.

Considering the node-selecting XML query language *Core XPath* (cf., [14]), let $Q(D)$ denote the set of nodes that Q selects in an XML document D. In [21], the following tight bounds on the complexity of processing *Core XPath* queries have been obtained:

(1) For every Core XPath query Q there is an algorithm which decides for an input XML document D, whether $Q(D) \neq \emptyset$. This algorithm performs a single pass over its input and uses internal memory of size $O(h(D))$, where $h(D)$ denotes the height of the tree representation of D.

(2) There is a Core XPath query Q such that for all functions r, s with $r(H) \cdot s(H) = o(H)$, there exists no algorithm on one r/w stream which, when given as input an XML document D, decides whether $Q(D) \neq \emptyset$ and uses at most $r(h(D))$ head reversals and internal memory space of size at most $s(h(D))$.

The upper bound (1) is proved by standard automata theoretic techniques. For the lower bound (2), one can again use the lower bound for the *set disjointness* problem.

Let us end this subsection with an example exposing that *randomized* $(r, s, 1)$-bounded algorithms are computationally much stronger than deterministic ones. We introduce randomization to the computation model in such a way that in each computation step a coin may be tossed to decide what to do in this step. The probability $\Pr(A \text{ accepts } w)$ that input w gets accepted by algorithm A is then defined in a straightforward way (see [33, 20] for precise definitions).

Let us consider the problem *short-multiset-equality*, which receives as input two lists of bitstrings $x_1, \ldots, x_m$ and $y_1, \ldots, y_m$, where each x_i and each y_j has length at most $2 \log m$ (for arbitrary m). The task is to decide whether the multisets $\{x_1, \ldots, x_m\}$ and $\{y_1, \ldots, y_m\}$ are equal, i.e. whether they contain the same elements with the same multiplicities. Note that the input length is $N = \Theta(m \log m)$.

An easy communication complexity argument shows that there is no deterministic $(r, s, 1)$-bounded algorithm on one r/w stream that solves this problem with $r(N) \cdot s(N) = o(N)$. In particular, this implies that every deterministic algorithm that solves the problem *short-multiset-equality* by performing a single pass over the input requires internal memory of size $\Omega(N)$ (see [33] for details). However, by using standard fingerprinting techniques, the problem can be solved by a randomized algorithm with internal memory of size $O(\log N)$ as follows:

Theorem 3.3 ([20]) *There is a randomized algorithm which, when given the parameter m, performs a single pass over the input and solves the short-multiset-equality problem with internal memory of size $O(\log N)$ such that*

- *each "yes"-instance is accepted with probability 1, and*
- *each "no"-instance is rejected with probability $\geqslant 2/3$.*

Proof sketch: The basic idea is to identify the input strings x_i, y_j with natural numbers and to associate two polynomials $f(z) := \sum_{i=1}^{m} z^{x_i}$ and $g(z) := \sum_{j=1}^{m} z^{y_j}$ with the input $x_1, \ldots x_m, y_1 \ldots y_m$. Obviously, the two polynomials are equal if, and only if, the input is a "yes"-instance of the *short-multiset-equality* problem.

The basic idea of the algorithm is to choose a random number r, to evaluate $f(r)$ and $g(r)$, and to accept if, and only if, $f(r) = g(r)$. This algorithm will accept with probability 1 if the input is a "yes"-instance. However, if the input is a "no"-instance, the two polynomials f and g only have few common points, and thus the algorithm will reject with high probability.

A closer look shows that this procedure can be implemented by performing a single pass over the input. The algorithm computes $f(r)$ and $g(r)$ on-the-fly and uses arithmetic modulo a prime number of moderate size such that $O(\log N)$ bits of internal memory suffice. □

3.2 Several r/w streams

Let us now turn to the computation model where an arbitrary number t of r/w streams are available.

A straightforward implementation of the *merge sort* algorithm shows that the problem *short-sorting* can be solved by using three r/w streams, $O(\log N)$ head reversals, and internal memory of size $O(\log N)$. I.e., *short-sorting* can be solved by an $(O(\log N), O(\log N), 3)$-algorithm on r/w streams. By using suitable coding tricks, one can perform sorting even with internal memory of size $O(1)$ and with a total number of two r/w streams [10]. Thus, using two r/w streams instead of a single one substantially increases the power of the computation model (cf., Theorem 3.2, stating that any $(r, s, 1)$-bounded algorithm for *short-sorting* requires that $r(N) \cdot s(N) = \Omega(N)$). The following theorem shows that this is essentially optimal: If instead of $O(\log N)$ only $o(\log N)$ head reversals are available, even internal memory of size $O(N^{1-\varepsilon})$ and an arbitrary number t of r/w streams do not suffice to solve *short-sorting*.

Theorem 3.4 ([20]) *Let t be a positive integer and let ε be an arbitrary number with $0 < \varepsilon < 1$.*
The problem short-sorting cannot be solved by any $(o(\log N), O(N^{1-\varepsilon}), t)$-bounded algorithm on r/w streams.

To prove this theorem, straightforward communication complexity based arguments (as used in the previous subsection) utterly fail. The reason is that we can easily communicate arbitrarily many bits from one part of the input to any other part by just copying the first part to a second r/w stream and then reading it in parallel with the second part of the input. This requires no internal memory and just two head reversals on the r/w streams.

But still, although the use of several r/w streams allows to copy large consecutive segments of the input from one place to another, there seems no easy way of significantly *permuting* the input without using too many head reversals. This intuition was confirmed in [22, 20]. Note that, unlike the lower bounds mentioned in Section 2, Theorem 3.4 does not rely on any kind of "indivisibility" assumption. Furthermore, the lower bound of Theorem 3.4 even holds for randomized algorithms which output either the correctly sorted sequence or the answer *"I don't know"*, and where the answer *"I don't know"* is returned with probability $\leqslant 1/2$ (cf., [20]).

Note that the *short-multiset-equality* problem as well as the *set disjointness* problem can easily be reduced to the sorting problem. Thus, both problems can be solved by a deterministic $(O(\log N), O(1), 2)$-bounded algorithm on r/w streams. Furthermore, from Theorem 3.3 we obtain a randomized $(O(1), O(\log N), 1)$-bounded algorithm for *short-multiset-equality* which never produces false negative answers, and which produces false positive answers with probability $\leqslant 1/3$. The next theorem shows that no corresponding algorithm with complementary acceptance and rejectance probabilities exists (not even when allowing up to $o(\log N)$ head reversals, internal memory of size $O(N^{1-\varepsilon})$, and an arbitrary number of r/w streams).

Theorem 3.5 ([20]) *Let t be a positive interger and let ε be an arbitrary number with $0 < \varepsilon < 1$.*
There is no $(o(\log N), O(N^{1-\varepsilon}), t)$-bounded randomized algorithm for the short-multiset-equality problem on r/w streams such that

- *each "no"-instance is rejected with probability 1, and*
- *each "yes"-instance is accepted with probability $\geqslant 2/3$.*

For the *set disjointness* problem, in the presence of considerably less than $\log N$ head reversals, not even a randomized algorithm with 2-sided bounded error exists:

Theorem 3.6 ([7]) *Let t be a positive interger and let ε be an arbitrary number with $0 < \varepsilon < 1$.*
There is no $(o(\log N / \log\log N), O(N^{1-\varepsilon}), t)$-bounded randomized algorithm for the set disjointness problem on r/w streams such that

- *each "no"-instance is rejected with probability $\geqslant 2/3$,*
- *each "yes"-instance is accepted with probability $\geqslant 2/3$.*

The above results easily lead to the following statements on the data complexity of relational algebra queries and queries posed against XML data [20, 7] (for any choice of $t \geqslant 1$ and ε with $0 < \varepsilon < 1$).

(1) For every relational algebra query Q, the problem of evaluating Q on a stream consisting of the tuples of the input database relations, can be solved by an $(O(\log N), O(1), 2)$-bounded deterministic algorithm on r/w streams.

(2) There exists a relational algebra query Q_1 such that the problem of evaluating Q_1 on a stream of the tuples of the input database relations cannot be solved by any $(o(\log N), O(N^{1-\varepsilon}), t)$-bounded algorithm on r/w streams.

(3) The task of checking whether the join of two relations A and B is empty cannot be performed by any $(o(\log N / \log\log N), O(N^{1-\varepsilon}), t)$-bounded randomized algorithm on r/w streams with a two-sided bounded error of at most 1/3.

(4) There is an XQuery query Q_2 such that the problem of evaluating Q_2 on an input XML document of length N cannot be solved by any $(o(\log N), O(N^{1-\varepsilon}), t)$-bounded algorithm on r/w streams.

(5) There is an XPath query Q_3 such that the problem of checking, for an input XML document D of length N, whether $Q_3(D) \neq \emptyset$, cannot be solved by any $(o(\log N), O(N^{1-\varepsilon}), t)$-bounded randomized algorithm on r/w streams with 1-sided bounded error that accepts "yes"-instances with probability 1 and that rejects "no"-instances with probability $\geqslant 2/3$.

(6) There is an XQuery query Q_4 such that the problem of checking whether the result of Q_4 on an input XML document of length N is empty cannot be solved by any $(o(\log N/\log\log N), O(N^{1-\varepsilon}), t)$-bounded randomized algorithm on r/w streams with a two-sided bounded error of at most 1/3.

Suggestions for further reading: An overview of the model of r/w streams is given in [33]; technical details can be found in the articles [21, 20, 25, 7, 6].

A related computation model based on r/w streams and intermediate sorting steps is the *StrSort* model of [3] that was further considered in [32]. In [12], the *W-Stream* model, a restriction of the StrSort model in which intermediate sorting steps are prohibited, was introduced. This model can also be viewed as a restriction of the computation model of r/w streams with a single r/w stream, where only forward scans are allowed.

An important future task: It would be interesting to further study the computational power of the model with multiple r/w streams and intermediate sorting steps. First steps in this direction were taken in [3, 32]. Similarly as with the external memory model from Section 2, the lower bound proofs currently known for this model rely on the assumption that data items are indivisible and that only comparisons between data items are allowed. It is a challenging future task to develop methods for proving lower bounds in the StrSort model that do not rely on such an indivisibility assumption.

4 Finite Cursor Machines

Finite cursor machines (FCMs, for short) were introduced in [19] as an abstract model of database query processing. Informally, they can be described as follows:

The input for an FCM is a relational database. Each relation is represented by a *table*, i.e., an ordered list of rows, where each row corresponds to a tuple in the relation. The *size* n of the input is defined as the sum of the number of rows of each input table.

Data elements are viewed as "indivisible" objects that can be manipulated by a number of built-in operations. We assume that all data items belong to a fixed, infinite universe $\mathbb{D}$, which is equipped with a number of built-in predicates (including, e.g., the equality predicate and a linear order). This feature is very convenient for modeling standard operations on data types like integers, floating point numbers, or strings, which may all be part of the set

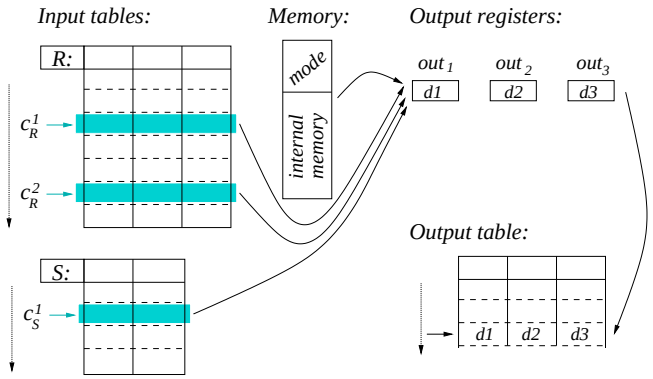


Figure 3: The model of finite cursor machines.

$\mathbb{D}$. The universe $\mathbb{D}$ together with the built-in predicates form the so-called *background structure*.

FCMs can operate in a finite number of *modes* (corresponding to states of a finite automaton). Additionally, they can use *internal memory* in which they can store bitstrings. We speak of $O(1)$-FCMs (respectively, $o(n)$-FCMs) when restricting attention to FCMs where internal memory is of constant size (respectively, of size $o(n)$). FCMs access each input relation through a finite number of *cursors*, each of which can read one row of a table at any time. The model incorporates certain *streaming* aspects by imposing a restriction on the movement of the cursors: They can move on the tables sequentially, in one direction only. Thus, once the last cursor has passed a row of a table, this row can never be accessed again during the computation. Note, however, that several cursors can be moved asynchronously over the same table at the same time, and thus, entries in different, possibly far apart, regions of the table can be read and processed simultaneously.

For producing an output table, a finite number of *output registers* are available, each of them capable of storing an element in $\mathbb{D}$. Whenever the so-called *outputmode* is set to a special value *out*, the machine automatically produces the next output tuple: The tuple consisting of the values in the output registers (in some predefined order) is appended to the output table.

An illustration of the model is given in Figure 3. The formal notion of FCMs is defined in the framework of *abstract state machines*; details can be found in [19].

Figure 4 gives an example of an FCM program that computes the query Q defined on a ternary relation R over $\mathbb{N}$ that returns the sum of the first and second attribute of each row whose third attribute contains a value bigger than 100. In the example, there is a single output register out_1 and a single cursor c that accesses the rows of R, starting in the first row. The program starts with outputmode set to the value *init* and is applied again and again until the cursor has been moved beyond the last row of R. The command $c := next_R(c)$ advances the

```
 1: if outputmode = out then
 2:   par
 3:     outputmode := init
 4:     c := next_R(c)
 5:   endpar
 6: else
 7:   if attribute^3_R(c) > 100 then
 8:     par
 9:       out_1 := attribute^1_R(c) + attribute^2_R(c)
10:       outputmode := out
11:     endpar
12:   else
13:     c := next_R(c)
14:   endif
15: endif
```

Figure 4: Example of an FCM program.

cursor to the next row of R. The par/endpar construct encloses lines of code that are executed in parallel. The code in lines 7–11 enforces that if the third column of the currently visited row contains a value > 100, then the output register out_1 is filled with the sum of the values in the row's first two columns, and the outputmode *out* is activated, enforcing that the content of register out_1 is appended as a new tuple of the output table. In the next execution of the program, the code in lines 1–5 deactivates the outputmode (by setting it to the value *init*) and advances the cursor to the next tuple in R.

FCMs model quite faithfully what happens in relational database query processing. For example, the *selection* $\sigma_\theta(R)$ of all rows of R that satisfy a certain selection condition θ can be implemented by an FCM in a straightforward way. If tables are *sorted*, then also the *projection* operator and the *union*, *intersection*, and *difference* of two relations can be accomplished by an FCM. Also the *semijoin*[1] $R \ltimes_\theta S$ can be computed by an FCM operating on sorted tables, provided that the join condition θ consists of a conjunction of equalities and at most two inequalities [19]. This leads to the following observation concerning the *semijoin algebra*, i.e., the restriction of relational algebra where, instead of joins, only semijoins are allowed.

Theorem 4.1 ([19]) *Every semijoin algebra query can be computed by a query plan composed of a finite number of $O(1)$-FCMs and sorting operations.*

Notice that FCMs are capable of computing some relational algebra queries that are not expressible in the semijoin algebra. An example is the Boolean query asking whether $R = \pi_1(R) \times \pi_2(R)$, i.e., asking whether the binary relation R is the cartesian product of the projection of R to its first and second component, respectively. Also, examples of queries that are computable by FCMs but not expressible in relational algebra are known (cf., [19]).

Furthermore, *sliding window joins* for a fixed window size w (enforcing that the join operator is successively applied to portions of the data, each portion consisting of a number w of consecutive rows of an input table) can be easily computed by an FCM. Note, however, that general *joins* cannot be computed since the output size of a join may be quadratic in the size of the input, while $O(1)$-FCMs can output only a linear number of tuples.

In connection with Theorem 4.1, the question arises whether intermediate sorting steps are really necessary. The next theorem answers this question affirmatively. Let R, S, and T be relations, such that S has arity $\geqslant 2$, and consider the composition

$$R \ltimes_{x_1=y_1} (S \ltimes_{x_2=y_1} T)$$

of two semijoins. $S \ltimes_{x_2=y_1} T$ returns all tuples s in S for which there exists a tuple t in T whose first component coincides with the second component of s. $R \ltimes_{x_1=y_1} (S \ltimes_{x_2=y_1} T)$ returns all tuples r in R for which there exists a tuple s in $(S \ltimes_{x_2=y_1} T)$ whose first component coincides with the first component of r.

Theorem 4.2 ([19])
The query "Is $R \ltimes_{x_1=y_1} (S \ltimes_{x_2=y_1} T)$ nonempty?", where R and T are unary relations and S is a binary relation, is not computable by any $o(n)$-FCM, even if the input consists of all sorted versions of the relations R, S, T.

Here, the notion of "sorted versions" of a relation is defined as follows. Recall that the universe $\mathbb{D}$ of data items is equipped with a linear order $<$. We consider sorting in ascending or descending order. Then, a relation of arity p can be sorted lexicographically in $p! \cdot 2^p$ different ways: For any permutation π of $\{1, \ldots, p\}$ and any *ordering scheme* $\sigma : \{1, \ldots, p\} \to \{\text{asc}, \text{desc}\}$, let $sort_{\pi,\sigma}$ be the operation that takes a p-ary relation R and lexicographically sorts it as follows: It sorts the $\pi(1)$-th column first (ascendingly or descendingly, depending on the value of $\sigma(1)$), the $\pi(2)$-th column second (ascendingly or descendingly, depending on the value of $\sigma(2)$), ..., and sorts the $\pi(p)$-th column last (ascendingly or descendingly, depending on the value of $\sigma(p)$). Theorem 4.2 assumes for any relation, that for any applicable permuation π and ordering scheme σ, the table sorted according to $sort_{\pi,\sigma}$ is present as an input table.

Note that Theorem 4.2 is sharp in terms of arity of the input relations and in terms of size of internal memory: If S is unary (and R and T of arbitrary arities), then the corresponding query is computable by an $O(1)$-FCM on sorted inputs. Furthermore, if internal memory of size $O(n)$ (rather than $o(n)$) is available, then the entire input database can be loaded into internal memory, and the query can be processed there.

[1] $R \ltimes_\theta S$ returns all tuples r of R for which there exists a tuple s of S such that (r, s) satisfies condition θ.

Suggestions for further reading: The formal definition of FCMs as well as detailed proofs of the results mentioned in this section can be found in [19].

An important future task: The main open question from [19] is as follows: Is there a Boolean relational algebra query that cannot be computed by a finite composition of FCMs and sorting operations? The conjectured answer is "yes", since every query that can be computed by a finite composition of sorting operations and $O(1)$-FCMs (over an efficiently decidable background structure) can be evaluated by a deterministic algorithm the performs $O(n \log n)$ steps (where n denotes the size of the input database) — and reasonable conjectures in parameterized complexity (cf., [18]) imply that there are Boolean relational algebra queries that cannot be evaluated in time $O(n \log n)$ (with respect to data complexity).

5 Mpms-Automata

In XML databases, an *index* over an XML document typically consists of a number of *streams*, one for each label that occurs in the document. We write $T_{\mathtt{a}}$ to denote the stream associated with label `a`. Each stream $T_{\mathtt{a}}$ consists of *positional encodings* of all elements, in document order, that occur in the document and that carry the label `a`.

A widely used encoding scheme is the *BEL encoding* (cf., e.g., [8, 36]), in which each element is encoded as a triple (*Begin*, *End*, *Level*), where *Begin* and *End* denote the positions of the starting tag and ending tag of the element in the document, and *Level* denotes the level at which the corresponding node occurs in the associated XML document tree. An example of an XML document, the BEL encoding of its elements, and the corresponding index streams is given in Figure 5.

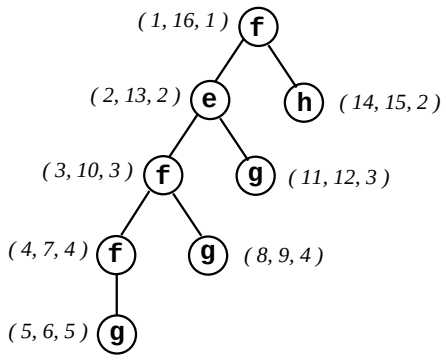
Most currently known algorithms for processing so-called *twig join queries* (i.e., *XPath* queries of a particular kind) can be viewed as implementations of the following computation model (cf., e.g., [8, 36]). Let us consider a fixed query Q — for example, the query `//e/g` asking for all `g`-labeled nodes whose parent carries label `e`. The input to the evaluation algorithm is an indexed XML document, i.e., a collection of streams $T_{\mathtt{a}}$ of positional encodings, for each label `a` occurring in the document. For each occurrence of each label `a` in the *query*, the algorithm may use a *cursor* (or, *head*), with which the stream $T_{\mathtt{a}}$ can be processed once from left to right. An algorithm can move heads asynchronously, and it can read from a head position many times, until it decides to advance it to the next position. The output is written to a write-only output stream.

In [36], Shalem and Bar-Yossef gave a precise characterization of the memory requirement for evaluating twig queries with this computation model. Among other results, they showed the following.

XML doc.:

```
1:  <f>
2:   <e>
3:    <f>
4:     <f>
5:      <g>
6:      </g>
7:     </f>
8:     <g>
9:     </g>
10:   </f>
11:   <g>
12:   </g>
13:  </e>
14:  <h>
15:  </h>
16: </f>
```

XML tree with BEL encoding:



Corresponding index streams:

$T_{\mathtt{e}}$: (2, 13, 2)	$T_{\mathtt{f}}$: (1, 16, 1), (3, 10, 3), (4, 7, 4)
$T_{\mathtt{h}}$: (14, 15, 2)	$T_{\mathtt{g}}$: (5, 6, 5), (8, 9, 4), (11, 12, 3)

Figure 5: XML document, tree with BEL encoding, and corresponding index streams.

Theorem 5.1 ([36]) *Let* `e` *and* `g` *be two distinct labels, and let $\mathcal{A}$ be an algorithm in the above computation model that answers the question "Is there an* `e`*-labeled node that has a* `g`*-labeled child" on any indexed XML input document. Then, for every $n \geqslant 1$ there is an XML document D of depth n on which $\mathcal{A}$ uses at least $n - O(\log n)$ bits of memory.*

For reasoning about the power of the above computation model, the notion of *mpms-automata*[2], introduced in [34], is useful. An illustration of the model is given in Figure 6. The formal definition of the model is as follows.

Let $\mathbb{D}$ be a (potentially infinite) set of data items, let t, m, k_f, k_b be integers with $t, m \geqslant 1$ and $k_f, k_b \geqslant 0$. An

mpms-automaton $\mathcal{A}$ with parameters $(t, \mathbb{D}, m, k_f, k_b)$

receives as input t streams $S_1, \ldots, S_t$ of elements in $\mathbb{D}$ (formally, we can view each stream S_i as a finite string over alphabet $\mathbb{D}$, i.e., as an element in $\mathbb{D}^*$).

The automaton's memory consists of m different states (note that this corresponds to a memory buffer consisting of $\log m$ bits). The automaton's state space is denoted by Q. We assume that Q contains a designated *start state* and that there is a designated subset F of Q of *accepting states*. On each of the input streams, the automaton has k_f heads that process the stream from left to right (so-called *forward heads*) and k_b heads that process the stream from right to left (so-called *backward heads*). The heads are allowed to move asynchronously. We use k to denote the total number of heads, i.e., $k = tk_f + tk_b$. In the *initial configuration* of $\mathcal{A}$ on input $S_1, \ldots, S_t$ the automaton is in the start state, all forward heads are

[2] *mpms* is short for multi-pass processing of multiple streams

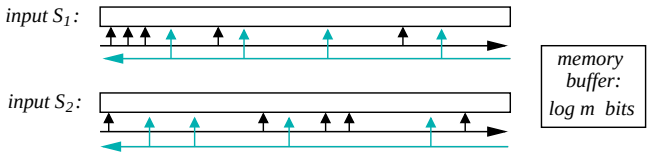


Figure 6: An mpms-automaton with $t = 2$ input streams, $k_f = 5$ forward heads, and $k_b = 4$ backward heads.

placed on the leftmost element, and all backward heads are placed on the rightmost element in the corresponding stream.

During each computation step, depending on (a) the current state (i.e., the current content of the automaton's memory) and (b) the elements of $S_1, \ldots, S_t$ at the current head positions, a transition function determines (1) the next state (i.e., the new content of the automaton's memory), (2) which of the k heads is advanced to the next position (where forward heads are advanced one step to the right, and backward heads are advanced one step to the left), and which (sequence of) elements of $\mathbb{D}$ is written to the output stream.

The automaton's computation on input $S_1, \ldots, S_t$ ends as soon as each head has passed the entire stream. The input is *accepted* if the automaton's state then belongs to the set F of accepting states, and it is *rejected* otherwise.

Consider, e.g., the variant $Disj_n$ of the *set disjointness* problem with input domain $\mathbb{D}_n := \{a_i, b_i, c_i : i \in \{1, \ldots, n\}\}$, where the input consists of two streams $S_1 = s_1 s_2 \cdots s_n$ and $S_2 = t_1 t_2 \cdots t_n$. The goal is to decide whether the sets $\{s_1, \ldots, s_n\}$ and $\{t_1, \ldots, t_n\}$ are disjoint. It is not difficult to see that the problem $Disj_n$ can be solved by an mpms-automaton with parameters $(2, \mathbb{D}_n, n+2, \sqrt{n}, 0)$ (cf., [34]). An according *lower bound* is given by the following theorem.

Theorem 5.2 ([34]) *For all n, m, k_f, k_b such that, for $k = 2k_f + 2k_b$ and $v = (k_f^2 + k_b^2 + 1) \cdot (2k_f k_b + 1)$,*

$k^2 \cdot v \cdot \log(n+1) + k \cdot v \cdot \log m + v \cdot (1 + \log v) \leqslant n$,

the problem $Disj_n$ cannot be solved by any mpms-automaton with parameters $(2, \mathbb{D}_n, m, k_f, k_b)$.

In [36], the variant $RDisj_n$ of the problem $Disj_n$ was considered, where inputs are restricted to streams $S_1 = s_1 \cdots s_n$ and $S_2 = t_1 \cdots t_n$ where, for each $i \in \{1, \ldots, n\}$, $s_i \in \{a_i, b_i\}$ and $t_{n-i+1} \in \{a_i, c_i\}$. Theorem 5.1 can then be proved by (1) using the lower bound for $RDisj_n$ stated in the next theorem and (2) reducing the $RDisj_n$ problem to the problem of answering the query stated in Theorem 5.1.

Theorem 5.3 (implicit in [36]) *The problem $RDisj_n$ cannot be solved by any mpms-automaton with parameters $(2, \mathbb{D}_n, m, 1, 0)$ for $m < \frac{2^n}{2n}$.*

Suggestions for further reading: Details on the computation model for processing indexed XML documents can be found in [8, 36] and the references given therein. The formal definition of the model of mpms-automata is given in [34].

An important future task: Consider also randomized versions of mpms-automata, design efficient randomized approximation algorithms for particular problems, and develop techniques for proving lower bounds in the randomized model.

6 Stream Processing

The *data stream* scenario considers data that is not stored but, instead, has to be processed on-the-fly by using only a limited amount of memory. Thus, the basic data stream model can be viewed as the restriction of the computational model of read/write streams where a single forward scan can be performed on a single r/w stream (cf., Section 3.1). When designing data stream algorithms, one aims at algorithms whose memory size is far smaller than the size of the input.

Typical application areas for which data stream (or, *one-pass*) processing is relevant are, e.g., IP network traffic analysis, mining text message streams, or processing meteorological data generated by sensor networks. Data stream algorithms are also used to support query optimization in relational database systems. In fact, virtually all query optimization methods in relational database systems rely on information about the number of distinct values of an attribute or the self-join size of a relation — and these pieces of information have to be maintained while the database is updated. Algorithms for accomplishing this task have been introduced in the seminal paper [4].

Lower bounds on the size of memory needed for solving a problem by a one-pass algorithm are usually obtained by applying methods from *communication complexity* (cf., Section 3.1). In fact, for many concrete problems it is known that the memory needed for solving the problem by a deterministic one-pass algorithm is at least linear in the size N of the input. For some of these problems, however, *randomized* one-pass algorithms can still compute good *approximate* answers while using memory of size sublinear in N. Typically, such algorithms are based on *sampling*, i.e., only a "representative" portion of the data is taken into account, and *random projections*, i.e., only a rough "sketch" of the data is stored in memory. An example of such an algorithm is given in the proof of Theorem 3.3.

It should be noted that many of the sophisticated data stream algorithms achieve a surprisingly good performance (cf., [28]). For example, it is not at all obvious how to maintain information on the number of *distinct* elements that occurred in a stream, without storing a list of all those elements (since, intuitively, for each element that arrives, one has to check whether this is a *new* ele-

ment or just the repetition of an element that had already occurred in the stream). Here, randomized algorithms are known which give good approximate solutions to this problem while using just a logarithmic number of bits [4].

Suggestions for further reading. In recent years, the database community has addressed the issue of designing general-purpose *data stream management systems* and query languages that are suitable for new application areas where massive amounts of transient data have to be processed. To get an overview of this research area, [5] is a good starting point. Foundations for a theory of stream queries have been laid in [23]. A comprehensive overview of efficient algorithms for data stream processing can be found in [28]. In the context of XML query processing and validation, stream-based approaches have been examined in detail (cf., e.g., [35, 17, 9, 30]).

7 MapReduce & the Mud Model

Implementations of the *MapReduce* programming model, e.g., by Google [11] or the Apache Hadoop project [24], are comonnly used frameworks for distributed processing of large datasets. Users give a high-level specification of an algorithm in terms of a *map* and a *reduce* function. The underlying system then automatically parallelizes the computation across large-scale clusters of machines, handles failures, and schedules inter-machine communication to make efficient use of the network and the disks.

In this programming model, the input consists of a set of (key, value) pairs. The *map* function takes an input pair and produces a set of intermediate (key, value) pairs. The system then automatically groups together all intermediate values with the same intermediate key k and passes them to the *reduce* function. The *reduce* function merges (or, *aggregates*) these values together to form a smaller set of values for that key (typically, just a single value is produced for each key k).

In [13], Feldman et al. introduced an abstract algorithmic model for massive, unordered, distributed (mud, for short) computation, as implemented by systems for MapReduce. For the special case that just a single key is available, a *mud algorithm* is a triple $A = (\Phi, \oplus, \eta)$ with the following parameters:

- $\Phi : \mathbb{D} \to Q$, where $\mathbb{D}$ is the domain of potential input data items, and Q is the set of intermediate values (or, *messages*),
- $\oplus : Q \times Q \to Q$ maps two messages to a single message,
- $\eta : Q \to \mathbb{D}$ is the so-called *post-processing* operator that produces the final output.

When the algorithm is executed on an input $X = x_1, \ldots, x_n$ with $x_i \in \mathbb{D}$, the function Φ produces the sequence of messages $Y = y_1, \ldots, y_n = \Phi(x_1), \ldots, \Phi(x_n)$. The operator $\oplus$ is used to aggregate these messages into a single message. Note that the output can depend on the order in which $\oplus$ is applied. Formally, for any binary tree $\mathcal{T}$ with n leaves, and for any permutation π of $\{1, \ldots, n\}$, let $m_{\mathcal{T},\pi}(X)$ denote the message $q \in Q$ that results from applying $\oplus$ along the topology of $\mathcal{T}$ with the sequence $y_{\pi(1)}, \ldots, y_{\pi(n)}$ used as input at the leaves of $\mathcal{T}$. The overall output of the mud algorithm then is the data item $A(X) := \eta(m_{\mathcal{T},\pi}(X))$. It should be emphasized that $\mathcal{T}$ and π are *not* part of the algorithm, but rather, the algorithm designer needs to make sure that $\eta(m_{\mathcal{T},\pi}(X))$ is independent of the particular choice of $\mathcal{T}$ and π. This is required to ensure that mud algorithms serve as an abstract model of *distributed* computations that are independent of the underlying implementation. A sufficient condition that guarantees independence of $\mathcal{T}$ and π is associativity and commutativity of $\oplus$.

We say that a function $f : \mathbb{D}^n \to \mathbb{D}$ is *computed* by a mud algorithm A if $f(X) = A(X)$ for all $X \in \mathbb{D}^n$. Obviously, mud algorithms can only compute *symmetric* functions, i.e., functions f where $f(x_1, \ldots, x_n) = f(x_{\pi(1)}, \ldots, x_{\pi(n)})$ for any permutation π.

The *communication complexity* of a mud algorithm is defined as $\log |Q|$, i.e., the number of bits needed to represent a message. The *space* (resp., *time*) *complexity* of a mud algorithm is defined as the maximum space (resp., time) complexity of its component functions Φ, $\oplus$, η. Any mud algorithm can be simulated by a data stream algorithm in a straightforward way. A main technical result of [13] shows that, in some sense, also the reverse is true:

Theorem 7.1 ([13]) *Any deterministic streaming algorithm that computes a symmetric function $f : \mathbb{D}^n \to \mathbb{D}$ can be simulated by a mud algorithm with the same communication complexity, and the square of its space complexity.*

Suggestions for further reading: The mud model and some extensions were introduced in [13]. An overview of MapReduce can be found in [11, 24].

An important future task: The *time* complexity of the simulation provided by Theorem 7.1 is superpolynomial, and thus the simulation does not immediately provide distributed algorithms of high performance. It would be interesting to develop more time-efficient simulations.

Acknowledgments. I thank Monika Henzinger and S. Muthu Muthukrishnan for sending me valuable information on the mud model. Furthermore, I thank Katharina Hahn for helpful comments on an earlier version of this paper.

References

[1] J. Abello and J. Vitter, editors. *External Memory Algorithms*, volume 50. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1999.

[2] A. Aggarwal and J. Vitter. The input/output complexity of sorting and related problems. *Communications of the ACM*, 31(9):1116–1127, 1988.

[3] G. Aggarwal, M. Datar, S. Rajagopalan, and M. Ruhl. On the streaming model augmented with a sorting primitive. In *Proc. FOCS'04*, pages 540–549, 2004.

[4] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58:137–147, 1999.

[5] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proc. PODS'02*, pages 1–16, 2002.

[6] P. Beame and D.-T. Huynh-Ngoc. On the value of multiple read/write streams for approximating frequency moments. In *Proc. FOCS'08*, pages 499–508, 2008.

[7] P. Beame, T. Jayram, and A. Rudra. Lower bounds for randomized read/write stream algorithms. In *Proc. STOC'07*, pages 689–698, 2007.

[8] N. Bruno, N. Koudas, and D. Srivastava. Holistic twig joins: Optimal XML pattern matching. In *Proc. SIGMOD'02*, pages 310–321, 2002.

[9] C. Y. Chan, P. Felber, M. Garofalakis, and R. Rastogi. Efficient filtering of XML documents with XPath expressions. *VLDB Journal*, 11(4):354–379, 2002.

[10] J. Chen and C.-K. Yap. Reversal complexity. *SIAM Journal on Computing*, 20(4):622–638, 1991.

[11] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[12] C. Demetrescu, I. Finocchi, and A. Ribichini. Trading off space for passes in graph streaming problems. In *Proc. SODA'06*, pages 714–723, 2006.

[13] J. Feldman, S. Muthukrishnan, A. Sidiropoulos, C. Stein, and Z. Svitkina. On distributing symmetric streaming computations. In *Proc. SODA'08*, pages 710–719, 2008.

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

[15] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *Journal of the ACM*, 48(3):431–498, 2001.

[16] G. Graefe. Query evaluation techniques for large databases. *ACM Computing Surveys*, 25(2):73–170, 1993.

[17] T. Green, A. Gupta, G. Miklau, M. Onizuka, and D. Suciu. Processing XML streams with deterministic automata and stream indexes. *ACM Transactions on Database Systems*, 29(4):752–788, 2004.

[18] M. Grohe. Parameterized complexity for the database theorist. *SIGMOD Record*, 31(4):86–96, 2002.

[19] M. Grohe, Y. Gurevich, D. Leinders, N. Schweikardt, J. Tyszkiewicz, and J. Van den Bussche. Database query processing using finite cursor machines. *Theory of Computing Systems*, 44(4):533–560, 2009.

[20] M. Grohe, A. Hernich, and N. Schweikardt. Lower bounds for processing data with few random accesses to external memory. *Journal of the ACM*, 56(3):1–58, 2009.

[21] M. Grohe, C. Koch, and N. Schweikardt. Tight lower bounds for query processing on streaming and external memory data. *Theor. Comput. Sci.*, 380(1-2):199–217, 2007.

[22] M. Grohe and N. Schweikardt. Lower bounds for sorting with few random accesses to external memory. In *Proc. PODS'05*, pages 238–249, 2005.

[23] Y. Gurevich, D. Leinders, and J. Van den Bussche. A theory of stream queries. In *Proc. DBPL'07*, pages 153–168, 2007.

[24] Hadoop: Open-source software for reliable, scalable, distributed computing. `http://hadoop.apache.org/`.

[25] A. Hernich and N. Schweikardt. Reversal complexity revisited. *Theor. Comput. Sci.*, 401(1-3):191–205, 2008.

[26] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[27] U. Meyer, P. Sanders, and J. Sibeyn, editors. *Algorithms for Memory Hierarchies*. Springer LNCS vol. 2625, 2003.

[28] S. Muthukrishnan. Data Streams: Algorithms and Applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

[29] M. Nodine and J. Vitter. Greed sort: An optimal sorting algorithm for multiple disks. *Journal of the ACM*, 42(4):919–933, 1995.

[30] D. Olteanu. SPEX: Streamed and progressive evaluation of XPath. *IEEE Trans. Knowl. Data Eng.*, 19(7):934–949, 2007.

[31] R. Ramakrishnan and J. Gehrke. *Database Management Systems*. McGraw-Hill, 2002.

[32] M. Ruhl. *Efficient Algorithms for New Computational Models*. PhD thesis, MIT, 2003.

[33] N. Schweikardt. Machine models and lower bounds for query processing. In *Proc. PODS'07*, pages 41–52, 2007.

[34] N. Schweikardt. Lower bounds for multi-pass processing of multiple data streams. In *Proc. STACS'09*, pages 51–61, 2009.

[35] L. Segoufin and C. Sirangelo. Constant-memory validation of streaming XML documents against DTDs. In *Proc. ICDT'07*, pages 299–313, 2007.

[36] M. Shalem and Z. Bar-Yossef. The space complexity of processing XML twig queries over indexed documents. In *Proc. ICDE'08*, pages 824–832, 2008. Full version available at `http://webee.technion.ac.il/people/zivby/`.

[37] L. J. Stockmeyer. *The complexity of decision problems in automata and logic*. PhD thesis, MIT, 1974.

[38] M. Y. Vardi. The complexity of relational query languages. In *Proc. STOC'82*, pages 137–146, 1982.

[39] J. Vitter. External memory algorithms. In *Proc. PODS'98*, pages 119–128, 1998.

[40] J. Vitter. External memory algorithms and data structures: Dealing with massive data. *ACM Computing Surveys*, 33:209–271, 2001.

[41] J. Vitter and E. Shriver. Algorithms for parallel memory I: Two-level memories. *Algorithmica*, 12(2–3):110–147, 1994.

[42] M. Yannakakis. Algorithms for acyclic database schemes. In *Proc. VLDB'81*, pages 82–94, 1981.

XML: Some Papers in a Haystack

Mirella M. Moro
Univ. Fed. Minas Gerais, UFMG
mirella@dcc.ufmg.br

Vanessa Braganholo
Univ. Fed. Rio de Janeiro, UFRJ
braganholo@dcc.ufrj.br

Carina F. Dorneles
Universidade de Passo Fundo, UPF
dorneles@upf.br

Denio Duarte
Univ. do Estado de Santa Catarina
denio@unochapeco.edu.br

Renata Galante
Univ. Fed. Rio Grande Sul, UFRGS
galante@inf.ufrgs.br

Ronaldo S. Mello
Univ. Fed. Santa Catarina, UFSC
ronaldo@inf.ufsc.br

ABSTRACT

XML has been explored by both research and industry communities. More than 5500 papers were published on different aspects of XML. With so many publications, it is hard for someone to decide where to start. Hence, this paper presents some of the research topics on XML, namely: XML on relational databases, query processing, views, data matching, and schema evolution. It then summarizes some (some!) of the most relevant or traditional papers on those subjects.

1. INTRODUCTION

XML is a language for specifying semi or completely structured data. It has been explored over and over by both research and industry communities. More than ten years after its proposal, its initial expectation of "solving all the problems in the world" has not been fulfilled. However, we cannot say XML failed, since it solved some really important problems very efficiently. Some of those problems include data integration, data interoperability, and data publishing on the Web. Moreover, XML is adopted as a standard language by many industries (from retail to healthcare) for exchanging data. For those and other reasons, XML is the most successful, ubiquous technology for the Web, at the same level as URI, HTTP, and HTML [98].

If someone wants to start a research on XML, the first questions that come to mind are "what papers one must read" and "what part of the XML literature one needs". There is one problem though with such questions. Much research has already been done, and a considerable amount of papers have already been published about XML. If a person needs to start any study on XML, s/he will probably start reading the W3C documents, but that is not enough.

Specifically, the DBLP Computer Science Bibliography[1] Faceted Search points out XML as the 10^{th} most popular keyword[2], appearing in 1250 publications. Using the DBLP CompleteSearch with *XML* as search word returns 5529 publications. How does one find the desired publication over so many? One solution is to specify a second search word such as *query*. In this case (search words *XML* and *query*), the CompleteSearch returns 604 publications, which is still a lot.

This paper is intended to people who would like to start a research on XML or simply start studying XML from the research point-of-view. It summarizes **some** of the research topics on XML: XML on relational databases (Section 2), views (Section 3), query processing (Section 4), data matching (Section 5), and schema evolution (Section 6). It then presents some of the most relevant or traditional papers on those subjects. By "relevant or traditional papers", we mean those that are more frequently cited or that considerably changed the way XML research was done.

Note that this paper is not intended to cover the whole vast literature on XML. For example, there are papers about the three major commercial DBMSs (*i.e.* IBM DB2, Microsoft SQL Server, Oracle) that discuss many practical aspects of supporting XML management, for example [12, 62, 81, 82]. Other industrial papers discuss specific topics, such as schema evolution [11]. Furthermore, some research groups and companies have also their own XML engines [14, 42, 43, 54]. We do not plan to detail those solutions. Instead, we focus on a very selected set of papers on some XML research areas. Nevertheless, this paper intends to be a starting point for any person who decided to study this wonderful, versatile, powerful language, called XML.

2. XML ON RELATIONAL DATABASES

The hierarchical nature of XML brought several challenges to the database community. One of the first ones was simple, but very tricky: How to store XML data? Relational databases seemed to be a good alternative, since most of the data (both in academia and industry) was/is stored in RDBMS (Relational Database Management Systems).

The first paper to suggest the use of relational database engines for managing data in files was [1]. The file contents should then be stored in the relational database using some mapping schema. For this to be possible, the file should have a "strict inner structure". This requirement resembles XML documents that have a DTD or schema.

2.1 Storage

Based on such observation, many publications followed with the goal of storing [24, 25, 35, 44, 57, 83] and querying [32, 83, 85, 89] XML documents using relational databases.

XML-based approaches. The first proposals to store generic XML documents (schemaless) appeared in [44]. The *Edge* and *Attribute* approaches are able to store any XML document (regardless of a schema) in relations. Later, a different solution, based on *dynamic intervals*, introduced both storage and query translation algorithms [32].

DTD-based approaches. Then, the pioneer proposals to store XML documents in relations by exploring their

[1] http://www.informatik.uni-trier.de/~ley/db access on 02/23/09
[2] http://dblp.l3s.de/browse.php?browse=mostPopularKeywords access on 02/23/09

.

schema (DTD) are in [83]. It proposes *Basic*, *Shared*, and *Hybrid Inlining* and discusses their pros and cons. However, query translation is not discussed.

Constraint-based approaches. Some approaches construct the relational schema based on constraints of the XML documents. For example, [57] explores DTD constraints such as cardinalities and referential integrity. On the other hand, [24] uses XML key and keyref, while [25] explores XML functional dependencies (XFDs) to generate a relational schema that has as few redundancy as possible.

Discussion. It is very important to notice that querying depends on the method for storing XML data in the relations. The Edge and Attribute approaches require many joins to recompute the document. The dynamic interval uses sub-queries and the *union all* operator to reproduce the original XML instance.

All of those publications have influenced the native and hybrid storage alternatives currently in use.

2.2 Updates

Once the storage problem was handled, it was necessary to update the documents stored in the RDBMS. There are three main proposals for updating XML documents stored in relational databases [88, 89, 91]. All of them assume a default mapping of the XML document to relations, such as the *shared inlining method* [83] and the approach of [57].

Tatarinov et al. [88] propose an extension to XQuery to support updates. This extension comprises primitives such as `UPDATE`, `DELETE`, `INSERT`, `REPLACE`, and `RENAME`. They also discuss how to translate those updates to the underlying RDBMS for updating XML documents using the shared inlining method. In [89], Tatarinov et al. use the XQuery extension proposed in [88] to deal with ordered XML documents. They propose a storage method for ordered XML documents in relations, and show how updates that affect order can be translated. Finally, [91] provides an updatability study for XML documents stored in RDBMS.

It is important to notice that [88] presented the first proposal of an update language for XML. Many of the concerns risen on that paper served as starting points to the study of update-related problems regarding XML [8].

3. XML VIEWS

A natural follow-up to store XML data as relations is to generate XML *from* relational data. The literature deals with this problem by using the concept of views. Many approaches explore building and querying XML views over relational databases [6, 13, 23, 41, 61, 84, 86]. Most of them tackle the problem by building a *default* XML view from the relational source, and then using an XML query language to query the default view [13, 23, 41, 86]. Their concerns are basically the following: *(i)* how to compose queries with view definitions, in order to get a *single* resulting expression that represents both the query and the view; *(ii)* how to use the relational engine to execute the query resulting from step *i*; and *(iii)* how to use the relational tuples resulting from step *ii* to build the resulting XML document.

Each approach has a different technique to construct the XML view using the relational engine to retrieve data. Some transform the XML view definition into extended SQL [23, 84, 86], others use internal representations to map the XML view to several SQL queries [13, 41] or to map the relational schema to an XML Schema [61]. For speeding-up queries, some approaches maintain materialized views [6]. The classical papers on this subject are *SilkRoute* [41] and *XPERANTO* [86]. The SQL language has also been extended to support generating XML documents from relational data. The extensions include SQL/XML, FOR XML, SQLX, and are supported by most database vendors.

Once a view is defined, there are also the problems of updating it, and translating it back to the underlying relational database. The pioneer work is [19], where the solutions is based on the *Nested Relational Algebra* to define the XML views. Later, the idea was extended to support an XQuery-based view definition language [20, 21]. The main idea of those works is to map XML view updates to Relational View updates, and then map the updates back to the database by using existing View Update work for relational databases.

Then, the approach in [92] presents an updatability study of XQuery views published over relational databases. Their results are analogous to the results of [19]. However, neither of those papers discusses how updates are translated to the underlying relational database. Recently, a two step approach to the problem was proposed in [95]: the first step is schema checking, which uses schema information to decide if the update is translatable to the database. In some cases, a second step may be required to make the decision: data checking, which is an expensive operation, and thus must be avoided whenever possible.

4. XML QUERY PROCESSING

The success of XML databases depends heavily on efficient methods for manipulating XML data. XML requires efficient query processing techniques over tree-structured data. Given the many differences between that type of data and relational data, the well known and efficient algorithms from relational databases cannot be (directly) employed to process XML data efficiently.

There are many algorithms for XML query processing, which can be divided in two categories: *(i)* the ones that employ the infra-structure of a relational database (*i.e.*, based on the mapping from XML to relational tables/columns) [32, 83, 85, 89, 90], and *(ii)* the ones that employ a native XML engine (*i.e.*, data is handled in its original tree format; access methods consider the native tree structure of the data; new, XML-structure-aware indexes may be employed) [3, 22, 27, 28, 55, 60, 70, 79, 94, 99, 102]. This section focuses on XML query processing using a *native* engine. Moreover, there are different semantics for XML queries. Here, we consider the basic structural constraints: structural join and tree-pattern query (or twig). Other semantics, such as keyword search and document filtering, are not covered due to space constraints. Finally, we present some other very specific topics related to query processing.

Structural Joins. This type of join is the simplest possible structure and considers only ancestor//descendant and parent/child pairs. The query result may be the pair or one of its components. The first native approach to implement structural joins is the *MPMGJN* algorithm [102]. Then, *StackTree* [3] is the state-of-the-art algorithm for structural joins, since it defined the problem and presented a very nice initial solution. Different nuances of structural joins use partitioning techniques, which include the work on [60], first partition-based technique for structural joins. Finally, there is also some work on processing structural joins when the document suffers insertions [31].

Tree-Pattern Query, or Twig. This type of query considers path and subtree structures, *i.e.*, combinations of the basic axis (ancestor//descendant and parent/child) with predicates. The query result may be an element, a path, or a subtree of the document that satisfies the constraints. The XML twig algorithms may be divided into four very distinct categories (see [70] for definition of the categories and an experimental comparison). Note that those categories do not consider optimizations according to the query workload. The only optimizations allowed are those based on the data, and the data alone (*i.e.*, clustering, indexes, and so on).

The simplest (possibly the most naïve) way to process twigs is by merging the results of structural joins. Both *StackTree* [3] and *MPMGJN* [102] (from structural joins) are binary join algorithms, *i.e.*, they join only a pair of inverted lists (or only one edge in the query twig). Since a complete twig query consists of a series of binary joins, the problem of join order selection has to be considered seriously.

A dynamic programming method to select an optimal or sub-optimal order of binary structural joins for XML twig queries is proposed in [99]. The *StackTree* binary join algorithm and the corresponding dynamic-programming-based join order selection algorithm have been integrated into Timber [54] (the native XML database prototype from the University of Michigan). Other techniques followed those by aggregating indexes to the lists to be merged, *e.g.*, [27, 55].

The state-of-the art algorithm for processing twigs is a holistic approach, the *TwigStack* [22]. It performs a pipelining join by joining multiple inverted lists at one time without generating intermediate results. That paper also introduces *XB-Twigstack*, a *TwigStack* with indexes.

Other algorithms followed by proposing optimizations for different nuances (restrictions, requirements, and profiles), specialized numbering schemes such as Dewey (*e.g.*, [63]), or using different holistics, such as subsequence matching (*e.g.*, [79, 94]). Other techniques consider the query workload, such as *APEX* [28], which defines an index and considers the query workload in order to compact even more the index.

A third way to process twigs is through a finite state machine (FSM). Each XML element is processed only once through sequential access. For example, an extension to FSM with stacks (similar to *TwigStack*) is proposed on [70].

Other XML Query-related Topics. It is very important to notice that it is impossible to cover 604 papers on XML query processing. Besides the work aforementioned, we could also cite other punctual papers, which deal with very specific, query-related topics. Examples include: index advisors [39], unordered XQuery processing [48], query unnesting [65], evolving queries [69], structural summaries [71], among many others.

5. XML DATA MATCHING

The central problem on data matching is to find out heterogeneous data that represent the same real world object. Solutions to this matter are required by several applications, like data integration, database design, and querying heterogeneous data sources [37]. XML data matching is a complex task, and involves at least two levels: schema and instance.

5.1 Schema Matching

Schema matching is an active research area since the Eighties, but with more theoretical than effective proposals until today. Two approaches are usually considered for schema matching [40]: *(i)* to define a global schema that maintains mappings to a set of local schemas; or *(ii)* to define, for each pair of schemas, a set of mappings among their components. The first one is classified into two main categories [59]: *(i.1) Global-as-View* (*GAV*), and *(i.2) Local-as-View* (*LAV*). A *GAV* approach builds a global schema from the semantic matching of the local schemas, *i.e.*, the global schema is a unified view of the local schemas. The *LAV* approach builds a global schema from scratch, and then defines a set of mappings to the local schemas, *i.e.*, the local schemas are views of the global schema. Despite of being initially proposed for databases, these approaches have been applied to XML schema matching, with adaptations that consider the semistructured nature of XML data. Nonetheless, this is still an active research area, since most of the existing work do not deliver satisfactory results [45].

Initial work on XML schema matching focused on defining a global schema for XML data. This happened because XML data sources are usually on the *Web* and have no associated schemas. Therefore, *mediator-based* architectures have been developed [97]. There are two layers in this kind of architecture: *wrapping* and *mediation*. The wrapping layer is responsible for extracting a schema for each XML source, or providing mappings from a predefined global schema to the XML data sources. The mediaton layer is responsible for managing a global schema according to a GAV [67, 73, 75, 80, 101] or LAV [34, 36] approach. For example, in the *Enosys* data integration platform [73], the mediator module allows the user to define XML views from the global schema. A drawback on some architectures is that the matching process is manual, *i.e.*, the mediators act as a tool that helps the user to define global schema or mappings among XML local sources [29, 66, 73, 80]. This limitation has been solved by semi-automatic approaches [33, 34, 36, 67, 75, 101].

Another idea is to adopt a *canonical model*, *i.e.*, a data model on which the XML global schema is defined. Some proprosals consider the proper XML model to define the canonical schema, which is useful for applications that need to persist and/or to manage unified data in XML format [34, 36, 66, 73, 101]. Others define the global schema based on a conceptual model, which is more suitable to capture data semantics, and useful for applications that query heterogeneous XML data sources [33, 64, 67, 75, 80].

Regarding the XML schema matching process, differences among the proposals are mainly on their XML model constructs. For example, most approaches do not deal with the matching of mixed elements or an element content model specified as a choice among several nested elements [29, 34, 36, 64, 66, 75, 80]. Others try to overcome such limitation, *e.g. BInXS* [67]. Regarding the comparison strategy for defining schema components to be matched, there are usually dictionaries or *Thesauri* (*linguistic* strategy) [34, 64, 67], and/or *metrics* (functions and/or algorithms) applied on structural and/or constraint properties of the XML schema, like data types and cardinality constraints [64, 67].

Few approaches define binary schema mappings among XML local schemas instead of creating a global schema, *e.g. HDM* [66], *Xere* [33], and *YAT* system [29]. In the first two, XML schema of local sources are converted to schemas in an intermediate conceptual model. Semantic mappings are further defined between conceptual and XML schemas. *YAT* supports the manual definition of mappings between XML schemas represented through a graph-based model.

These proposals illustrate that schema mapping approaches for XML can be provided at a conceptual or logical level.

Different from those traditional schema matching approaches, other systems consider information about XML instances and machine learning techniques to deduce semantic correspondences between heterogeneous schemas' elements [36]. Despite of the additional analysis of XML data contents (besides schema information, which can contribute to more exact schemas component matchings) approaches like that are criticized by their complexity.

5.2 Instance Matching

Due to the hierarchical nature of XML data, most approaches on instance matching consider both structure and content aspects. The techniques are very different from each other. Some employ *diff* algorithm [30, 93] or a TF/IDF weighting scheme [76], while others use a tree-edit distance-based metric [4, 51, 52, 58], or a similarity function [96].

Regarding diff algorithms, the *XyDiff* algorithm [30] detects changes in XML documents that are stored in a data warehouse, and uses signatures to match (large) subtrees that were left unchanged between the old and the new versions. Then, it propagates those matches to ancestors and descendants to obtain more matches. It also uses XML specific information such as ID attributes. Another diff algorithm is the *XDiff* algorithm [93], which integrates key XML structure characteristics with standard tree-to-tree correction techniques. Such algorithms focus on identifying changes in versions of a document and introduce cost models to quantify those changes.

Likewise, the tree-edit distance metric is widely used to match XML documents [4, 51, 52, 58]. Specifically, in [51, 52], the approximate matching is quantified in terms of distance notions. The central idea is to incorporate the tree-edit distance in a join framework, which is used in XML integration processes. In [4], the authors propose the concept of approximate tree join, which intuitively assesses that two objects are the same if they have (almost) the same tree. They propose the *pq-grams* distance to approximately match hierarchical information. Such measure is defined as the distance between ordered labeled trees using the tree-edit distance and similarity functions. In [100], the authors propose to transform tree-structured data into an approximate numerical multidimensional vector, which encodes the original structure information. The algorithm embeds the proposed distance into a filter-and-refine framework to process similarity search on tree-structured data.

Some work consider both content and structure during the instance matching [74, 77]. For example, in [77], the authors extend the classical approach to duplicate detection in flat relational data, that is the sorted neighborhood method to be applied in nested XML elements, detecting duplicates at each level of the XML hierarchy. In addition, the framework proposed in [96] characterizes duplicates in a matching process by considering the description of data instances, and a similarity score using a similarity measure.

The choice of an appropriated similarity function is an important task in a matching process, since it defines the result quality. A problem related to the use of disparate similarity functions appears when the similarities scores of the XML nodes are combined. As the individual functions usually generate scores that are not comparable, there is no general straightforward way for combining independent distinct functions into a single measure. Other recent work has addressed such a problem [38, 53].

6. XML SCHEMA EVOLUTION

Schema evolution deals with updating a schema when it no longer meets the user or the application requirements. The main problem is how to allow schemas to change while maintaining the access to the existing XML data.

The approach employed to allow schemas to evolve depends on the XML schema language. The two most used schema languages are DTD and XML Schema (XSD) [10, 68]. However, XSD is more powerful than DTD because XSD, unlike DTD, permits decoupling an element tag from its type (or content model), and an element may have different types depending on the context [72, 78, 56].

The structure employed to represent XML documents and schemas is important to implement an *evolution framework*. XML documents are mostly represented as unranked labeled trees (*i.e.*, their nodes have no priory bound on the number of children). Each tree node represents an element (or an attribute). XML schemas, on the other hand, do not have a standard representation (as XML documents have). Thus, the representations found in the literature are: regular tree grammars (RTG) [17, 26, 72], direct graphs (DG) [2, 87], and mixed representation (*e.g.*, based on applying more than one structure to model a schema) [50, 78]. Although XML schemas may have different representations, the primitives for evolving them follow, as expected, the same basis: objects may be inserted, deleted and/or updated.

Update Primitives. Update primitives for XML schemas are slightly different from those to update schemas on traditional databases (*e.g.*, relational databases). XML schema objects may still be inserted, deleted, and/or replaced. However, XML schemas have their own characteristics: *(i)* objects have cardinality, for example, an element may repeat n times ($n \geq 0$ or $n \geq 1$); *(ii)* objects may be moved, for example, a sub-tree may be moved from a parent node to another one; and, *(iii)* objects may be complex and may have their complex type changed.

Non-conservative Primitives. Non-conservative schema updates define a set of operations that, when applied to a schema, may lead to an inconsistent state. In this case, revalidation and adaptation processes are necessary. The revalidation of an XML document identifies whether an initially valid document is still valid with respect to the updated schema definitions (*e.g.*, [5, 7, 15]). The adaptation of an XML document is required if the revalidation process fails. This process operates on the document structure so it respects the most up to date schema definitions. Initial approaches were based on object-oriented databases [2, 87] and schema versioning [46]. Using information from the DTD, schema evolution may be triggered by patterns detected on documents through usage of data mining techniques [9]. *MXML* [47] is a multidimensional model for representing the history of XML documents and the evolution of their schema. Finally, [49] describes a document evolution and adaptation proposal, and [50] extends it for the incremental validation of documents upon schema evolution.

Conservative Primitives. This kind of primitives is important in XML-based applications since: *(i)* it is not necessary to revalidate all XML document collections (or database); *(ii)* data loss can be avoided because it may be necessary to delete tags (and their information) from

initially valid documents; and *(iii)* when documents to be revalidated are stored in different sites, not only their transfer cost should be considered (in addition to the whole revalidation cost) but also problems due to the access control should be faced. However, conservative primitives are not complete, that is, some update primitives cannot be mapped into conservative ones. Several approaches have been proposed to evolve XML schemas. Some of them keep the instances validity without revalidation [16, 18] (*e.g.*, all existing valid instances are guaranteed to be valid with relation to updated schema), while others have to test, in some cases, the validity of all existing valid instances [2, 49, 87].

7. CONCLUDING REMARKS

XML has been explored over and over by both research and industry communities. This paper summarized some of the research topics on XML by presenting some of the most relevant/traditional papers on those subjects. It was not intended to cover the vast XML literature, but to point out some directions to those who are interested in learning this powerful, versatile language called XML. The topics addressed on this paper were not randomly chosen. They reflect the area of expertise of the authors. Overtime, we plan to expand it to include more XML research areas.

Acknowledgements. The authors thank the reviewers for their feedback. This work was partially supported by CNPq, CAPES, FAPERJ, and FAPEMIG, Brazil.

8. REFERENCES

[1] S. Abiteboul, S. Cluet, and T. Milo. Querying and Updating the File. In *VLDB*, 1993.

[2] L. Al-Jadir and F. El-Moukaddem. Once Upon a Time a DTD Evolved into Another DTD... In *OOIS*, 2003.

[3] S. Al-Khalifa et. al. Structural Joins: A Primitive for Efficient XML Query Pattern Matching. In *ICDE*, 2002.

[4] N. Augsten, M. Böhlen, and J. Gamper. Approximate matching of hierarchical data using pq-grams. In *VLDB*, 2005.

[5] A. Balmin, Y. Papakonstantinou, and V. Vianu. Incremental Validation of XML Documents. *ACM TODS*, 29(4):710–751, 2004.

[6] A. Balmin et. al. A Framework for Using Materialized XPath Views in XML Query Processing. In *VLDB*, 2004.

[7] D. Barbosa et. al. Efficient Incremental Validation of XML Documents. In *ICDE*, 2004.

[8] M. Benedikt et. al. Adding Updates to XQuery: Semantics, Optimization, and Static Analysis. In *XIME-P*, 2005.

[9] E. Bertino et. al. Evolving a Set of DTDs According to a Dynamic Set of XML Documents. In *EDBT*, 2002.

[10] G. J. Bex, F. Neven, and J. V. Bussche. DTDs versus XML schema: a practical study. In *WebDB*, 2004.

[11] K. S. Beyer et. al. DB2/XML: Designing for Evolution. In *SIGMOD*, 2005.

[12] K. S. Beyer et. al. DB2 Goes Hybrid: Integrating Native XML and XQuery with Relational Data and SQL. *IBM Systems Journal*, 45(2):271–298, 2006.

[13] P. Bohannon et. al. Optimizing View Queries in ROLEX to Support Navigable Result Trees. In *VLDB*, 2002.

[14] P. Boncz et. al. MonetDB/XQuery: a fast xquery processor powered by a relational engine. In *SIGMOD*, 2006.

[15] B. Bouchou and M. H. F. Alves. Updates and Incremental Validation of XML Documents. In *DBPL*, 2003.

[16] B. Bouchou and D. Duarte. Assisting XML Schema Evolution that Preserves Validity. In *Brazilian Symp. on Data Base (SBBD)*, 2007.

[17] B. Bouchou et al. Extending Tree Automata to Model XML Validation under Element and Attribute Constraints. In *ICEIS*, 2003.

[18] B. Bouchou et al. Schema Evolution for XML: A Consistency-Preserving Approach. In *Mathematical Foundations of Computer Science*, 2004.

[19] V. Braganholo, S. B. Davidson, and C. A. Heuser. On the Updatability of XML Views over Relational Databases. In *WebDB*, 2003.

[20] V. Braganholo, S. B. Davidson, and C. A. Heuser. From XML View Updates to Relational View Updates: Old Solutions to a New Problem. In *VLDB*, 2004.

[21] V. Braganholo, S. B. Davidson, and C. A. Heuser. PATAXÓ: a Framework to Allow Updates Through XML Views. *ACM TODS*, 31(3):839–886, 2006.

[22] N. Bruno, N. Koudas, and D. Srivastava. Holistic Twig Joins: Optimal XML Pattern Matching. In *SIGMOD*, 2002.

[23] S. Chaudhuri, R. Kaushik, and J. Naughton. On Relational Support for XML Publishing: Beyond Sorting and Tagging. In *SIGMOD*, 2003.

[24] Y. Chen, S. B. Davidson, and Y. Zheng. Constraint Preserving XML storage in Relations. In *WebDB*, 2002.

[25] Y. Chen, S. B. Davidson, and Y. Zheng. RRXS: redundancy reducing XML storage in relations. In *VLDB*, 2003.

[26] B. Chidlovskii. Using Regular Tree Automata as XML Schemas. In *Proc. IEEE Advances in Digital Libraries Conference*, May 2000.

[27] S.-Y. Chien et. al. Efficient Structural Joins on Indexed XML Documents. In *VLDB*, 2002.

[28] C.-W. Chung, J.-K. Min, and K. Shim. APEX: an adaptive path index for XML data. In *SIGMOD*, 2002.

[29] S. Cluet et. al. Your Mediators Need Data Conversion! In *SIGMOD*, 1998.

[30] G. Cobéna, S. Abiteboul, and A. Marian. Detecting Changes in XML Documents. In *ICDE*, 2002.

[31] E. Cohen, H. Kaplan, and T. Milo. Labeling Dynamic XML Trees. In *PODS*, 2002.

[32] D. DeHaan et.al. A comprehensive XQuery to SQL translation using dynamic interval encoding. In *SIGMOD*, 2003.

[33] G. Della Penna et. al. Interoperability Mapping from XML Schemas to ER Diagrams. *Elsevier DKE*, 59:166–188, 2006.

[34] C. Delobel et. al. Semantic Integration in Xyleme: a Uniform Tree-based Approach. *Elsevier DKE*, 44(3):267–298, 2003.

[35] A. Deutsch, M. Fernandez, and D. Suciu. Storing Semistructured Data with STORED. In *SIGMOD*, 1999.

[36] A. Doan, P. Domingos, and A. Halevy. Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach. In *SIGMOD*, 2001.

[37] A. Doan and A. Y. Halevy. Semantic Integration Research in the Database Community: A Brief Survey. *AI Magazine*, 26(1):83–94, 2005.

[38] C. F. Dorneles et. al. A Strategy for Allowing Meaningful and Comparable Scores in Approximate Matching. In *CIKM*, 2007.

[39] I. Elghandour et al. An XML Index Advisor for DB2. In *SIGMOD*, 2008.

[40] A. Elmagarmid, M. Rusinkiewicz, and A. Sheth. *Management of Heterogeneous and Autonomous Database Systems*. Morgan Kaufmann, San Francisco, CA, 1999.

[41] M. Fernandez et. al. SilkRoute: A framework for publishing relational data in XML. *ACM TODS*, 27(4):438–493, 2002.

[42] T. Fiebig and H. Schöning. Software AG's Tamino XQuery Processor. In *XIME-P*, 2004.

[43] T. Fiebig et. al. Natix: A Technology Overview. In *NODe - Web and Database-Related Workshops*, 2002.

[44] D. Florescu and D. Kossmann. A performance evaluation of alternative mapping schemes for storing XML data in a re- lational database. *Tech.Report 3684*, INRIA, 1999.
[45] A. Gal. The Generation Y of XML Schema Matching Panel Description. In *XSym*, 2007.
[46] R. M. Galante et. al. Temporal and versioning model for schema evolution in object-oriented databases. *Data Knowl. Eng.*, 53(2):99–128, 2005.
[47] M. Gergatsoulis and Y. Stavrakas. Representing Changes in XML Documents using Dimensions. In *Xsym*, 2003.
[48] T. Grust, J. Rittinger, and J. Teubner. eXrQuy: Order Indifference in XQuery. In *ICDE*, 2007.
[49] G. Guerrini, M. Mesiti, and D. Rossi. Impact of XML schema evolution on valid documents. In *WIDM*, 2005.
[50] G. Guerrini, M. Mesiti, and D. Sorrenti. XML schema evolution: Incremental validation and efficient document adaptation. In *XSym*, 2007.
[51] S. Guha et. al. Approximate XML joins. In *SIGMOD*, 2002.
[52] S. Guha et. al. Integrating XML Data Sources using Approximate Joins. *ACM TODS*, 31(1):161–207, 2006.
[53] S. Guha et.al. Merging the Results of Approximate Match Operations. In *VLDB*, 2004.
[54] H. V. Jagadish et.al. TIMBER: A native XML database. *The VLDB Journal*, 11(4):274–291, 2002.
[55] H. Jiang et. al. XR-Tree: Indexing XML data for efficient structural joins. In *ICDE*, 2003.
[56] A. H. F. Laender et. al. An X-ray on Web-available XML Schemas. *SIGMOD Record*, 38(1):37–42, 2009.
[57] D. Lee and W. W. Chu. Constraints-Preserving Transformation from XML Document Type Definition to Relational Schema. In *ER*, 2000.
[58] K.-H. Lee, Y.-C. Choy, and S.-B. Cho. An Efficient Algorithm to Compute Differences between Structured Documents. *IEEE TKDE*, 16(8), 2004.
[59] M. Lenzerini. Data Integration: A Theoretical Perspective. In *PODS*, 2002.
[60] Q. Li and B. Moon. Partition Based Path Join Algorithms for XML Data. In *DEXA*, 2003.
[61] C. Liu, M. W. Vincent, J. Liu, and M. Guo. A Virtual XML Database Engine for Relational Databases. In *XSym*, 2003.
[62] Z. H. Liu, M. Krishnaprasad, and V. Arora. Native Xquery Processing in Oracle XMLDB. In *SIGMOD*, 2005.
[63] J. Lu et al. From Region Encoding to Extended Dewey: on Efficient Processing of XML Twig Pattern Matching. In *VLDB*, 2005.
[64] J. Madhavan, P. Bernstein, and E. Rahm. Generic Schema Matching with Cupid. In *VLDB*, 2001.
[65] N. May, S. Helmer, and G. Moerkotte. Strategies for Query Unnesting in XML Databases. *ACM Trans. Database Syst.*, 31(3):968–1013, 2006.
[66] P. McBrien and A.Poulovassilis. A semantic approach to integrating XML and structured data sources. In *CAISE*, 2001.
[67] R. S. Mello and C. A. Heuser. BInXS: A Process for Integration of XML Schemata. In *CAISE*, 2005.
[68] L. Mignet, D. Barbosa, and P. Veltri. The XML Web: a First Study. In *WWW*, 2003.
[69] M. M. Moro, S. Malaika, and L. Lim. Preserving XML queries during schema evolution. In *WWW*, 2007.
[70] M. M. Moro, Z. Vagena, and V. J. Tsotras. Tree-Pattern Queries on a Lightweight XML Processor. In *VLDB*, 2005.
[71] M. M. Moro, Z. Vagena, and V. J. Tsotras. XML Structural Summaries. *PVLDB*, 1(2):1524–1525, 2008.
[72] M. Murata et al. Taxonomy of XML Schema Language using Formal Language Theory. *ACM TOIT*, 5(4), 2005.
[73] Y. Papakonstantinou and V. Vassalos. Architecture and implementation of an XQuery-based information integration platform. *IEEE Data Eng. Bull.*, 25(1):18–26, 2002.
[74] U. Park and Y. Seo. An Implementation of XML Documents Search System based on Similarity in Structure and Semantics. In *Int Workshop on Challenges in Web Information Retrieval and Integration*, pages 97–103. IEEE Computer Society, 2005.
[75] K. Passi et. al. A Model for XML Schema Integration. In *EC-Web*, 2002.
[76] E. Popovici, G. Ménier, and P.-F. Marteau. SIRIUS XML IR System: Approximate Matching Structure and Textual Content. In *INEX*, 2006.
[77] S. Puhlmann, M. Weis, and F. Naumann. XML duplicate detection using sorted neighborhoods. In *EDBT*, 2006.
[78] M. Raghavachari and O. Shmueli. Efficient Schema-Based Revalidation of XML. In *EDBT*, 2004.
[79] P. Rao and B. Moon. Sequencing XML data and query twigs for fast pattern matching. *ACM TODS*, 31(1):299–345, 2006.
[80] P. Rodriguez-Gianolli and J. Mylopoulos. A Semantic Approach to XML-Based Data Integration. In *ER*, 2001.
[81] M. Rys. XML and relational database management systems: inside microsoft sql server 2005. In *SIGMOD*, 2005.
[82] M. Rys, D. D. Chamberlin, and D. Florescu. XML and Relational Database Management Systems: the Inside Story. In *SIGMOD*, 2005.
[83] J. Shanmugasundaram et. al. Relational Databases for Querying XML Documents: Limitations and Opportunities. In *VLDB*, 1999.
[84] J. Shanmugasundaram et. al. Efficiently Publishing Relational Data as XML Documents. *The VLDB Journal*, pages 65–76, 2000.
[85] J. Shanmugasundaram et. al. A general technique for querying XML documents using a relational database system. *SIGMOD Record*, 30(3):20–26, Sept. 2001.
[86] J. Shanmugasundaram et.al. Querying XML Views of Relational Data. In *VLDB*, 2001.
[87] H. Su et al. XEM: Managing the evolution of XML documents. In *RIDE-DM*, 2001.
[88] I. Tatarinov et. al. Updating XML. In *SIGMOD*, 2001.
[89] I. Tatarinov et. al. Storing and Querying Ordered XML Using a Relational Database System. In *SIGMOD*, 2002.
[90] A. Vyas, M. F. Fernández, and J. Siméon. The Simplest XML Storage Manager Ever. In *XIME-P*, 2004.
[91] L. Wang, M. Mulchandani, and E. A. Rundensteiner. Updating XQuery Views Published over Relational Data: A Round-trip Case Study. In *XSym*, 2003.
[92] L. Wang and E. A. Rundensteiner. On the updatability of XML Views Published over Relational Data. In *ER*, 2004.
[93] Y. Wang, D. DeWitt, and J. Cai. X-Diff: An Effective Change Detection Algorithm for XML Documents. In *ICDE*, 2003.
[94] H. Wang et. al. ViST: a dynamic index method for querying XML data by tree structures. In *SIGMOD*, 2003.
[95] L. Wang et. al. An Optimized Two-Step Solution for Updating XML Views. In *DASFAA*, 2008.
[96] M. Weis and F. Naumann. DogmatiX Tracks down Duplicates in XML. In *SIGMOD*, 2005.
[97] G. Wiederhold. Mediators in the Architecture of Future Information Systems. *Computer*, 25(3):38–49, Mar 1992.
[98] E. Wilde and R. J. Glushko. XML Fever. *Commun. ACM*, 51(7):40–46, 2008.
[99] Y. Wu, J. M. Patel, and H. V. Jagadish. Structural join order selection for XML query optimization. In *ICDE*, 2003.
[100] R. Yang, P. Kalnis, and A. K. H. Tung. Similarity Evaluation on Tree-structured Data. In *SIGMOD*, 2005.
[101] X. Yang, M. L. Lee, and T. W. Ling. Resolving Structural Conflicts in the Integration of XML Schemas: A Semantic Approach. In *ER*, 2003.
[102] C. Zhang et. al. On Supporting Containment Queries in Relational Database Management Systems. *SIGMOD Record*, 30(2):425–436, 2001.

ASSET Queries: A Declarative Alternative to MapReduce

Damianos Chatziantoniou*
Department of Management Science and Technology,
Athens University of Economics and Business (AUEB)
damianos@aueb.gr

Elias Tzortzakakis
Institute of Computer Science
Foundation for Research and Technology (FORTH)
tzortzak@ics.forth.gr

ABSTRACT
Today's complex world requires state-of-the-art data analysis over truly massive data sets. These data sets can be stored persistently in databases or flat files, or can be generated in real-time in a continuous manner. An associated set is a collection of data sets, annotated by the values of a domain D. These data sets are populated using a data source according to a condition θ and the annotated value. An ASsociated SET (ASSET) query consists of repeated, successive, interrelated definitions of associated sets, put together in a column-wise fashion, resembling a spreadsheet document. We present DataMingler, a powerful GUI to express and manage ASSET queries, data sources and aggregate functions and the ASSET Query Engine (QE) to efficiently evaluate ASSET queries. We argue that ASSET queries: a) constitute a useful class of OLAP queries, b) are suitable for distributed processing settings, and c) extend the MapReduce paradigm in a declarative way.

Categories and Subject Descriptors
H.2.3 [**Database Management**]: Languages—*Query Languages*

General Terms
Design, Management, Languages.

Keywords
MapReduce , Spreadsheets, ASSET Queries, DataMingler.

1. INTRODUCTION
Business needs of modern applications require advanced data analysis over voluminous data sets, usually partitioned across different disks or processing nodes, possibly in different formats (e.g. flat files and/or multiple-vendor databases). To accommodate the enormous processing requirements of these applications, novel hardware/database architectures have been proposed (e.g. [8],[17]) and programming paradigms have been developed (e.g. MapReduce [9].) In addition, several well-funded start-ups, such as AsterData, Greenplum, Netezza and Vertica now offer products for large-scale data analytics alongside with IBM, Oracle and Teradata.

The goals of this work are the following:

(a) Distributed computation of OLAP queries over very large data warehouses: a data warehouse may be distributed to several nodes for reliability, load-efficiency and cost-efficiency reasons. An interesting research question is what kind of OLAP questions can be efficiently evaluated over distributed data warehouses, and how much and which part of the computation one can "move" to the node hosting a partition.
(b) Simple query formulation: representation of a query should be as simple and intuitive as possible, yet amenable to appropriate distributed processing rewrites. MapReduce is an option, but it lacks declarative simplicity ([1],[10],[14]). In addition, there is a plethora of OLAP queries that could benefit from a MapReduce implementation, but cannot be expressed as MapReduce jobs (e.g. pivoting, hierarchical comparisons, complex comparisons, trends, correlated aggregation [7].)
(c) Heterogeneous data sources: data sources can be persistent or continuous, databases of different vendors, or flat files. In general, a data source can be anything that presents a relational interface to our system and has an iterator defined over it. While we briefly discuss ASSET queries in the context of data streams in this paper, a detailed presentation can be found in [4].

As a running example, consider a financial application with schema:

Clients(clientID, address, zip, income)
Stocks(stockID, categID, description)
Transactions(clientID, stockID, volume, timestamp, type, amount)

Typical data warehousing queries include:

Q1. For each stock, find certain demographics of its buyers (e.g. average income) and compare them to those of the stock's category.
Q2. Find the most frequent stock category a user buys or sells, for a predefined set of users.
Q3. For each stock, compare the demographics (e.g. average income) of its large-volume buyers versus its small-volume buyers.

* Part of this research was performed while the author was visiting Aster Data Systems Inc., Redwood Shores, CA.

Additional queries of this "style" can also be found in [6],[7]. They constitute a useful subclass of OLAP queries and share a pattern: for each tuple of a table, they compute a "define-subset, reduce-subset" sequence, where each "define-subset" phase of a step uses previously defined aggregates and/or the tuple's attributes. For example, consider query Q2. While SQL is the de facto option, one could formulate it in a stepwise, set-oriented fashion:

```
1:  for each user u {
2:    T={find the transactions of u};
3:    F={StockIDs of T's members};
4:    S={select the stocks with StockID in F};
5:    C= mostOften(S.categID);
6:    print (u, C);
7:  }
```

Line 2 defines a subset of transactions T; Line 3 defines a (set-valued) aggregate over T; Line 4 defines a subset of stocks; Line 5 defines an aggregate over this set. One could find such formulations easier than SQL, especially those with a procedural background. At the same time, such implementations, with the appropriate indexing, data structure selection and memory sizes can perform significantly better than current DBMS. Finally, if a data source is partitioned across nodes, parallel processing becomes "cleaner". Such queries resemble MapReduce jobs, but there are two notable differences: (a) there may be several reduce sets for the same map value, and (b) these reduce sets can be correlated.

Our approach is based on the concept of associated set (ASSET), which is just a collection of tuples associated to a value (not necessarily atomic). An ASSET query consists of one or more associated sets, put together in a specific way. It resembles a spreadsheet report, where earlier columns serve as the basis for later columns via formulas and the first few columns are somehow initialized to external values. In our framework, users initially define the base columns using a database or flat file and they then build incrementally the report by adding columns. Each cell of the new column represents a data set (the associated set), populated from a data source according to a condition θ, involving attributes of the source and aggregates of previously defined associated sets. We claim that this column-wise formulation of a query can be not only intuitive and flexible, but also efficient and robust in terms of associated sets' evaluation. Similar claims for spreadsheet-like query languages can be found in [3],[18].

In this paper, we present (a) the ASSET Query Engine (QE), which parses, optimizes and executes ASSET queries, and (b) DataMingler, a spreadsheet-like tool to express ASSET queries. In our view, ASSET queries can be useful in:

Novel Programming Paradigms: MapReduce [9] is one of the most active research issues over the last few years. The claim is that with appropriate configuration of the Map and Reduce functions, a large number of computational tasks can be easily represented and efficiently executed. While this approach offers significant procedural flexibility over declarative approaches and employs a simple computational model, it lacks the optimizability and ease of use of modern database systems [10]. While we completely agree with the claims of [10], the ability to loop over the values of a domain and define an (associated to that value) data set is quite appealing both in terms of representation and evaluation. It has been used, directly or indirectly, in parameterized query processing, in set-valued attribute proposals [13], in grouping constructs [6], relational join operators ([5],[16]) and elsewhere. The goal should be to balance the trade-offs between declarative optimizability and procedural flexibility in a database-proper way, such as in Hive [12] and Pig [14]. For example, Pig adds a semi-declarative layer over MapReduce, by proposing a language combining declarative and procedural features. In our approach, ASSET queries "restrict" the Map function to have a declarative nature while the Reduce function can be anything (using C++).

Performance: The answer to an ASSET query is represented by a – possibly nested – data structure, which can always be made memory resident (horizontally segmented, if necessary). This data structure can be indexed multiple times, using different methods; can be decorrelated, if parts of it contain the same data; can be computed locally or partitioned and sent to the nodes containing the data sources. The claim is that the representation of an ASSET query is appropriate to identify and employ the above-mentioned optimization techniques.

Integrating Heterogeneous Data Sources: By allowing a column's data source to be essentially an iterator over anything that presents a relational interface, we can integrate into the same report data sets from heterogeneous data sources.

SQL Extensions: Sticking to SQL for query formulation has traditionally been a "must do" for every proposal, given its popularity among database users. Fortunately, it seems that there is a simple and intuitive SQL extension that allows ASSET query formulation, following the syntactic paradigm of [7].

2. EVALUATION OF ASSET QUERIES

Let us re-visit query Q2. The basic idea is simple. First, for each client id (table B_0 in Figure 1), define the set of client's transactions $\mathcal{T}$, using as data source the Transactions table, and keep the stock ids of $\mathcal{T}$ as a set-valued aggregate function, called *all()*. These are the F_i sets in Figure 1. B_0 is then extended with a column containing F_i sets and is named B_1. Then, for each row *i* of B_1, define the set of stocks $\mathcal{S}$ with stockID in F_i, using as data source the Stocks table, and compute the

mostOften(categID) element of $\mathcal{S}$. The entire process of defining column-wise this query is depicted in Figure 1.

Figure 1: Representation of query Q2

Therefore the idea is to start from a set of base columns and add recursively new columns: for each row define a set of values using some condition θ (the associated set), compute one or more aggregates over these sets, extend the base schema with these aggregates and start over. The definition of the associated set is done declaratively, while the aggregate functions over the associated set is a C++ method with a simple- or vector-type return value. Each associated set's definition resembles a single MapReduce application with the mapping phase (initial values) already computed.

The claims are: (a) ASSET queries can *express* more complex OLAP queries than MapReduce, queries similar to [7], and (b) there are significant optimization opportunities in representing a query in such a way.

Let us revisit query Q2: in most distributed data warehousing configurations, Transactions table will be partitioned across several processing nodes, while Stocks table will be replicated at each node. One evaluation plan would be to distribute B_0 to all nodes, compute the partial S_i's, and ship them back to and concatenate them (union) at the coordinating node. The optimizer could deduce that S_1, S_2, etc. are used later only for membership testing and keep them as inverted lists (to the row ids) to facilitate the efficient computation of $\mathcal{S}$ sets. These optimizations are possible due to the representation of ASSET queries. One can argue that the size of B_1 may be large, but given current RAM sizes, it actually is very feasible and makes sense: maximize RAM's role as much as possible. The impact on performance by employing these optimizations techniques can be huge: from non-ending queries down to a few hundreds of seconds (Section 5).

Due to space limitations, we cannot define formally the concept of associated set. In short, given a set of base values V, an associated set A w.r.t. a data source S is a collection of *empty* data structures, able to hold S' elements, *annotated* by the values of V, i.e. there is one data structure for each $v \in V$. One can consider this as the *schema* of the associated set. An *instance* of A, w.r.t. a condition θ, is the collection of the data structures populated by S' elements, according to θ and v. The purpose is to compute aggregates over the data structures. An ASSET query consists of a base set of values V (i.e. a table) and successive definitions of associated sets, where aggregates of previously defined associated sets can be used for the definition of subsequent associated sets. An ASSET query is constructed incrementally, column-by-column, where a column corresponds to an associated set and is described by a data source, a defining condition and the aggregates to be computed over the associated set.

Evaluation of ASSET queries can be optimized in several ways. For example, an associated set and its aggregates can be computed either locally or remotely, where the data source (or partition of it) exists, by sending all the required data to the remote node; not materializing the associated set if the required aggregate computation is distributive or algebraic; build appropriate indexes on previously defined associated sets by analyzing the condition θ (e.g. build - or keep - it as an inverted list, if θ asks for membership into the associated set); choosing the most appropriate data structure to represent the associated set (B+ trees, min-max heaps, etc.); keep a single instance of identical associated sets of different rows of the base table, by creating a linked list within the new column. We have implemented most of these into our ASSET QE.

The first step in evaluating an ASSET query is to assign the associated sets into *computational rounds*: a computational round consists of independent to each other (directly or transitively) associated sets – there is a dependency between two associated sets if the one uses aggregates of the other (in its defining condition or aggregates). The output of a computational round is the base table for the next one. The data sources of the associated sets consist of data partitions. A computational round consists of one or more *basic computations*, one for each data partition of the computational round. It is a local computation of the associated sets using this data partition. To accommodate an optimization framework, we have developed two operators, one for the computational round and one for the basic computation:

- $AS_R(B, A_1(S_1),\ldots A_k(S_k))$, where B is the base table, $A_1,\ldots, A_k$ the associated sets that have to be computed in the computational round and $S_1,\ldots,S_k$ their respective data sources.

- $AS_B(B, S, A)$, where B is the base table, S a data partition and A the set of associated sets using this data partition.

This process is graphically depicted by Figure 2. The base table is sent to each data partition's node and the basic computation executes there. The whole process is coordinated by a main (coordinating) program. For example, consider Query Q2. Transactions table may be partitioned across several nodes (sources). The list of clientIDs (i.e. B_0) could be sent to all nodes, the partial F_is computed locally at the node and sent back to the coordinator, where they are concatenated to form F_is.

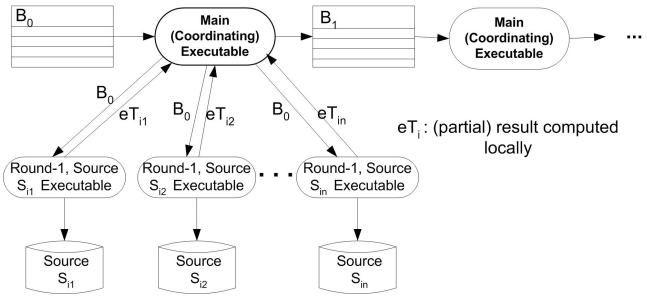


Figure 2: Evaluation of ASSET queries

This plain execution strategy can be generalized to a more elaborate one, where base tables are horizontally partitioned according to their estimated sizes and the computation of the ASSET query "flows" from left to right, possibly in parallel. The following algorithm describes this idea:

```
1: compute B0; round=1;
2: repeat {
3:    Partition B0 to B0^1, B0^2, …, B0^k;
4:    A={the set of assoc. sets of current round};
5:    B1 = AS_R^round(B0^1,A) U...U AS_R^round(B0^k,A);
6:    round++;
7:    B0 = B1;
8: } until (round>n);
```

Note that the combination step of line 5 can be omitted if partitioning remains the same from one computational round to the next. In general, one can think of an evaluation strategy represented by a graph with fork and join points at the end of computational rounds.

The complete architecture of our system is shown in Figure 3. An ASSET query is formulated either graphically using DataMingler or textually using an extended version of SQL . In both cases, an XML-based specification file is generated. Section 3 describes query formulation of ASSET queries. The XML-based specification file of an ASSET query is passed to the main parser (assetGenGlobal), which coordinates the execution of two lower-level parsers (assetGenRound and assetGenBasic) and produces the main (coordinating) C++ program. The assetGenRound parser assigns the associated sets to computational rounds and the assetGenBasic parser generates efficient C++ programs implementing the basic computations. The main C++ program manages the ASSET structure (the data structure representing the answer of the ASSET query) and coordinates the basic computations. All the generated C++ programs are then compiled and executed. Section 4 presents the ASSET Query Engine.

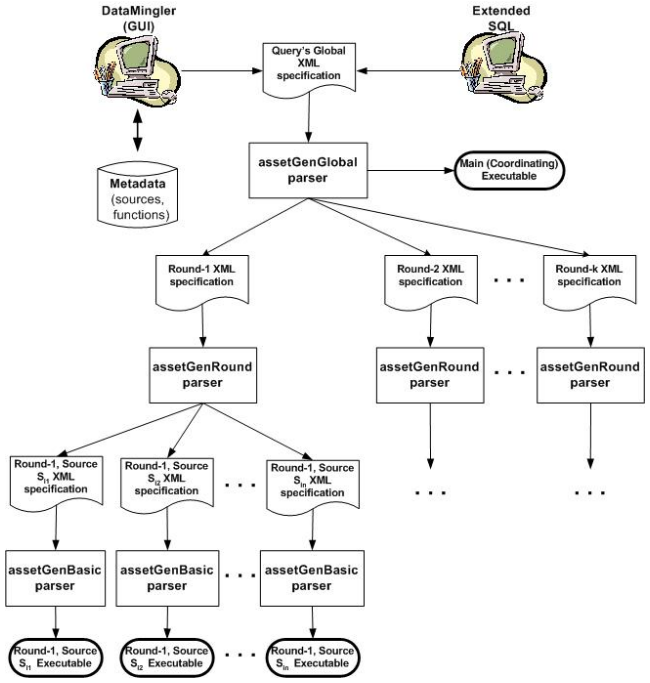


Figure 3: Expressing and evaluating ASSET queries

3. QUERY FORMULATION

ASSET queries can be formulated by either a GUI called DataMingler, or an extended SQL textual interface. Both generate an XML-based specification of the query that is fed to the ASSET QE.

3.1 Extended SQL

We follow the formalism of grouping variables [7]. The idea is quite simple: we want a syntax that allows the addition of "extra" columns to the resulting table of an SQL query – similar to an outer-join operation. We propose an "extended by" clause to declare the associated sets and their respective data sources and a "such that" clause to provide their defining conditions. These clauses immediately follow a <select..from..where> query. The proposed syntax is:

```
select A from R where θ group by A'
extended by A1(S1), A2(S2), … , An(Sn)
such that θ1, θ2, ..., θn
```

The selection list A may contain aggregate functions defined over the associated sets A_1, A_2, …, A_n. The answer

of the <select..from..where.. group by> SQL query serves as the base-values table. Condition θ_i involves attributes of the base-values table, constants and aggregates of associated sets $A_1,\ldots,A_{i-1}$., i=1,…n. For example, query Q2 can be expressed as (recall from Section 2 that *all(attr)* is an aggregate function returning the set of *attr* values of the involved table):

```
select clientID, Y.mostOften(categID)
from Clients
extended by X(Transactions), Y(Stocks)
such that X.clientID=clientID,
          Y.stockID in X.all(stockID)
```

3.2 DataMingler – A Spreadsheet-Like Tool

We have developed a spreadsheet-like GUI to manage data sources, user-defined aggregate functions and ASSET queries. It has been implemented in C++ code using the Qt4 C++ library for Windows from Trolltech. Since Qt is platform independent, DataMingler can be easily compiled for Unix/Linux operating systems.

3.2.1 Data Source Management

An ASSET query uses heterogeneous and possibly multi-partitioned data sources. These sources may refer to local or remote databases, data streams or flat files and must firstly be appropriately defined through DataMingler. Each description consists of the source's schema and a number of attributes specific to the type of the source (e.g. delimiter and location for flat files; IP, port, username and password for databases, etc.) All data sources are stored in an XML-based specification file. Currently we support databases (Postgres, MySQL, Oracle, SQL Server), flat files and socket-based streams. Figure 4 shows the first step in defining the Transactions data source.

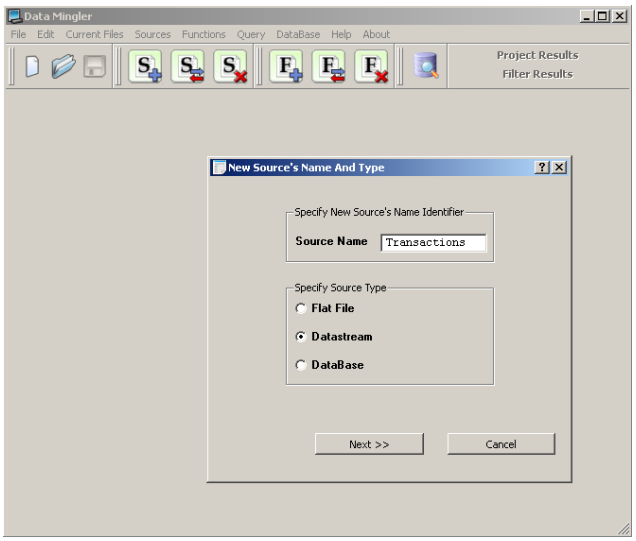


Figure 4: Defining a new data source in DataMingler

All data sources may consist of multiple partitions, not necessarily of the same schema – only common attributes appear in query formulation. A partition in the case of databases/flat files/data stream is just another table/file/stream source, located locally or remotely. As a result, a data source may consist of multiple tables/files/streams distributed at several processing nodes.

3.2.2 Aggregate Functions

The goal is to describe the signature of a C++ function into an XML-based dictionary, so some type-checking and user-guidance can take place. The user specifies the input parameters and their types and the type of the return value (Figure 5).

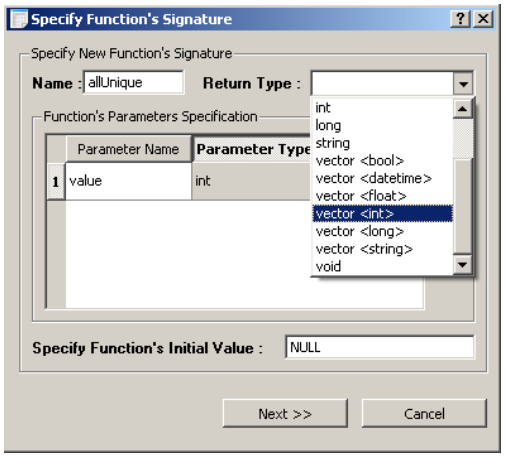


Figure 5: Defining a new aggregate function, allUnique, returning a set of distinct integers

The user also specifies a "gluing" function, in the case of distributed computation of an associated set (e.g. "sum" is the gluing function for "count".) Aggregate functions can be either distributive or algebraic (holistic computations can be achieved through aggregate functions returning the entire or part of the associated set and the use of "null" associated sets, described later). In the case of algebraic aggregate functions, the user must specify the involved distributive functions, the correspondence between the parameters and the finalization code (in C++).

3.2.3 Asset Queries

Users specify ASSET Queries through DataMingler in a spreadsheet-like manner, column by column. The user initially specifies a base table that can be an SQL query over one of the database sources, the contents of a flat file source or manually inserted schema and values. Thus, the first columns of the spreadsheet correspond to the base-values table attributes. The spreadsheet document is then extended with columns representing associated sets, one at a time. The user specifies the name, source, defining condition and aggregate functions of the associated set. The data source can be (a) one of the existing data sources described earlier through DataMingler, (b) of type "this", in which case the so-far defined spreadsheet table serves as the data source to the associated set, and (c) of type "null", in which case the user specifies an expression involving aggregates of previously defined columns – similar to a

spreadsheet formula involving only same-row cells. Figure 6 shows the window for the definition of an associated set.

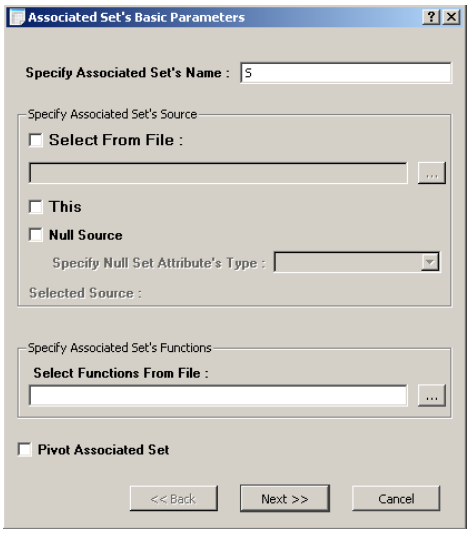


Figure 6: Building an ASSET query

Associated sets may be correlated, since aggregations performed over one associated set may be used by another. This might occur during specification of the latter's defining condition, its functions' parameters or its computation formula in case of "null" sets. DataMingler identifies dependencies, performs a topological sort and places them into "processing rounds", as required by the ASSET QE, explained in the following section.

4. The ASSET QUERY ENGINE (QE)

Once an ASSET query has been formulated and represented as an XML-based specification, it is passed to the ASSET QE for optimization, code generation and execution.

assetGenGlobal: This is the top-level parser of the ASSET QE. It gets the XML-based specification of an ASSET query and generates (a) the round-related XML specifications of the query and (b) the main (coordinating) C++ program for the query. Each round-related specification contains the data sources' description of the round and the associated sets that will be computed. Note that from this point on, each partition of a data source becomes a distinct, individual data source. The query's main C++ program, instantiates and populates all the necessary data structures, creates all the local indexes and decorrelation lists over the ASSET structure and coordinates all the *basic computational threads* executing locally or remotely. In the latter case, it sends parts of the ASSET structure to the appropriate nodes and receives back (and glues together) the computed column(s).

assetGenRound: This is the round-level parser: it groups the associated sets of the round by source and generates an XML-based specification file for each source. Recall that with the term "source" we mean partitions of the original data sources. It determines whether the computation over the source will execute locally or remotely, deduces the indexes and decorrelation lists over the base-values table and resolves the minimal base-values table that has to be sent to the remote node (in case of remote computation.) Currently supported indexes are hash maps, binary trees and inverted lists, deduced by the defining condition of the associated sets.

assetGenBasic: This is the source-level parser that gets a source-specific XML-based specification file and generates an efficient C++ program (the "basic computational thread") to scan the data source and compute the associated sets related to that source. This thread communicates with the main program to receive the round-specific base table (only the required columns), builds indexes over and decorrelates the base table, computes the associated sets and serializes the result back to the coordinating program (if executing remotely). The engine also decides to decorrelate the base table on a single attribute with respect to an associated set (i.e. we may have different decorrelation lists for different associated sets), if the associated set is using a hash index on that attribute and its estimated cardinality is low (this can be measured while receiving the base table).

Once all the basic computational threads have been generated, then the whole process is driven by the query's main C++ program. We currently assume that the entire ASSET structure (the output of the ASSET query) fits in main memory – which is not unrealistic for a large class of ASSET queries and today's memory sizes. However, since the entire code generation assumes boundary limits of the ASSET structure, we can easily specify the computation of an ASSET query in horizontal chunks - currently has to be done manually, by altering the query's main C++ program.

5. PERFORMANCE

Figure 7 shows the performance of the query Q2 when the size of Transactions table varies from 100M to 600M records (15GB to 90GB) – all in one partition. We assumed 10M clients and 10K stocks. We tried to compare ASSET query engine's performance with standard SQL formulation using PostgreSQL but we could not get any results for even 200M records after 21 hours. All experiments performed on a Linux Dell machine with a Quad Core Intel Xeon CPU @ 3.00GHz having 12 disks, 300GB each at 15K rpm, RAID5 configuration and 32GB of main memory.

6. CONCLUSIONS AND FUTURE WORK

In this paper we introduced ASSET queries, a useful class of data analysis queries that can be efficiently evaluated in distributed settings. While the basic component of an ASSET query, the associated set, is similar to a MapReduce operator, the ability to have multiple

associated sets, possibly correlated, placed next to each other in the same query, increases greatly the complexity of data analysis performed and offers significant optimization capabilities.

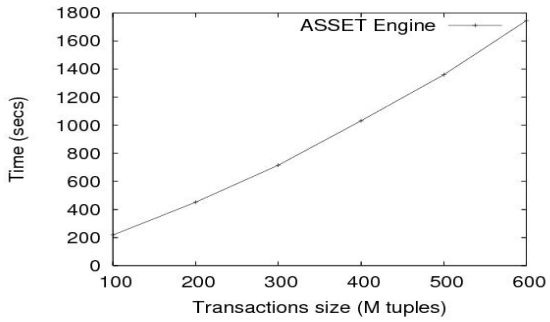


Figure 7: ASSET QE performance on query Q2

The main differences between ASSET queries and MapReduce can be summarized as: a) the data set of the mapping phase is defined using a condition, instead of using an arbitrary mapping function – less flexibility, more room for optimization, b) one may define multiple reduce sets for the same value, by using different defining conditions – this is not obvious how it can be done in *one* MapReduce, c) correlated aggregation ([6],[11]), an important topic in data analysis, can be expressed in one query instead of multiple, providing additional hints to the optimizer, and d) while overlapping reduce sets can be easily defined in ASSET queries by arbitrary conditions, in MapReduce special configuration of the mapping function is required.

We argued in [4] that ASSET queries' framework seems appropriate for expressing a significant class of data stream queries. In this case, data sources are continuous streams of data, associated sets are frequently represented as queues and the dependencies between associated sets dictate the order of update of the ASSET structure.

Current research involves theoretical work, improving implementation and benchmarking. We would like to: (a) develop a theoretical (algebraic) framework for associated sets, (b) extend our optimization platform, and (c) compare our performance results to those in [1] and [15]. In addition, it seems that a query language over the *powerset* of a relation (defining a set of subsets of the relation) could unify several proposals for ad hoc OLAP. We are investigating whether the associated sets framework can serve as the basis of such a language.

7. ACKNOWLEDGMENTS

This research work was partially supported by EU-ICT Grant ST-5-034957-STP. The authors would like to thank Prof. Diomidis Spinellis for pointing out useful references.

8. REFERENCES

[1] Abouzeid, A., et al. *HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Worksloads.* In VLDB, 2009, 922-933.

[2] Akinde, M. O., Bohlen, M. H., Johnson, T., Lakshmanan, L. V. S., Srivastava, D. *Efficient OLAP Query Processing in Distributed Data Warehouses*. In EDBT, 2002, 336-353.

[3] Bin, L. and Jagadish, HV: *A Spreadsheet Algebra for a Direct Data Manipulation Query Interface.* In ICDE, 2009, 417-428.

[4] Chatziantoniou, D., Pramatari, K., Sotiropoulos, Y. *COSTES: Continuous Spreadsheet-like Computations.* In Intern. Workshop on RFID Data Management (ICDE 2008).

[5] Chatziantoniou, D., Akinde, M., Johnson, T. and S. Kim, S. *The MD-Join: An Operator for Complex OLAP.* In Int. Conf. on Data Engineering (ICDE), 2001, 524-533.

[6] Chatziantoniou, D. and Ross, K. *Querying Multiple Features of Groups in Relational Databases*. In VLDB, 1996, 295-306.

[7] Chatziantoniou, D. *Using grouping variables to express complex decision support queries*. In DKE Journal, 61(1), 2007, 114-136.

[8] Cieslewicz, J., Berry, J., et al. *Realizing parallelism in database operations*. In DaMoN, 2006.

[9] Dean, J. and Ghemawat S. *MapReduce: Simplified Data Processing on Large Clusters*. In 6th Symp. on Operating System Design and Implementation, 2004, 137-150.

[10] DeWitt, D.J, Stonebraker, M. *MapReduce: A major step backwards.* The Database Column, http://www.databasecolumn.com/2008/01/mapreduce-a-major-step-back.html

[11] Gehrke, J., Korn, F. and Srivastava, D. *On Computing Correlated Aggregates Over Continual Data Streams.* In SIGMOD Conference, 2001, 13-24.

[12] Hive Project. `http://hadoop.apache.org/hive/`

[13] Mamoulis, N. *Efficient Processing of Joins on Set-valued Attributes.* In ACM SIGMOD, 2003, 157-168.

[14] Olston, C., Reed, B., Srivastava, U., Kumar, R. and Tomkins, A. *Pig latin: a not-so-foreign language for data processing.* In ACM SIGMOD, 2008, 1099-1110.

[15] Pavlo, A., et al. *A Comparison of Approaches to Large-Scale Data Analysis.* In ACM SIGMOD, 2009, 165-178.

[16] Steenhagen, HJ., Apers, P., Blanken, HM. *Optimization of nested queries in a complex object model*. In EDBT, 1994, 337-350.

[17] Stonebraker, M., et al. *C-Store: A Column-oriented DBMS.* In Intern. Conf. on VLDB, 2005, 553-564.

[18] Witkowski, A., Bellamkonda, S., et al. *Spreadsheets in RDBMS for OLAP.* In SIGMOD, 2003, 52-63.

Peter Buneman Speaks Out

on Phylogeny, the Integration of Databases and Programming Languages, Curated Databases, British Plumbing, the Value of Talking to Users, When to Ignore the Literature, and More

by Marianne Winslett

http://homepages.inf.ed.ac.uk/opb/

[A brief reminder: http://www.sigmod.org/interviews has video versions of many installments of this column, as well as many interviews that have not yet appeared in print. For example, the video version of this issue's column has been available for years.]

Welcome to this installment of ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today we are at the 2006 SIGMOD conference in Chicago. I have here with me Peter Buneman, who is a professor in the School of Informatics at the University of Edinburgh. Before that, he was a professor at the University of Pennsylvania for many years. He has research interests in databases, programming languages, scientific databases, and mathematical phylogeny. Peter and his coauthors received a best paper award from Computer Networks in 2002. He is an ACM Fellow, a trustee of the VLDB Endowment, and a Fellow of the Royal Society of Edinburgh. His PhD is from the University of Warwick. So, Peter, welcome!

I understand that you like very good whiskey, and Scotland is famous for its whiskey. Is that why you went back to Scotland?

Well, I do like very good whiskey, although I do not drink it very much; but no, that is not the reason I went back to Scotland. It is a rather complicated story. I have always had an affection for Scotland; I know the place very well. We have had a house in a remote part of Scotland for a long time. My return was certainly helped by the fact that I was very comfortable in Edinburgh, which is a city that I know very well, and where I have lots of friends. I probably wouldn't have tried anything much more adventurous than going back to Edinburgh. I think the reasons for going back will emerge later on as we talk.

You have been building a very strong research group in Edinburgh. I know what a big job that is! What motivates you to take on that task?

I think I went slightly mad! [Laughs.] I have to say that I was extremely comfortable and happy at the University of Pennsylvania. I had wonderful colleagues there. But there is a point at which you realize you are becoming a bit of an old fogy, and part of the furniture. At that point you should step aside and let your younger colleagues take charge – they are superbly competent, so why not? – and you should try something else. Either you are going to stay there forever, or you should get out and do something new. Edinburgh made me a very attractive offer, and I thought it would be an interesting challenge to build up a really good database research group in the UK, which frankly has lacked one for a long time. We have been quite successful in building up a database group. Also, as I am sure you are aware, the UK has also hired Georg Gottlob, so I think that all combined, we are doing very well.

You are not the only database researcher who has recently moved from the US to Europe. Have conditions recently changed in some way that makes this route more attractive?

I think Europe is becoming a bit more competitive now, waking up to the need to bring computer science researchers back from the US. There are increasing funds and opportunities for doing this, and I was attracted partly on such funding. I don't think it is a major trend at the moment. I don't think we are going to see a mass defection.

What is the difference between the educational systems at Edinburgh and the University of Pennsylvania – or, if you prefer, between the UK and the US?

From the point of view of a teacher, I think they are rather minor, until you get to the PhD. The PhD in the UK is much shorter; the UK students are expected to do it in three years. As you know, in databases in the US, the PhD can take 5 or 6 years, sometimes even longer. That means that when UK PhDs graduate, their publication records aren't typically what you would expect of a US PhD. This means that in order to be competitive in the US market, they have to go on and do postdocs. So there are many more postdoctoral positions in the UK than in the US, and this is partially how we have populated our database group at Edinburgh.

I see this trend toward postdocs increasing in the US. I think as the academic job market saturates here in the US, computer science researchers generally are going to be in less immediate demand. Then you are going to see an increasing number of postdocs in universities. Postdocs are already common in other subjects, such as physics and biology, where students hardly ever go straight from a PhD into a teaching job.

You began your research career in functional programming and type theory. How did you get from there to here?

Actually, I began my research career in mathematical phylogeny. Perhaps I should tell the story slightly differently. I started by doing some work on the phylogeny – the derivation – of medieval manuscripts that used to be copied from each other. A professor of the English language wanted to know what the tree of copying was. There was some very interesting mathematics to be done in the work to compute that tree. When we did that work, we knew that the techniques were also applicable in genetics, but there wasn't much genetic material around at that time to try it out on. We tried it on some common protein sequences and it worked very well.

Then, several things happened. One of them was that I met my wife, who is an American, and we decided to move to America. When I got to America, I told the chairman of my department about what I had been doing. He said that medieval manuscripts weren't exactly wealth-creating, and that I should go off and talk to people who do databases. So that is how I got into the field. Of course now, mathematical phylogeny is big business, and the techniques I worked on are quite widely used.

I was always fascinated by programming languages, and I always felt there should be a stronger connection between programming languages and databases. I used to enjoy programming very much, functional programming in particular.

You were one of the first people to work in the intersection of databases and programming languages. Why is it that after 25 years of people thinking about the relationship between programming languages and databases, the two communities largely still don't talk to each other?

I don't think that is quite true. I think there is quite a lot of back and forth between the two communities. I know a number of database people going to programming language conferences, and vice versa. For example, certainly there are at least two talks at this conference (SIGMOD 2006) given by programming language people. And I think that it is quite good that the database community brings these people in. If you look at PODS, you will find a lot of papers now with people worrying seriously about type systems and things like that. That work is directly influenced by the programming language community.

I think the more interesting question is whether there has been any *practical* meeting of the two communities; and I think there has been. There is now, of course, the realization that SQL is a functional programming language and people are trying to keep it that way, and make it compositional, and talk about these kinds of properties. But more importantly, if you look at the things that are going on in C# and Linq, you will see exactly what we have been striving for: a clean integration of databases into nice programming languages. And a lot of the ideas lead back, through a chain of work, to the original work on the relationship of programming languages to databases.

The problem is that in computer science, we expect immediate gratification. We don't expect an invention in physics to have commercial impact – though if it did, that would be regarded as great. Even in computer science, sometimes it can take a while for a good new idea to have commercial impact.

You are a coauthor of a well-known paper that talks about the principles of object-oriented databases. Mike Stonebraker recently told us that "object databases are a zero billion dollar market." So was all that object-oriented database research a waste of time?

I don't think so. First of all, that paper was not on object-oriented databases, it was on *type systems* for databases. It was called "Types and Persistence in Database Programming Languages" (ACM Computing Surveys, June 1987). And when we wrote it, we did not know a lot about object-oriented databases. But, I think generally, that work, and the work on object-oriented databases, although they have not been hugely successful commercially, have been very influential in other aspects of databases. For example, we have object-relational systems, and we have Mike Stonebraker's Postgres, all of which were heavily influenced by research in object-oriented databases. I don't know whether object databases are a zero billion dollar market or not, but the work was influential, and it certainly changed the course of database research. Again, we should not necessarily judge ideas by their immediate commercial value.

Mike Stonebraker also said that XML is a very good format for transferring data, but other than that, it is largely a pain. Do you agree?

I have to agree! XML is designed for data exchange. As such, it is very nice; it is great to have a standard format for moving data around, and not have everyone invent a new one, which is what was happening before. But I think that what has happened is that people are trying to turn XML into a data model, or something like that. Look, I am a teacher; I have to teach about databases for a living. Now, as you know, when you teach relational database design, it is quite difficult. Students have a hard time doing it, and they create really bad designs. XML is a serialization format, and things like XML Schema and DTDs are very complicated. To get these students to design databases using XML Schema would be terribly difficult, really impossible. We shouldn't be thinking that way. It would complicate the process unnecessarily to say that we should also be thinking about the serialization of data when we design its schema. That would just be wrong. I think the basic idea of XML is very nice, but all this elaborate superstructure has gone too far. When it comes to database design, we have to simplify. We have to look for very simple core ideas, and not try to make it more and more complicated.

You have been working on data provenance for a number of years. Where will the research on data provenance have its impact?

I hope ultimately it will have impact in various areas. One is the way in which we design databases and use them. For example, it is only recently that we have started to think that maybe when we use databases we should actually make them record their whole history. It is true that in commercial databases you can roll back to a limited extent, but that is not the same as recording all history. You can't ask questions across time with a database that just supports rollback.

Just the other day, I was talking to someone who had built a database for the government to record the real estates sales in Edinburgh. As soon as a house was sold, they updated the old record to reflect the new owners. The database had absolutely no concept of history, and so the real estate lawyers were no better off than before; they had to shuffle through all the paper transactions to determine the history of a house. Omitting historical records from the database design was a huge mistake, but it is the sort of thing that is all too easily done when people design a new database.

That example points to one way in which we can see provenance potentially have an effect: we can design databases in a different way. There are people thinking about this, and devising ways

in which this notion of history can be built into database systems so that they can automatically keep provenance information properly. Remembering history is very important, because provenance is one of the major aspects of data quality, especially in scientific data. People just don't believe what they find on the internet unless they know who put it there.

How would you address the scalability issues if support for provenance is built into a typical DBMS?

That is what we are working on at the moment: we are trying to find efficient ways of recording provenance.

There is another answer to that same question, in that provenance tends to be more important in curated databases, which typically are rather small. In large databases, things are much more uniform, with large blobs of information all coming from the same place and sharing the same provenance. So, we have to find efficient coding techniques for encoding provenance in these larger databases, but I think it is going to work. In other words, I think we are going to get large databases that look fairly uniform in respect to provenance, and small databases that tend to be more heterogeneous with respect to provenance. I think we are going to find good ways of recording both kinds of provenance.

What is the next big thing in research related to data provenance?

Provenance is part of data quality, as I said earlier. Provenance is also very closely related to data integration. We tend to lose provenance information during integration. But then there are other things like annotation, database archiving, recording the history of databases, citation of data in databases – all of these are closely related to provenance, and we are just starting to work on them. And of course there is lots more work to do on data quality.

What are curated databases?

That's a good question! Go to the reference shelves of your local library; you probably haven't been there for ten years, right? A lot of old encyclopedias, dictionaries, gazetteers – people just stopped printing them around 1990 because they have all become databases. You don't go and look things up in the paper version of the Encyclopedia Britannica now; you look them up in the online subscription version of the Encyclopedia Britannica. All these books that used to be on the reference shelves of the library are being published in curated databases now.

These databases are the result of human effort: people decide which data to put into them, in an extremely labor intensive process. I just found out that the Oxford English Dictionary has about 50 people working full time putting entries into it, and another 100 people working part time.

At the same time, the ease of publishing data in this form has produced a wealth of new curated databases. This is very clear in molecular biology, where there are something like 800 curated databases. All of these were constructed by people; they are like large collaborative books. The move to curated databases has occurred in the past ten years, and we have largely ignored the problems of these people.

What are the problems of these people?

We have talked about some of them already. For example, provenance! There used to be standards that people used to explain exactly where each piece of data came from; now people are

just cutting and pasting data into their database. They don't want to ignore the problem of recording where they took the data from, but life is too short for them to record it. Provenance information is largely made up of annotations, and how do you annotate something that is in a database with a fairly rigid structure?

Of course, the knowledge in the curated database evolves, and you want to keep a history of the previous states of knowledge, so you also need to support archiving.

Database curators also face the more general problem of data quality. Errors in the data are common. Can we use automated data cleaning techniques? There is a lot of good work to be done there.

As always, there is a certain amount of data transformation going on when data is added to a curated database, and there is data integration as well.

The main point to make is that we've been thinking of provenance or lineage, or whatever it is called, in the context of views. But curated databases are not views. They are built by people, by the sweat of their brow. They are very expensive databases. I have found a lot of very interesting problems associated with curated databases, and the database community is just starting to look at them.

What is a good methodology for choosing research topics?

I know I am more associated with theory than with systems, but I have always found that the greatest inspiration comes from talking to the people who actually use databases, the customers. And here you have to be careful. If you go to a scientist and you say, "I do databases," they will come to you for help with their SQL, or with building their database, and so on. But if you dig a little deeper, you often find that they are trying to do something with their data that we haven't thought about. Most of my recent work has been prompted by doing just that – talking to users who are consumers of database technology, who are faced with these problems.

To give a very recent example, a colleague of mine at Edinburgh was doing something very strange. I couldn't figure out why he was doing this. He had one of these curated databases built by contributions from several hundred people. He was trying to give everybody credit for their bits of the database. So he was trying to produce a citation system for his database. That comes back to the question of how we can cite data in databases. That problem is something you can actually work on rather easily, and you can get some simple solutions.

How do you like to spend your time outside of work?

I spend a lot of time at my house in the highlands. I also like to go sailing or boating, and I have a boat on the sea. I also enjoy building things. I have quite a big workshop, and I make things. I think it is very relaxing. To sit at a lathe and turn a bowl – it takes your mind off everything. I also enjoy the usual things like walking and music.

What have you made in your workshop?

Last week I finished making a desk for myself, and now I can put all my computers in it. My wife hates computers with lots of cables all over the place, so I built a desk in which all the cables and wires are carefully hidden.

And I understand that you did this with equipment that you imported from America?

My wife liked the idea of building a house in Scotland, but she wasn't going to give up American plumbing or her hairdryer. I said that we could buy a hairdryer in Britain. She said no to that, so I agreed that we could buy a transformer to convert the current. Then we realized that we would be bringing a lot of other electrical items from America, and so we wanted the house to have both American and British wiring. We put mostly US plumbing in it as well. I also built a workshop, and since we had a 110 volt power supply, I loaded the moving van up with lots of power tools. At that time, and probably still now, power tools were a lot cheaper in the US than they are in the UK.

You have been the program chair of SIGMOD, PODS and ICDT. Do you have any comments on the conference system?

This is a very difficult problem, and I am not sure I have anything new to add to the discussion. It is really sad when you see good papers rejected because of incompetent refereeing, and this does happen.

There are good things and bad things about SIGMOD and VLDB. One is that they are very extensive, and inclusive; they tend not to be narrow. But everybody will tell you that their best submitted papers were rejected from these conferences, and their weaker papers were accepted. When you get older, you regard these outcomes as a statistical thing, but it is very hard to tell your students that the paper they have put their hearts into has been rejected. I wish we could somehow strengthen the quality of refereeing by ensuring that people with the appropriate skills are assigned to referee each paper.

Maybe the issues with our conference system will just shake out in time, and we will become more like other disciplines, with a more stable view of what the subject is. I think that our current problems are a necessary side effect of the nice property that our field is permanently in a state of flux, and permanently rather popular and successful as a research area.

Do you have any words of advice for fledgling or midcareer database researchers or practitioners?

When I was very young – when I was a post-doc, actually – I was reading the literature on some topic or other, when my postdoctoral advisor, who was a very brilliant but rather crusty gentleman, came in and said, "You have done enough reading! Ignore the literature. Go away and think!" It turned out to be quite good advice because two of the ideas I had were quite original, and the third one was Quicksort! Fortunately I realized that my idea for Quicksort was not original before I tried to publish it.

This was of course a very long time ago. Still, I think at some point it is a good idea to stop reading the papers, and do what I said earlier: talk to the users of databases, or talk to people who work in other fields. You often get very good ideas from just casual remarks that people make, or very odd things that they do.

Among all your past research, do you have a favorite piece of work?

I still have some fondness for the work I did in mathematical phylogeny, because it did have real impact. Those were the days when you could do very simple things, and they would work out well.

I think the cumulative effect of the work on the interaction between programming languages and databases is important, but I cannot pinpoint one particular paper. I think it has had a beneficial effect on the community.

If you magically had enough extra time to do one additional thing at work that you are not doing now, what would it be?

At work, I would love to go off and do something wacky like study the confluence of information theory and physics. I always have tremendous respect for people who have time to do that sort of thing; unfortunately, I am not likely to have the time. In databases, there are a lot of new things that I would like to keep up with that I can't. For example, I think some of this new work on probabilistic databases is very interesting. I would love to keep up with that, but I haven't been able to. But really, if I had the opportunity, I would go off and work on something completely different.

If you could change one thing about yourself as a computer science researcher, what would it be?

I would organize my time better. I am not a good organizer, and as a result I waste an awful lot of time. I think if I organized my time better I would have much more fun doing research. I am already having lots of fun!

Thank you very much for talking with me today.

Thank you.

Report on the 10th International Workshop on Web Information and Data Management (WIDM)*

Chee-Yong Chan
National University of Singapore, Singapore
chancy@comp.nus.edu.sg

Neoklis Polyzotis
University of California-Santa Cruz, USA
alkis@ucsc.edu

1 Introduction

The 10th *ACM International Workshop on Web Information and Data Management (WIDM 2008)* was held in Napa Valley, California, USA, in conjunction with the 17^{th} International Conference on Information and Knowledge Management (CIKM), on October 30, 2008.

Continuing the tradition of the previous WIDM workshops, the main objective of the workshop was to bring together researchers, industrial practitioners, and developers to study how Web information can be extracted, stored, analyzed, and processed to provide useful knowledge to the end users for various advanced database applications.

The call for papers resulted in the submission of 53 papers from 22 countries. The program committee accepted 20 papers that were grouped in the following subject areas: *Data Mining and Clustering*, *Systems Issues*, *Web 2.0 and Social Networks*, and finally *Ranking and Similarity Search*.

2 Paper Presentations

2.1 Data Mining and Clustering

The paper by *M. Hu*, *A. Sun*, and *E.P. Lim* entitled *Event Detection with Common User Interests* deals with the problem of identifying events that can be detected through the publication of online documents (articles, blog entries, etc.) and the search queries posed over said documents. The proposed solution correlates the stream of queries to the stream of documents and maps each query to a set of documents such that (a) the documents are published near the time of the query, and (b) the documents are relevant for the query. Events are detected by computing clusters of queries based on the similarity between the corresponding document-sets, where each cluster represents queries on the same event.

P. Senellart, *A. Mittal*, *D. Muschick*, *R. Gilleron*, and *M. Tomassi* in their paper entitled *Automatic Wrapper Induction from Hidden-Web Sources with Domain Knowledge* propose an automated method to infer a web service wrapper for an HTML form. The method works in two stages, using solely knowledge about the domain of the data source accessed through the form. First, the form's fields are matched to concepts based on lexical information and the provided domain knowledge, and the matching is verified by probing the form with instances of the concepts. Subsequently, a unsupervised machine learning algorithm is used (bootstrapped with the domain knowledge) in order to understand the structure of the HTML pages returned by invoking the form.

In the paper entitled *PIXSAR: Incremental Reclustering of Augmented XML Trees*, *L. Shnaiderman*, *O. Shmueli* and *R. Bordawekar* propose a clustering-based approach for the physical storage of an XML document. The proposed method maintains access statistics during query processing per parent-child and sibling-sibling edge, and then partitions the XML tree in disjoint sub-trees (each stored on the same page) representing elements that are frequently accessed together in the evaluation of the workload. The statistics are updated continuously and incremental reclustering may be performed if the access patterns in the workload change.

The paper entitled *A Study of the Relationship between Ad Hoc Retrieval and Expert Finding in Enterprise Environment* by *J. Zhu* evaluates how the results of search queries affect the task of expert finding. Specifically, given a search query, the experts are identified by examining the co-occurrence between query terms and expert names in the documents most relevant to the query. The paper analyzes empirically whether the parameters that affect the relevance of documents also have an effect on the identification of experts. The following three parameters are considered: background smoothing, anchor texts, and the in-degree of documents.

Finally, *S. Huang, X. Wu* and *A. Bolivar* in their paper entitled *The Effect of Title Term Suggestion on E-commerce Sites* question the assumption that sellers in e-commerce sites provide a descriptive title to their products. Data collected from eBay shows that a significant number of items have a very short title and are thus missed by customer queries. To address this issue, the authors employ the idea of query expansion: the title of an item is pre-processed at the time of registration, and a set of additional terms are suggested based on terms found in the query logs and titles of other related items. The seller can then select which of the suggested terms should be included in the title in order to increase the chance of successful customer searches.

*http://widm2008.comp.nus.edu.sg/

2.2 System Issues

M. Klein and *M. Nelson* in their paper entitled *A Comparison of Techniques for Estimating IDF Values to generate Lexical Signatures for the Web* evaluate methods for estimating the inverse document frequency (IDF) of terms at the scale of the Web. The authors examine three possible approximation schemes: the NG method uses the Google N-Grams data set and estimates the IDF as the frequency of the corresponding unigram; the LC method estimates the IDF based on a sample of web pages downloaded from the Internet Archive and the Open Directory Project; finally, the SC method googles the term and scrapes the screen of results for the reported document frequency. An empirical comparison shows that the three methods yield similar approximations results.

In their paper entitled *High-Performance Priority Queues for Parallel Crawlers*, *M. Marin, R. Paredes* and *C. Bonacic* examine efficient data structures for prioritizing the URLs downloaded by a highly parallel crawler. The authors propose two new data structures that can be implemented efficiently in a parallel system: (a) a parallel queue that uses binary tournaments upon a complete binary tree in order to identify the top URL, and (b) the Quick Heap structure (QH for short) that uses Hoare's QuickSelect algorithm to perform a partial sorting and identify efficiently the top k URLS, where k is a parameter in the system. An experimental study demonstrates that the new queues can enable significant performance improvements in a parallel crawler.

C. Garcia-Alvarado and C. Ordonez in their paper entitled *Information Retrieval from Digital Libraries in SQL* describe the implementation of an IR framework mostly in standard SQL, with the motivation of supporting ad-hoc information retrieval on top of a conventional RDBMS. The proposed implementation aims to be both portable and efficient, and it supports several common term weighting schemes.

The paper *HiPPIS: An Online P2P System for Efficient Lookups on d-Dimensional Hierarchies* by *K. Doka, D. Tsoumakos* and *N. Koziris* describes a DHT-based index for relational data sets conforming to a star schema. The indexing scheme, termed HiPPIS, indexes the tuples in the DHT by fixing a specific level on each dimension. A query that constrains exactly the same set of levels is answered directly from a single DHT node, whereas any other type of query is answered by flooding. To amortize the cost of flooding, HiPPIS creates soft-state indices that cache the location of tuples for recently flooded queries. Moreover, HiPPIS is able to adjust dynamically the set of indexed dimension levels in order to track changes in the workload.

Finally, *M. Karnstedt, K.U. Sattler, M. Ha, M. Hauswirth, B. Sapkota* and *Roman Schmidt* in their paper entitled *Approximating Query Completeness by Predicting the Number of Answers in DHT-based Web Applications* propose the metric of query completeness as an indicator of query progress in a DHT-based peer-to-peer system. Intuitively, query completeness measures the fraction of answers that have been received compared to the total number of answers. The authors propose and analyze several approximation schemes for this metric that can be computed as the query is routed in the overlay network.

2.3 Web 2.0 and Social Networks

The paper entitled *From Web 1.0 to Web 2.0 and back - How did your Grandma use to tag?* by *S. Kinsella, A. Budura, G. Skobeltsyn, S. Michel, J. Breslin* and *K. Aberer* presents a study to compare the relationship between "Web 1.0 tags" that are extracted from Web 1.0 anchortext and metadata and "Web 2.0 tags" that are obtained from the tagging portal del.icio.us. The study reveals that the Web 1.0 tags generated from a simple tag extraction method have a significant overlap with the Web 2.0 tags. Thus, the simple extraction method can be applied to bootstrap tagging portals or enrich the set of tags in tagging portals.

In the paper entitled *Modeling the Mashup Space*, *S. Abiteboul, O. Greenshpan* and *T. Milo* introduce a formal framework for specifying mashups. A mashup is modeled as a dynamic network of interacting *mashlets*, which are the basic components of the proposed model. Mashlets can query data sources, import other mashlets, use external Web services, and specify complex interaction patterns between its components. The state of a mashlet consists of a set of relations and its logic is expressed in terms of Datalog-style active rules. The concepts of the model is illustrated with a personal health information system demonstrating its expressiveness and usefulness.

In their paper entitled *Nereau: Query Expansion Using Social Bookmark*, *C. Biancalana, A. Micarelli* and *C. Squarcella* present a new approach to enhance query expansion with personalization by exploiting tag information from social bookmarking services. A user model (in the form of a three-dimensional matrix of co-occurrence values) is first built by analyzing the user's previous search queries and visited urls and deriving relevant terms from the visited web pages using the tag information from social bookmarking services. Using the user model, a new search query is expanded into multiple queries with the results organized into categories.

J. Park, T. Fukuhara, I. Ohmukai, H. Takeda and *S.-G. Lee* in their paper entitled *Web Content Summarization Using Social Bookmarks: A New Approach for Social Summarization* propose a novel Web content summarization technique to create text summaries by exploiting user feedback from social bookmarking services. First, representative words are extracted from user comments, which are then used to extract sentences that contain the representative words; the sentences are then scored and a summary is then formed using the top-k sentences.

In the final paper of the session, entitled *Granular Mod-*

eling of Web Documents: Impact on Information Retrieval Systems, *E. Fersini, E. Messina* and *F. Archetti* examine the use of a granular representation of web pages for improving the accuracy of web page classification and improving web page ranking. The approach proposed for the first problem is based on the assumption that a web page's blocks that contain images, referred to as image blocks, contain more significant information about the web page contents. An unsupervised algorithm is proposed to identify the most informative image blocks and their most relevant terms using an inverse term importance metric. In the second problem, the authors exploits the semantic relationships among document blocks for page ranking computation, where the probability of clicking a hyperlink is estimated by the degree of textual coherence between the source and destination web pages through the block containing the hyperlink.

2.4 Ranking and Similarity Search

In their paper *Quantify Music Artist Similarity based on Style and Mood*, *B. Shao*, *T. Li*, and *M. Ogihara* discuss the use of style and mood aspects to quantify music artist similarity. Their proposed approach operates by first obtaining the style and mood descriptions of music artists from the All Music Guide website, which are then used to compute style and mood similarity taxonomies based on a hierarchical co-clustering algorithm. The similarity measure for each of style and mood is derived by taking the average of four normalized similarity values computed using known approaches. The final combined artist similarity function is computed as a weighted sum of the mood similarity and style similarity.

In the paper entitled *Boosting the Ranking Function Learning Process using Clustering*, *G. Giannopoulos, T. Dalamagas, M. Eirinaki* and *T. Sellis* examine the problem of how to increase the training input for ranking function learning systems without requiring more explicit or implicit user feedback. Acquiring adequate training data (in the form of relevance judgements for query-result pairs) to train a ranking function typically requires a long training period or involve a large number of users. The basic idea of the proposed approach is to expand the initial set of relevance judgements obtained from implicit user feedback by first clustering the search result documents based on their content similarity. After removing clusters that have low coherence in terms of the distribution of the relevance judgement of the cluster documents, each of the remaining clusters is labeled a relevance judgement based on the majority of the relevance judgements in the cluster. In this way, relevance judgements are estimated for documents that have not been viewed thereby increasing the set of training data.

Y. Sun, H. Li, I. Councill, J. Huang, W.-C. Lee and *C. Lee Giles* in their paper entitled *Personalized Ranking for Digital Libraries Based on Log Analysis* propose a personalized ranking method that is based on user preference models to improve the accuracy of predicting user actions. A user preference is modeled as a vector, termed the user preference vector, in the document feature space. The user preference vectors are obtained by training on implicit user feedback extracted from web log mining results, where each user feedback is represented by a document pair indicating that the user prefers the first document over the second one. The level of relevance of a document for a user is defined as the inner product of the document vector and the user preference vector. When a user submits a query to the system, the system first retrieves all the documents based on lexical similarity, and then re-ranks the documents based on the user's preference vector.

R. Pon, A. Cardenas and *D. Buttler* in their paper entitled *Online Selection of Parameters in the Rocchio Algorithm for Identifying Interesting News Articles* study how to dynamically adapt parameter values for the Rocchio algorithm to improve recommendation performance for a news articles filtering application. To enable more effective document filtering for different users, the authors propose an enhanced approach, termed *eRocchio*, where each incoming document is evaluated by multiple instantiations of the Rocchio formulation in parallel. Each Rocchio instantiation has its own unique weight parameter value and adaptive thesholder to optimize its corresponding instantiation. The best instantiation is then selected using a F-measure metric, and the recommendation outcome is used to adaptively update each instantiation.

In the final paper of the session, entitled *Supporting the Automatic Construction of Entity Aware Search Engines*, *L. Blanco, V. Crescenzi, P. Merialdo* and *P. Papotti* present a domain-independent approach to automatically search the web for pages that are publishing data about instances of a conceptual entity of interest. Given an input set of sample web pages from several distinct websites about some conceptual entity, the proposed approach first crawls these websites to collect more web pages about other instances of the conceptual entity. This is performed assuming that pages that describe different instances of a conceptual entity within a website share a common template. The system then automatically extracts a description of the entity using a set of keywords and launches web searches to look for new pages about the conceptual entity. By analyzing the returned web pages using the extracted entity description, new pages that represent the conceptual entity are identified and used to recursively trigger the search process.

Report on Workshop on Operating Systems Support for Next Generation Large Scale NVRAM (NVRAMOS 2009)

Sang-Won Lee Sooyong Kang Youjip Won Jongmoo Choi

Sungkyunkwan University
Suwon, Korea
wonlee@ece.skku.ac.kr

Hanyang University
Seoul, Korea
{sykang, yjwon}@hanyang.ac.kr

Dankook University
Yongin, Korea
choijm@dku.edu

1. MOTIVATIONS

NAND Flash memory solid state disk (hereafter SSD) technology is advancing rapidly in capacity and speed. Also, we envision that byte-addressable NVRAM devices, which is being thought as the future replacement of Flash memory technology will be commercially viable for storage within the next 3–7 years. Modern computer system takes hierarchical organization. It consists of CPU, Main Memory and Storage. This hierarchical organization is natural outcome of economic concern. Modern Operating System is designed to effectively exploit the physical characteristics of the device in each system hierarchy. Process management, memory management, and storage management subsystems are all designed to fill the gaps (speed and space) among the layers in different hierarchies while maximizing cost performance ratio. Current operating system paradigm draws clear line between memory and storage and handles them in very different way. Large scale byte-addressable NVRAM combines the physical characteristics of main memory with the non-volatility of storage, presenting new challenges for system designers. From the DBMS and Operating System's point of view, the advancement of this device calls for a reconsideration of legacy operating system architectures that have been designed based upon the notion of system hierarchy: CPU, memory and storage.

NVRAMOS Workshop (Workshop on Systems Support for large Scale NVRAM) is biannual event which invites leading experts from academia as well as industry in the area of large scale NVRAM, Flash Storage, Storage Class Memory, NVRAM device technology and related areas. The workshop specifically focuses on Operating Systems aspect of exploiting new semiconductor technology (FLASH, PRAM, MRAM, FRAM, Solid Electrolyte) as storage device or memory/storage device. The primary goal of this workshop is to form a community in this field and provide a forum where the experts in this area can put forth their vision, share views, exchange ideas and have intensive discussions on the technical challenges we may face (or are facing already) in the next several years. The first NVRAMOS workshop was held in Nov. 2007. NVRAMOS workshop now successfully positions itself as prime forum which achieves the original goal. NVRAMOS workshop has been two day event and has been allocating extensive time slot for each presentation so that speaker and the workshop participants can have interactive and intense technical discussion about the talk theme in in-depth manner.

2. WORKSHOP OUTLINE

NVRAMOS 2009 Spring workshop (http://www.dmclab.hanyang.ac.kr/nvramos09) consists of three technical sessions: *DAMO, invited talks and peer-reviewed paper presentations*. Different from past three NVRAMOS Workshops, workshop committee made new session which is dedicated for "casual and informal" discussion on cutting edge technical themes. It is called DAMO session.

2.1 DAMO Session

The DAMO session is a kind of *gong show or Birds-Of-a-Feather (BoF) session* to warm up the workshop and to attract the participants' attention. Each speaker in DAMO session is allocated 20 minute time slot. The presenter first gives a quick review of his/her Flash research field, what they are working on, and/or presents his/her view on future research direction and outstanding problems.

The first presentation, by Bumsoo Kim (CEO, Indilinx Co[1]., Korea), reviewed the SSD development history from the 1st generation to the current 3rd generation, predicted that the 4th generation SSD will appear in the market in one or two years. In particular, he predicted that the enterprise market (including OLTP (online transaction processing) applications) would be more promising than PC or laptop market because of the urgent IOPS (IO per seconds) requirements.

The second presentation, by Tae-Sun Chung (Ajou University, Korea), gave a survey on Flash translation layers (called *FTL*), and argued that database systems should be aware of the software stacks from database

[1] it is known for as a controller provider of the OCZ Vertex SSDs

module, file systems down to the lowest FTL layer for better performance.

The third presentation, Sang-Won Lee (Sungkyunkwan University), explained the importance of access density (i.e. IOPS per GB) in OLTP applications and argued that the intrinsic high access density makes OLTP applications one of the most promising killer applications for SSD. In addition, he pointed out that the Flash-aware buffer management policy needs to be investigated because of the asymmetric speed of read and write operations in Flash memory SSD. Another important issue pointed out is the performance variations in transaction throughput when SSD is used as storage device. In particular, depending on the adopted FTL techniques, a commercial SSD shows high fluctuations in performance while the other SSD does relatively uniform throughput. He finished his talk saying that one of the virtues of the enterprise storage devices is the predictable performance and it should be considered in designing FTL techniques.

The fourth presenter, Jaehyuk Cha (Hanyang University, Korea), stressed the importance of SSD benchmarks. Even though most SSD support the same host interface (e.g. SATA) with hard disks, there are several characteristics to make SSDs distinguishable from hard disks, and also there are diverse design spaces which can affect the performance characteristics of SSD. For this reason, the existing IO benchmarks developed mainly for comparing the hard disks are too plain to reveal the intrinsic performance differences among various SSDs. He emphasized that it is critical to develop new IO benchmarks tailored for SSD. The presenter recommended the uFlip benchmark [2] as a good starting point for further micro benchmark developments which can help expose the performance characteristics between different SSDs.

The last presentation, by Sooyong Kang (Hanyang University, Korea), discussed the write buffer cache issue on SSD controller, which FTL can exploit to improve the write performance. In particular, he raised a question of *whether it is possible to find an optimal write buffer replacement scheme*. Further, when NVRAM (for example, PRAM, FeRAM, and MRAM, which are expected to be commercially available in the market) is used as write buffer. Two specific issues are discussed: 1) whether there exists an optimal off-line buffer replacement algorithm?, and 2) if so, what parameters should be considered for algorithm evaluation? For the second issue, the presenter suggested that the status of log blocks managed by FTL and temporal locality in access patterns as well as the size of page clusters (i.e. basic unit of replacement) should be taken into account. The presentation concluded that NVRAM buffer and the traditional log blocks in FTL should be managed as an integrated write buffer cache.

2.2 Invited Presentations

The NVRAMOS 2009 Spring invited three speakers from both industry and academia. The talks are on SSD simulator, Flash memory-based database systems, and SSD performance model.

The first speaker, Eui-Young Chung (Yonsei University, Korea), introduced their effort to develop a SSD simulator which is based on the Cycle-Accurate Model. The simulator not only explored the internal architecture of SSD, but also considered both the H/W (Channel/Way, DRAM cache management, Hybrid Flash memory arrays such as MLC+SLC combination, etc.) and S/W (FTL) features of SSD, simultaneously. During the discussion, many audience members suggested to conduct the simulation using various access patterns that database systems incur and compare the results with those of the previous works about SSD performance in the database systems, which can help not only evaluating the accuracy of the black-box approach but also finding out the bottleneck point inside SSD when it is used in the database system.

The second speaker, Bongki Moon (University of Arizona, USA), presented challenges and opportunities of Flash memory database systems. While the current SSD products still have a random scattered writes issue, Moon said that the SSD can provide a few immediate benefits for some DB operations: 1) reduce commit-time delay by fast logging, 2) reduce read time for multi-versioned data, and 3) Flash-friendly I/O patterns in temporary table spaces. To remedy the random scattered writes issue, he said that besides industries' efforts to increase the random write performance from storage systems' viewpoint, a great deal of efforts need to be also solicited from database systems' point of view. He briefly introduced In-Page Logging (IPL) approach [1] as an example. During the discussion, some audience members argued that two key assumptions in the IPL approach, 1) multiple writes are possible within a page, and 2) non-sequential page writes in a data block, are not valid any more. A participant from from industry added that these assumptions are difficult to accept not only in the current SSDs but (probably) also in the future SSDs, because the characteristics of Flash memory chip (e.g. the allowed number of partial writes per page), are getting worse as its capacity increases.

The last invited speaker, Dongjun Shin (Samsung Electronics), introduced an interesting SSD performance model, which is based on the well-known RISC instruction pipelining performance model. He argued that, because there is parallelism between computation and I/O in SSD, SSD operations can also be pipelined, which led his team to revisit the old model. Based on their experiment, the accuracy of the model was proven to be quite reliable. In modern SSDs, micro-controller issues I/O instructions in pipelined fashion, and there can be multiple outstanding instructions each of which accesses different NAND Flash

chips within SSD. Using the simulation model, they found an interesting fact that firmware overhead, which has been regarded as negligible, becomes rather significant especially when microcontroller of SSD works in deeply pipelined fashion. Their model showed that the performance of random I/O operation in SSD critically relies on the firmware efficiency and is currently being bound by firmware overhead. The audience agreed that it would be very challenging to design application-specific SSDs (e.g. DBMS-aware SSD); especially when we are to migrate part of functionalities of database's storage manager down to SSD controller, which generally increases the complexity of firmware.

2.3 Paper Presentations

Four accepted papers have been presented in the paper session. Two of them deal with the characteristics and performance issues of SSD, while the others explore the impacts of Storage Class Memory (hereafter SCM) on file systems, databases, and operating systems. SCM, also known as Non-volatile RAM (NVRAM) such as Ferroelectric RAM (FeRAM), Magnetoresistive RAM (MRAM), and Phase-change RAM (PRAM) has characteristics of both random accessibility and non-volatility [5]. It can be used not only as main memory for dynamic program execution but also as secondary storage for file systems.

The first presentation, by Youjip Won and Minsuk Choi (Hanyang University, Korea), evaluated the performance characteristics of various SSDs under a diversity of file systems and I/O workloads. Recently, SSDs have started to replace hard disks in laptop computer and even in server storage [3]. These trends raise an interesting question: "Are the existing file systems and I/O workloads primarily optimized for hard disks still valid for SSDs?" To answer the question, the authors conducted a quantitative analysis: 1) using three different SSDs, namely Intel X25-M, Samsung MMCRE64G5MPP, and Mtron MSD-SATA3525, and 2) mounting three different file systems, namely EXT3, REISER, XFS, NILFS, and FAT32 on the SSDs, and 3) applying four different I/O workloads with various file/record sizes, and 4) measuring throughput for each combinational case. Experimental results showed that SSD performance is very sensitive to I/O workloads and file system, and there is much room to develop efficient SSD-aware algorithms both for file systems and FTLs. The results triggered active discussions among the audience and some strange results were clarified, which were derived from the special features of FTL in SSDs and/or from the anomalous behaviors of file systems in Linux. Also, the audience agreed that the characteristics of SSDs are quite different from those of hard disks, which requires new studies for file systems and I/O workloads appropriate to SSDs.

The second presentation, by Junkil Ryu and Chanik Park (POSTECH, Korea), devised a black box approach to investigate the internal structures and algorithms inside SSDs. The performance of SSD heavily depends on the internal structures such as bus interleaving, chip interleaving, and the size of DRAM write buffer, and the FTL algorithms, e.g. mapping scheme and a garbage collection policy [4]. By applying the devised black-box approach, the authors retrieved an important performance parameter, called SSD management block, from commercially available SSD products, which can then be used to optimally configure OS policies in buffer cache and prefetching. During the discussion, the distinction and similarity of performance model between hard disks and SSDs were elaborated. Some participants described that the degree of parallelism and garbage collection policy are the most critical parameters in SSD as the seek time in hard disks.

The third presentation, by In Hwan Doh, Young Jin Kim, Eunsam Kim, Sam H. Noh (Hongik University), and Donghee Lee (University of Seoul), exploited Storage Class Memory as a reliable write buffer and analyzed the impacts of SCM on the performance and dependability of computer systems. With the SCM write buffer, the authors enhanced system reliability by providing transactional supports and improved system performance by applying delayed writes and reducing write requests to the disks. The audience suggested various ideas about how to make use of SCM to resolve the traditional problems of databases and file systems such as small random writes handling, intent logging and recovery, and data-intensive application supports.

The final presentation, by Seungjae Baek and Jongmoo Choi (Dankook University), examined how to apply SCM on embedded systems. They conjectured that there are three possible organizations in using SCM in the legacy computer organization, namely SCM as storage, SCM as main memory, and SCM as both. Their conjecture is in accordance with those suggested by M. K. Qureshi et al. [6], but the former mainly focused on the SCM as both organization, while the latter concentrating on the SCM as main memory one. The authors proposed a novel operating system for SCM as both organizations, which contain a new SCM manager, supporting not only file interfaces but also memory interfaces, simultaneously. In other words, file objects and memory objects are co-located in SCM, which makes it feasible to exchange between the objects without copy overheads. Some of the audiences recommended other possible organizations such as hybrid storage with SCM and SSD, and discussed tradeoffs among them in terms of energy, cost, and performance.

3. WORKSHOP CONCLUSIONS

Many participants agreed that the research on Flash memory SSD and its related system software technologies is one of the main-stream topics in current computer science field. Some participants carefully envision that SCM will be a next wave for storage when the large scale manufacturing technology is possible.

3.1 Implications on Database Researches

From the workshop presentations and discussions, we can learn several implications which Flash memory SSD and next generation NVRAM can have on database fields, and we summarize them in the followings:

First of all, despite the tardy "erase-before-overwrite" characteristics of Flash memory, the modern third generation high-end SSDs, relentlessly and drastically improved in the last few years and now are fast enough to compete with the enterprise class hard disks in terms of performance and even the cost effectiveness in OLTP applications. Moreover, among many potential applications for SSD, online transaction processing (OLTP) is considered as one of the most promising ones because its IO patterns are mainly comprised of random reads and writes, for which SSDs can considerably outperform hard disks. In particular, the database community has two popular macro benchmarks, TPC-C and TPC-H, both of which are very IO-intensive. In addition, the IO patterns from the benchmarks are further complicated by concurrently executing processes. Therefore, they could be good testbeds for comparing different SSDs. These macro benchmarks, in combination with micro benchmarks such as uFlip, could be more realistic than the traditional IO benchmarks for hard disks.

Second, if we take into account the rapid growth of SSD market, it is time to revisit all the major IO-related database techniques, including storage layout, join algorithms, buffer management, and index technology, from the perspectives of SSD, because most of them has assumed the hard disks as its secondary storage [7]. For example, while a certain algorithm could choose sequential scans when it is expected to work on hard disks, index-based version of the algorithm could be much better on SSDs. One thing to note is that the availability of fast SSD does not mean that database side optimizations such as in-page logging [1] are not valid or necessary any more. Instead, there would be many opportunities from database communities. In fact, since 2008, many diverse works on Flash-based DBMSs are appearing in major database conferences and workshops, including SIGMOD, VLDB, ICDE and Damon.

Finally, there are a few opportunities for DBMS architectural changes to properly exploit moderate size NVRAM (e.g. of several tens or hundreds megabytes). One example is to write the log tail directly and persistently in SCM, instead of writing the log buffer in DRAM and flushing it to the hard disk or Flash memory whenever every commit command is issued. By doing this, the response time and throughput can be improved. Another example is to exploit the NVRAM inside SSD controller. The FTL module needs to keep the mapping information between logical and physical address either in the unit of block or page. When the mapping information is changed, the update should be persistently stored in Flash memory. This can be a performance bottleneck in SSD. However, this overhead can be significantly relegated if we manage the mapping information (usually less than hundred megabytes) in NVRAM.

4. ACKNOWLEDGMENTS

NVRAMOS Workshops since its inception in Nov. 2007 have been sponsored by National Research Lab Grant(ROA 2007-000-20114-0) from Korean Science and Engineering Foundation(KOSEF).

5. REFERENCES

[1] Sang-Won Lee and Bongki Moon, Design of FlashFlash-based DBMS: An In-Page Logging Approach, Proceedings of the ACM SIGMOD conference, June, 2007

[2] Luc Bouganim, Bjorn Tor Jonsson, and Philippe Bonnet uFLIP: Understanding FlashFlash IO Patterns, CIDR 2009, January, 2009

[3] D. Narayanan, E. Thereska, A. Donnelly, S. Elinikety, and A. Rowstron, "Migrating Server Storage to SSDs: Analysis of Tradeoffs", Proceeding of the ACM EuroSys Conference, March 31-April 3, 2009.

[4] N. Agrawal, V. Prabhakaran, T. Wobber, J. D. Davis, M. Manasse, and R. Panigrahy, "Design Tradeoffs for SSD Performance", Proceedings of the 2008 USENIX Annual Technical Conference, pages 57–70, June 22-27, 2008.

[5] G. W. Burr, B. N. Kurdi, J. C. Scott, C. H. Lam, K. Gopalakrishnan, and R. S. Shenoy. "Overview of Candidate Device Technologies for Storage-Class Memory", IBM Journal of Research and Development, 52(4):449–464, 2008.

[6] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable High Performance Main Memory System Using Phase-Change Memory Technology", Proceedings of the 36th International Symposium on Computer Architecture (ISCA 2009), June 20-24, 2009.

[7] Sang-Won Lee and Won Kim, On Flash-based DBMS: Issues for Architectural Re-examination, Journal of Object Technology, September, 2007

Workshop on Theory and Practice of Provenance

Event Report

James Cheney
University of Edinburgh
jcheney@inf.ed.ac.uk

Abstract

Provenance, or metadata about the creation, influences upon, or other history of objects or data, has attracted attention in a wide variety of contexts in computer science over the last few years. This event report describes a recent workshop on "Theory and Practice of Provenance", intended as a forum for presenting novel ideas about provenance and encouraging interaction among provenance researchers.

1. Introduction

Provenance is metadata about the creation, influences upon, and other history of objects or data. In computer systems, such metadata is often used for security and profiling, and is essential for making informed judgments about data quality, integrity, and authenticity. Recent research in a variety of settings (databases and data warehouses, geographic information systems, scientific workflows, grid computing, and the Semantic Web) has begun addressing the problem of recording and exploiting provenance. In addition, forms of provenance are now used in several areas of advanced computer systems research such as probabilistic databases, incremental computation, information-flow security, file synchronization, and annotation management systems. Other topics, such as version control and archiving, may also benefit from better understanding of provenance.

However, provenance research has so far been carried out in relative isolation in a number of disciplines of computer science, such as databases, scientific computation, systems, and security; moreover, topics such as information flow security [5], dependence analysis in programming languages [2], causal models in artificial intelligence [4], and traceability in software engineering appear related, yet there has been little investigation of applications of this work to provenance. We believe that more meaningful interaction between the theory and systems communities, between academia and industry, and between different branches of computer science is needed to make progress on understanding and implementing reliable, efficient and scalable provenance management in computer systems.

The First Workshop on *Theory and Applications of Provenance* (TaPP) was held on February 23, 2009 in San Francisco, California, just before the 2009 USENIX Conference on *File and Storage Technology* (FAST 2009). The goals of the workshop included presenting cross-disciplinary, foundational and highly speculative research, raising the profile of provenance research, and increasing interaction between provenance researchers in several subdisciplines, the broader systems community, and industry. We believe that TaPP made significant contributions to all of these goals.

TaPP had a lightweight, but formal peer-review process with a nine-member program committee representing a variety of backgrounds, including databases, scientific workflow computation, software engineering, security, and systems. TaPP attracted 22 submissions, comprising 13 long papers and 9 short papers. Submissions were reviewed by at least three members of the program committee. Five long papers were selected for long presentations, and another five long papers and three short papers were selected for short presentations. This led to a crowded, but lively schedule.

The workshop attracted 38 participants (including speakers and organizers), representing a healthy mixture of academic and industrial backgrounds. TaPP was fortunate to be able to invite two excellent speakers using support from eSI. They were Joe Halpern (Cornell University) and Margo Seltzer (Harvard University). In the rest of this event report, we list the presentations and abstracts, and where appropriate discuss additional highlights of the workshop, particularly discussions following the invited talks.

TaPP 2009 was supported by the United Kingdom eScience Institute (eSI) through a Theme Program on "Principles of Provenance", and significant organizational support was provided by USENIX. USENIX has also indicated enthusiasm for supporting future iterations of the workshop. The proceedings of TaPP are published online by USENIX [3], and a complementary event report that provides further details about the audience questions and discussion has been published in the USENIX newsletter [1].

2. Contributions

2.1 Session 1: Joe Halpern, Cornell University

Professor Halpern's talk, entitled "Causality, Responsibility, and Blame: A Structural-Model Approach" focused on

mathematical models of concepts such as *causality*. The concept of causality has troubled philosophers and scientists for hundreds of years, and scientists have been wary of the concept because of Hume's critical analysis of causality as a subjective (or psychological) rather than objective phenomenon. Halpern, together with Pearl, Chockler and Kupferman, has made valuable contributions towards mathematically characterizing the concepts of *cause* and *actual cause*. We invited Professor Halpern based on our belief that mathematical models of causality are important for understanding provenance, especially forms of provenance based on intuitions such as "cause", "dependence", or "influence". We believe the lively discussion during and after the talk confirms our belief that the provenance community can benefit from learning more about the models of causality developed by Halpern and Pearl [4].

2.2 Session 2: Provenance and Security

A Formal Model of Provenance in Distributed Systems. Issam Souilah, University of Southampton, UK; *Adrian Francalanza*, University of Malta, Malta; Vladimiro Sassone, University of Southampton, UK

Abstract: We present a formalism for provenance in distributed systems based on the π-calculus. Its main feature is that all data products are annotated with metadata representing their provenance. The calculus is given a provenance tracking semantics, which ensures that data provenance is updated as the computation proceeds. The calculus also enjoys a pattern-restricted input primitive which allows processes to decide what data to receive and what branch of computation to proceed with based on the provenance information of data. We give examples to illustrate the use of the calculus and discuss some of the semantic properties of our provenance notion. We conclude by reviewing related work and discussing directions for future research.

Towards Semantics for Provenance Security. *Stephen Chong*, Harvard University

Abstract: Provenance records the history of data. Careless use of provenance may violate the security policies of data. Moreover, the provenance itself may be sensitive information, necessitating restrictions on the use of both data and provenance to enforce security requirements. This paper proposes extensional semantic definitions for provenance security. The semantic definitions require that provenance information released to the user does not reveal confidential data, and that neither the provenance information given to the user, nor the programs output, reveal sensitive provenance information.

Scalable Access Controls for Lineage. Arnon Rosenthal, Len Seligman, *Adriane Chapman*, and Barbara Blaustein, The MITRE Corporation

Abstract: Lineage stores often contain sensitive information that needs protection from unauthorized access. We build on prior work for security and privacy of lineage information, focusing on complex conditions and scalable administration. We use Attribute-Based Access Control (ABAC) to express conditions based on many attributes, instead of roles. We then make administration and management more scalable, instead of managing large, monolithic access predicates for each object. To do so, we first support modular traceability and maintainability for separate concerns (e.g. security, legally mandated privacy, organizationally mandated privacy). We then provide constructs to manage authority when multiple administrators must collaborate. We show that these security techniques are needed for easy lineage security administration.

2.3 Session 3: Provenance for Web, Grid and Digital Libraries

On Explicit Provenance Management in RDF/S. Graphs P. Pediaditis, *G. Flouris*, I. Fundulaki, and V. Christophides, ICS-FORTH

Abstract: The notion of RDF Named Graphs has been proposed in order to assign provenance information to data described using RDF triples. In this paper, we argue that named graphs alone cannot capture provenance information in the presence of RDFS reasoning and updates. In order to address this problem, we introduce the notion of RDF/S Graphsets: a graphset is associated with a set of RDF named graphs and contain the triples that are jointly owned by the named graphs that constitute the graphset. We formalize the notions of RDF named graphs and RDF/S graphsets and propose query and update languages that can be used to handle provenance information for RDF/S graphs taking into account RDFS semantics.

Application of Named Graphs Towards Custom Provenance Views. *Tara Gibson*, Karen Schuchardt, and Eric Stephan, Pacific Northwest National Laboratory

Abstract: Provenance capture as applied to execution oriented and interactive workflows is designed to record minute detail needed to support a "modify and restart" paradigm as well as re-execution of past workflows. In our experience, provenance also plays an important role in human-centered verification, results tracking, and knowledge sharing. However, the amount of information recorded by provenance capture mechanisms generally obfuscates the conceptual view of events. There is a need for a flexible means to create and dynamically control user oriented views over the detailed provenance record. In this paper, we present a design which leverages named graphs and extensions to the SPARQL query language to create and manage views as a server-side function, simplifying user presentation of provenance data.

Authenticity and Provenance in Long Term Digital Preservation: Modeling and Implementation in Preservation Aware Storage. *Michael Factor*, Ealan Henis, Dalit Naor, Simona Rabinovici-Cohen, Petra Reshef, and Shahar Ronen, IBM Research Lab in Haifa, Israel; Giovanni Michetti and Maria Guercio, University of Urbino, Italy

Abstract: A growing number of digital objects are desig-

nated for long term preservation - a time scale during which technologies, formats and communities are very likely to change. Specialized approaches, models and technologies are needed to guarantee the long-term understandability of the preserved data. Maintaining the authenticity (trustworthiness) and provenance (history of creation, ownership, accesses and changes) of the preserved objects for the long term is of great importance, since users must be confident that the objects in the changed environment are authentic. We present a novel model for managing authenticity in long term digital preservation systems and a supporting archival storage component. The model and archival storage build on OAIS, the leading standard in the area of long-term digital preservation. The preservation aware storage layer handles provenance data, and documents the relevant events. It collocates provenance data (and other metadata) together with the preserved data in a secure environment, thus enhancing the chances of their co-survival. Handling authenticity and provenance at the storage layer reduces both threats to authenticity and computation times. This work addresses core issues in long-term digital preservation in a novel and practical manner. We present an example of managing authenticity of data objects during data transformation at the storage component.

Steps Toward Managing Lineage Metadata in Grid Clusters. *Ashish Gehani* and Minyoung Kim, SRI International; Jian Zhang, Louisiana State University

Abstract: The lineage of a piece of data is of utility to a wide range of domains. Several application-specific extensions have been built to facilitate tracking the origin of the output that the software produces. In the quest to provide such support to extant programs, efforts have been recently made to develop operating system functionality for auditing filesystem activity to infer lineage relationships. We report on our exploration of mechanisms to manage the lineage metadata in Grid clusters.

2.4 Session 4: Margo Seltzer, Harvard University

Professor Seltzer spoke on the topic "The State of Provenance in 2019". She gave a talk prepared for the 11th Workshop on Theory and Practice of Provenance, which she expects to take place ten years in the future. Her talk envisages a world in which provenance research has borne fruit in many areas of computer science — provenance is part of all storage systems, databases, and applications; provenance is secure and queryable, and search engines such as "Poogle" offer provenance-aware informational retrieval. Professor Seltzer then outlined the development of this state of affairs, from the work that has already (really) been done in 2009, to her predicted view in 2019. She predicts that the main thrusts of this development will require major progress on *storing and querying* provenance, *standards* for transmitting and integrating provenance (encompassing both file systems and network protocols), and *formalism* or theoretical understanding of provenance. Furthermore, security concerns will come to dominate as provenance becomes essential for accountability for sensitive data, and once the technology has been developed, significant effort will also be needed to establish standards and governmental policy that will legislate good provenance practices.

We look forward to hearing an updated version of this talk in 2019.

2.5 Session 5: Provenance Systems I

A Framework for Fine-grained Data Integration and Curation, with Provenance, in a Dataspace. *David W. Archer*, Lois M.L. Delcambre, and David Maier, Portland State University

Abstract: Some tasks in a dataspace (a loose collection of heterogeneous data sources) require integration of fine-grained data from diverse sources. This work is often done by end users knowledgeable about the domain, who copy-and-paste data into a spreadsheet or other existing application. Inspired by this kind of work, in this paper we define a data curation setting characterized by data that are explicitly selected, copied, and then pasted into a target dataset where they can be confirmed or replaced. Rows and columns in the target may also be combined, for example, when redundant. Each of these actions is an integration decision, often of high quality, that when taken together comprise the provenance of a data value in the target. In this paper, we define a conceptual model for data and provenance for these user actions, and we show how questions about data provenance can be answered. We note that our model can be used in automated data curation as well as in a setting with the manual activity we emphasize in our examples.

The Case for Browser Provenance. *Daniel W. Margo* and Margo Seltzer, Harvard University

Abstract: In our increasingly networked world, web browsers are important applications. Originally an interface tool for accessing distributed documents, browsers have become ubiquitous, incorporating a significant portion of user interaction. A modern browser now also reads email, plays media, edits documents, and runs applications. Consequently, browsers process large quantities of data, and must record metadata, such as history, to help users manage their data. Most of the metadata that modern browsers record is actually provenance - metadata that captures the causality and lineage of data obtained via the browser. We demonstrate that characterizing browser metadata as provenance and then applying techniques from the provenance research community enables new browser functionality. For example, provenance can improve both history and web search by indicating contextual and personal relationships between data items. Users can also answer complex questions about the origins of their data by querying provenance. Our initial results suggest these features are feasible to implement and could perform well in modern browsers.

Provenance as Data Mining: Combining File System Metadata with Content Analysis. *Vinay Deolalikar* and

Hernan Laffitte, Hewlett Packard Labs
Abstract: Provenance describes how an object came to be in its present state. Thus, it describes the evolution of the object over time. Prior work on provenance has focused on databases and the file system. The database or file system is enhanced or augmented in order to capture additional information about the historical evolution of document collections, and thus answer the provenance question. We address the question of provenance for unstructured information (i.e., document corpora from file systems) but without any enhancements to the file system. To provide a solution in this setting, we model the provenance problem in such a setting as a problem of data mining. We show that data mining can provide provenance information for repositories of unstructured information, including chains of historical evolution. Thus, we do not require any additions to the file system, and we can operate on legacy documents. Experimental results indicate a strong performance of our approach.

This presentation led to an interesting discussion on how to objectively evaluate such "provenance mining" techniques, given that we cannot easily construct "gold standard" test data in the absence of systems that already record provenance.

2.6 Session 6: Provenance Systems II

Story Book: An Efficient Extensible Provenance Framework. *R. Spillane*, Stony Brook University; R. Sears, University of California, Berkeley; C. Yalamanchili, S. Gaikwad, M. Chinni, and E. Zadok, Stony Brook University
Abstract: Most application provenance systems are hard coded for a particular type of system or data, while current provenance file systems maintain in-memory provenance graphs and reside in kernel space, leading to complex and constrained implementations. Story Book resides in user space, and treats provenance events as a generic event log, leading to a simple, flexible and easily optimized system.

We demonstrate the flexibility of our design by adding provenance to a number of different systems, including a file system, database and a number of file types, and by implementing two separate storage backends. Although Story Book is nearly 2.5 times slower than ext3 under worst case workloads, this is mostly due to FUSE message passing overhead. Our experiments show that coupling our simple design with existing storage optimizations provides higher throughput than existing systems.

Making a Cloud Provenance-Aware. *Kiran-Kumar Muniswamy-Reddy*, Peter Macko, and Margo Seltzer, Harvard University
Abstract: The advent of cloud computing provides a cheap and convenient mechanism for scientists to share data. The utility of such data is obviously enhanced when the provenance of the data is also available. The cloud, while convenient for storing data, is not designed for storing and querying provenance. In this paper, we present desirable properties for distributed provenance storage systems and present design alternatives for storing data and provenance on Amazon's popular Web Services platform (AWS). We evaluate the properties satisfied by each approach and analyze the cost of storing and querying provenance in each approach.

Transparently Gathering Provenance with Provenance Aware Condor. *Christine F. Reilly* and Jeffrey F. Naughton, University of Wisconsin-Madison
Abstract: We observed that the Condor batch execution system exposes a lot of information about the jobs that run in the system. This observation led us to explore whether this system information could be used for provenance. The result of our explorations is Provenance Aware Condor (PAC), a system that transparently gathers provenance while jobs run in Condor. Transparent provenance gathering requires that the application not be altered in order to run in the provenance system. This requirement allows any application that can run in Condor to also run in PAC. Through SQL queries, PAC is able to answer a wide range of questions about the files used by a job and the machines that execute jobs.

3. Conclusions

We believe that there is a clear need for a combination of both new theoretical insights and new systems research in tackling the many challenges of data provenance. The Theory and Practice of Provenance workshop series will be supported by USENIX and will, we hope, develop into a forum for encouraging, recognizing and disseminating great new ideas in this area.

Acknowledgments We would like to gratefully acknowledge the USENIX staff for their support in organizing TaPP 2009, particularly Casey Henderson, Jane-Ellen Long, Devon Shaw, and Ellie Young.

References

[1] Event report: First workshop on theory and practice of provenance (TaPP '09). *;login: The USENIX Magazine*, 34(4):84–90, June 2009. Available online at: `http://www.usenix.org/publications/login/2009-06/`.

[2] Martín Abadi, Anindya Banerjee, Nevin Heintze, and Jon G. Riecke. A core calculus of dependency. In *POPL*, pages 147–160. ACM Press, 1999.

[3] James Cheney, editor. *First Workshop on the Theory and Practice of Provenance, February 23, 2009, San Francisco, CA, USA, Proceedings*. USENIX, 2009. `http://-www.usenix.org/event/tapp09/tech/`.

[4] J.Y. Halpern and J. Pearl. Causes and explanations: A structural-model approach–part I: Causes. *British J. Philos. Sci.*, 56:843–887, 2005.

[5] Andrei Sabelfeld and Andrew Myers. Language-based information-flow security. *IEEE Journal on Selected Areas in Communications*, 21(1):5–19, 2003.

Call for Papers: SIGMOD / PODS Conference

June 6-11, 2010, Indianapolis, Indiana, USA

http://www.sigmod2010.org

2010 ACM SIGMOD International Conference on Management of Data

The annual ACM SIGMOD conference is a leading international forum for database researchers, practitioners, developers, and users to explore cutting-edge ideas and results, and to exchange techniques, tools, and experiences. We invite the submission of original research contributions and industrial papers, as well as proposals for demonstrations, tutorials, and panels. We encourage submissions relating to all aspects of data management defined broadly, and particularly encourage work on topics of emerging interest in the research and development communities.

TOPICS OF INTEREST

General areas of interests include but are not limited to the following:

- New programming models and languages to extend the DBMS ecosystem beyond traditional data and query models
- Scalable Data Management on Cloud Computing Infrastructures
- Data-centric approaches for effective management of large-scale data-centers
- Innovative models, architectures, and algorithms for distributed and parallel data analytics
- Energy-efficiency and power management issues in large-scale data processing environments
- Database Management System and Algorithm Designs for emerging harwdware architectures: Multi-core processors, larger on-chip caches, large inexpensive RAM, and flash memory
- Data management applications (e.g., Web mashups, social networks, scientific databases, sensor networks)
- Performance and scalability (e.g., indexing, hardware accelerators)
- Application of Machine-learning and Data-mining techniques for autonomic database systems
- Other aspects of modern information systems such as security, privacy, personalization, user interfaces, etc.

IMPORTANT DATES

October 29, 2009: Abstract submission (research papers only)
November 5, 2009: Manuscript submission (research papers, industrial papers, demonstration proposals)
December 3, 2009: Tutorial and Panel proposals submission
February 15, 2010: Notification of acceptance
March 15, 2010: Final camera-ready papers due

SUBMISSION

All aspects of the submission and notification process will be handled electronically. Detailed submission instructions will be posted on the conference website. As has become the tradition for SIGMOD, research papers will be judged for quality and relevance through double-blind reviewing, where the identities of the authors are withheld from the reviewers. Thus, author names and affiliations must not appear in the paper, and bibliographic references must be adjusted to preserve author anonymity. Details on anonymity requirements are available on the web page.

ORGANIZATION

General Chair	Ahmed Elmagarmid (Purdue University)
Program Chair	Divyakant Agrawal (University of California at Santa Barbara)
Finance Chair	Marianne Winslett (University of Illinois at Urbana Champagne)
Publicity Chairs	Sourav Bhowmick (NTU Singapore) Kristen LeFevre (University of Michigan)
Proceedings Chair	Mohammed Mokbel (University of Minnesota)
Workshops Chair	Christian Jensen (Aalborg University)
Industrial Program Chair	Berthold Reinwald (IBM Almaden)
Tutorials Chair	Bernhard Mitschang (Universitat Stuttgart)
Keynotes & Panels Chair	Alon Halevy (Google, Inc.)
Demonstration Chair	Yufei Tao (Chinese University of Hong Kong)
Exhibits Chair	Luna Dong (AT&T Research)
New Researcher	Christoph Koch (Cornell)
Symposium Chairs	Yanlei Dao (University of Massachusetts)
Local Arrangements	Beth Plale (Indiana University) Barbara Fossum (Purdue University)
Local Workshops Chair	Melanie Wu (Indiana University)
Registration Chair	Hakan Ferhatosmanoglu (Ohio State University)
Undergraduate Research Symposium Chair	Sihem Amer-Yahia (Yahoo, Inc.)
New Initiatives Committee	Amr El Abbadi (University of California at Santa Barbara) Jeffrey F. Naughton (University of Wisconsin at Madison) Jennifer Widom (Stanford University)
Demonstrations Local Arrangements Chair	Brandeis H. Marshall (Purdue University)
Web / Information Chair	Wai Gen Yee (Illinois Institute of Technology)

PROGRAM COMMITTEE

Group Leaders

Gustavo Alonso (ETH Zurich)
Selcuk Candan (Arizona State University)
Graham Cormode (AT&T)
Susan Davidson (University of Pennsylvania)
Nick Koudas (University of Toronto)
Chen Li (University of California at Irvine)
David Lomet (Microsoft Research)
Sunil Prabhakar (Purdue University)
Evaggelia Pitoura (University of Ioannina)
Rajeev Rastogi (Yahoo Research India)
Timos Sellis (University of Athens)
Cyrus Shahabi (University of Southern California)

Committee Members

Daniel Abadi (Yale University)
Walid Aref (Purdue University)
Magdalena Balazinska (University of Washington)
Wolf-Tilo Balke (University of Hannover)
Abraham Bernstein (University of Zurich)

Phil Bernstein (Microsoft Research)
Peter Boncz (Centrum Wiskunde & Informatica)
Angela Bonifati (ICAR-CNR)
Fabio Casati (University of Trento)
Tiziana Catarci (University of Roma)
Stefano Ceri (Politecnico di Milano)
Kaushik Chakrabarti (Microsoft Research)
Chee Yong Chan (National University of Singapore)
Lei Chen (Hong Kong University of Science and Technology)
Yi Chen (Arizona State University)
Laura Chiticariu (IBM Almaden)
Panos Chrysanthis (University of Pittsburgh)
Chris Clifton (Purdue University)
Sabrina De Capiatani de Vimerati (University of Milan)
Amol Deshpande (University of Maryland)
Alin Dobra (University of Florida)
Christos Faloutsos (Carnegie-Melon University)
Alan Fekete (University of Sydney)
Phil Gibbons (Intel Research)
Oliver Guenther (Humboldt University)
Sudipto Guha (University of Pennsylvania)
Hakan Hacigumus (NEC Research Laboratories of America)
Marios Hadjieleftherin (AT&T Shannon Labs)
Theo Haerder (University of Kaiserslautern)
Stavros Harizopoulos (HP Labs)
Wynne Hsu (National University of Singapore)
Christopher Jermaine (Rice University)
Theodore Johnson (AT&T Research)
Ben Kao (University of Hong Kong)
Daniel Kifer (Pennsylvania State University)
Christoph Koch (Cornell University)
Yannis Kotidis (Athens University of Economics and Business)
Hans-Peter Kriegel (Ludwig-Maximilians-Universitat Munchen)
Laks V.S. Lakshmanan (University of British Columbia)
Tan Kian Lee (National University of Singapore)
Qiong Luo (Hong Kong University of Science & Technology)
Stefan Manegold (CWI: Centrum Wiskunde & Informatica)
Sharad Mehrotra (University of California at Irvine)
Ahmed Metwally (Google, Inc.)
Bongki Moon (University of Arizona)
Shinichi Morishita (University of Tokyo)
Kyriakos Mouratidis (Singapore Management University)
Suman Nath (Microsoft Research)
Wolfgang Nejdl (University of Hanover)
Raymond Ng (University of British Columbia)
Christopher Olston (Yahoo! Research)
Mourad Ouzzani (Purdue University)
Fatma Ozcan (IBM Almaden)
Wenfei Pan (University of Edinburgh)

Hwee Hwa Pang (Singapore Management University)
Yannis Papakonstantinou (University of California at San Diego)
Jian Pei (Simon Fraser University)
Lin Qiao (IBM Alamden)
Mirek Riedwald (Northeastern University)
Uwe Rohm (University of Sydney)
Kenneth Salem (University of Waterloo)
Pierangele Samarati (University of Milan)
Maria Luisa Sapino (Universite di Torino)
Liuba Shrira (Brandeis University)
Mehul Shah (HP Laboratories)
Alkis Simitsis (HP Labs)
Ambuj Singh (University of California at Santa Barbara)
Yannis Sismanis (IBM Almaden Tresearch)
Il-Yeol Song (Drexel University)
Nisheeth Srivastava (Bell Laboratories)
Torsten Suel (Polytechnic Institute of NYU)
Wang-Chiew Tan (University of California at Santa Cruz)
Nesime Tatbul (ETH Zurich)
Jun Tatemura (NEC Research Laboratories of America)
Gottfried Vossen (University of Muenster)
Haixun Wang (Microsoft Research Asia)
Min Wang (IBM T. J. Watson Research Center)
Wei Wang (University of North Carolina at Chapel Hill)
Dong Xin (Microsoft Research)
Haruo Y Yokota (Tokyo Institute of Technology)
Jeffrey Xu Yu (Chinese University of Hong Kong)
Carlo Zaniolo (University of California at Los Angeles)
Aidong Zhang (SUNY Buffalo)

29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)

The PODS symposium series, held in conjunction with the SIGMOD conference series, provides a premier annual forum for the communication of new advances in the theoretical foundations of database systems. For the 29th edition, original research papers providing new insights in the specification, design, or implementation of data-management tools are called for. We especially welcome papers addressing such insights in emerging database environments and applications. Topics that fit the interests of the symposium include the following (as they pertain to databases):

TOPICS OF INTEREST

Topics that fit the interests of the symposium include the following (as they pertain to databases):

- languages for semi-structured data (including XML and RDF);
- search query languages (including techniques from information retrieval);
- distributed and parallel aspects of databases (including cloud computing);
- dynamic aspects of databases (updates, views, real-time and sensor data, approximate query answering, data streams);
- incompleteness, inconsistency, and uncertainty in databases;
- schema and query extraction;
- data integration;
- data exchange;
- provenance;
- workflows;
- metadata management;
- meta-querying;
- semantic-Web data and ontologies;
- data mining and machine learning techniques for databases;
- constraints (specification, reasoning, mining, constraint databases);
- privacy and security;
- Web services;
- model theory, logics, algebras and computational complexity;
- data modeling;
- data structures and algorithms for data management;
- design, semantics, and optimization of query and database languages;
- domain-specific databases (multi-media, scientific, spatial, temporal, text).

IMPORTANT DATES

December 4, 2009: Abstract submission
December 11, 2009: Manuscript submission
March 3, 2010: Notification of acceptance
April 2, 2010: Final camera-ready papers due

SUBMISSION

All aspects of the submission and notification process will be handled electronically. Detailed submission instructions will be posted on the conference website. Note that, unlike the SIGMOD conference, PODS does not use double-blind reviewing, and therefore PODS submissions should be eponymous (i.e., the names and affiliations of authors should be listed on the paper). The results must be unpublished and not

submitted elsewhere, including the formal proceedings of other symposia or workshops. Authors of an accepted paper will be expected to sign copyright release forms, and one author is expected to present it at the conference.

ORGANIZATION

General Chair	Jan Paredaens (University of Antwerp)
Program Chair	Dirk Van Gucht (Indiana University)
Publicity Chair	Yuqing Wu (Indiana University)
Proceedings Chair	George Fletcher (Washington State University)

PROGRAM COMMITTEE

Pankaj K. Agarwal (Duke University)
Marcelo Arenas (Pontifical Catholic Univ. of Chile)
Mikhail Atallah (Purdue University)
Henrik Bjorklund (Umea University)
Mikolaj Bojanczyk (Warsaw University)
Rada Chirkova (North Carolina State University)
Graham Cormode (AT&T Labs Research)
Jan Hidders (Delft Univ. of Technology)
Benny Kimelfeld (IBM Almaden Research Center)
Tova Milo (Tel Aviv University)
Jeffrey Naughton (Univ. of Wisconsin)
Lucian Popa (IBM Almaden Research Center)
Riccardo Rosati (University of Rome La Sapienza)
Dan Suciu (University of Washington)
Val Tannen (University of Pennsylvania)
Dirk Van Gucht (Indiana University)
Jan Van den Bussche (Hasselt University)
Stijn Vansummeren (Hasselt University)
Limsoon Wong (National University of Singapore)

SIGMOD Demonstration Track

The SIGMOD demonstration program has become a popular venue to demonstrate the utility of database research prototypes. Because of its continued success, the demonstration program has become highly selective and, therefore, well regarded. Demonstrations of innovative database system research are solicited, which illustrate research contributions in an interesting, intuitive, and interactive manner.

We solicit submissions that will demonstrate major, new, advances in the state-of-the-art of data management technologies. A demonstration proposal should differ from regular research papers in several important aspects. First, it should clearly describe the overall architecture of the system or technology demonstrated, without being a short research paper. Second, it should put great emphasis on the motivation, applications, and novelty of the presented system or technology. Third and importantly, it should clearly describe the demonstration scenario. In particular, it should clearly explain how the audience can interact with the demonstrated system, in order to obtain understanding of the underlying technology. For demonstrations running over the web, a back-up scenario should be described, in case of low connectivity at the demonstration venue.

IMPORTANT DATES

November 5, 2009: Manuscript submission
February 15, 2010: Notification of acceptance
March 15, 2010: Final camera-ready papers due

SUBMISSION

All aspects of the submission and notification process will be handled electronically. Detailed submission instructions will be posted on the conference website. Submissions to the demonstrations track are not subject to double blind reviewing. The author name(s) and affiliation(s) must be present in the submitted document. Any proposal violating the length, file type, or formatting requirements will be rejected without review.

PROGRAM COMMITTEE

Chair
Yufei Tao (Chinese University of Hong Kong)

Committee Members
Carl-Christian Kanne (Univ. of Mannheim)
Henrique Andrade IBM
Lisa Amini (IBM)
Michalis Petropoulos (Univ. of Buffalo)
Mohamed F. Mokbel (Univ. of Minnesota)
Shimin Chen (Intel)
Sudarshan S. Chawathe (Univ. of Maine)
Themis Palpanas (Univ. of Trento)
Vasilis Vassalos (Athens University of Economics and Business)
Vladislav Shkapenyuk (AT&T)
Wook-Shin Han (Kyungpook National University)
Xiaokui Xiao (Nanyang Technological University)

SIGMOD Industrial Track

The industrial track of SIGMOD 2010 will be the forum for high quality presentations on innovative commercial software for all facets of information technology solving significant information management problems. The topics include innovations in commercial database management and middleware systems including information integration and warehousing, cloud computing platforms and applications, benchmarking, as well as significant applications leveraging database and information retrieval technology. Submissions that do not relate to commercial software or industrial-strength software intended for wide use are discouraged. Acceptance criteria will be innovativeness of software and the potential of impact.

We invite proposals for individual talks for the industrial track to be submitted electronically via the industrial track submission website. A talk proposal consists of a 500 word abstract. Reviews will not be provided for talk proposals. The industrial track committee will contact potential speakers upon review of proposals for further details and evaluation. Invitation for a talk may or may not result in a paper published in the conference proceedings. This decision will be at the discretion of the industrial track PC. We also invite proposals for entire sessions to be sent by email to the Industrial Program Chair. Such proposals should be about a coherent theme of relevance to the data management industry and identify the speakers in the session.

The deadline for the submissions is November 5, 2009. Submissions should be uploaded through the submission site.

PROGRAM COMMITTEE

Chair
Berthold Reinwald (IBM Research)

Committee Members
Ben Reed (Yahoo! Research)
Chet Murthy (IBM Research)
Dave Campbell (Microsoft)
James Hamilton (Amazon)
Jussi Myllymaki (Google)
Wen-Syan Li (SAP Research China)

SIGMOD Panel Proposals

We solicit proposals for panels at the 2010 SIGMOD Conference in Indianapolis, Indiana. Panel proposals are expected to address new, exciting, and controversial issues. The proposed panel should be provocative, informative, and entertaining.

Panel proposals must include:

1. Description of the panel topic (no more than one page)
2. Name, affiliation, brief bio, and contact information for the proposed panel chair
3. Names, affiliations, and brief bios for up to four panelists in addition to the panel chair. The proposed panelists must have made a commitment to participate.

A mix of industry and academic panel members is encouraged.

Please submit proposals by E-mail in PDF format to Alon Halevy, halevy [at] google.com, by 5PM PST on December 3, 2009.

SIGMOD Tutorial Proposals

We solicit proposals for tutorials for presentation at the 2010 SIGMOD conference. Proposals must provide an in-depth survey of the chosen topic with the option of describing a particular piece of work in detail. A meaningful summary of open issues in the topic would be a plus.

Proposals must be no more than five pages, using an 11 pt or larger font for the body of the text of the proposal, and must include enough details to provide a sense of both the scope of material to be covered and the depth to which it will be covered. Proposals should also indicate the tutorial length (typically 1.5 or 3 hours; if the tutorial can be either length, please be sure to identify which material is included for each length). Proposals should also identify any other venues in which all or part of the tutorial has been or will be presented, and explain how the current proposal differs from those other editions of the tutorial. Tutorial proposals must clearly identify the intended audience and any prerequisite knowledge for attendees. Proposals should include a brief (no more than 3 sentences) professional biography.

Please submit proposals by E-mail (in PDF format) to Bernhard Mitschang (Bernhard.Mitschang [at] ipvs.uni-stuttgart.de) by 5PM PST on December 3rd, 2009.

Guest editors:

Sal Stolfo, Columbia University (sal@cs.columbia.edu)
Gene Tsudik, UC Irvine (gts@ics.uci.edu)

Please email the guest editors with a brief description of the article you plan to submit by 15 Oct. 2009.

Final submissions due 15 Nov. 2009

Privacy-Preserving Sharing of Sensitive Information (PPSSI) is motivated by the increasing need for organizations or people who don't fully trust each other to share sensitive information.

Many types of organizations must often collect, analyze, and disseminate data rapidly and accurately without exposing sensitive information to wrong or untrusted parties. For example, census-takers collect private data with the understanding that it won't be released in a form traceable to the individual who provided it. Companies might be willing to divulge sensitive financial data to organizations that release only aggregate data for an industry sector. A hospital might share patient information with a state health agency but only to allow the latter to determine the number (and not the identities) of uninsured patients. While statistical methods for protecting data have been in use for decades, they're not foolproof and they generally involve a trusted third party to produce privacy-preserving statistical digests.

More recently, techniques employing secure multi-party function evaluation, encrypted keywords, and private information retrieval have been studied and, in a few cases, deployed, However there are no practical tools and technologies to guarantee data privacy, especially, whenever organizations have certain common goals and require exchanges of data. To this end, the objective of PPSSI technology is to enable multiple entities to cooperate and share information without exposing more than what is necessary to complete a common task.

Potential submission topics include (but are not limited to) the following:

IEEE
SECURITY&PRIVACY

Call for Papers:

Privacy-Preserving Sharing of Sensitive Information

For submission information and author guidelines, please visit www.computer.org/security/author.htm

- PPSSI requirements and policy enforcement; prospective policies governing PPSSI, including formal models and policy languages as well as trust models.
- Data "cleaning" and obfuscation techniques.
- Cryptographic protocols; innovative constructs, their performance and implementation issues, for example, private information retrieval, searching over encrypted data and private set operations.
- Data management; storage and data management issues arising in PPSSI settings.
- Secure hardware; architectures and technologies in support of PPSSI

In general, we welcome articles that address innovative conceptual, implementation, and experimental results relevant to PPSSI. Articles addressing requirements, especially from potential users of PPSSI technology, are encouraged as well. Articles describing products or narrow/specific applications as well as largely theoretical articles are discouraged.

www.computer.org/security/cfp.htm

Submissions will be subject to the peer-review methodology for refereed papers. Articles should be 6,000 words, maximum, with a maximum of 15 references. Articles should be understandable to a broad audience of people interested in security and privacy. The writing should be down to earth, practical, and original. Authors should not assume that the audience will have specialized experience in a particular subfield. All accepted articles will be edited according to the IEEE Computer Society style guide.

To submit a manuscript, please log on to Manuscript Central (https://mc.manuscriptcentral.com/cs-ieee) to create or access an account, which you can use to log on to *S&P*'s Author Center and upload your submission.