SIGMOD Officers, Committees, and Awardees

Chair	Vice-Chair	Secretary/Treasurer
Donald Kossmann Systems Group ETH Zürich Cab F 73 8092 Zuerich SWITZERLAND +41 44 632 29 40 <donaldk AT inf.ethz.ch>	Anastasia Ailamaki School of Computer and Communication Sciences, EPFL EPFL/IC/IIF/DIAS Station 14, CH-1015 Lausanne SWITZERLAND +41 21 693 75 64 <natassa AT epfl.ch>	Magdalena Balazinska Computer Science & Engineering University of Washington Box 352350 Seattle, WA USA +1 206-616-1069 <magda AT cs.washington.edu>

SIGMOD Executive Committee:
Anastasia Ailamaki, Magdalena Balazinska, K. Selçuk Candan, Curtis Dyreson, Donald Kossmann, Yannis Ioannidis, Richard Hull, and Ioana Manolescu.

Advisory Board:
Raghu Ramakrishnan (Chair), Yahoo! Research, <First8CharsOfLastName AT yahoo-inc.com>, Amr El Abbadi, Serge Abiteboul, Ricardo Baeza-Yates, Phil Bernstein, Elisa Bertino, Mike Carey, Surajit Chaudhuri, Christos Faloutsos, Alon Halevy, Joe Hellerstein, Renée Miller, C. Mohan, Beng-Chin Ooi, Z. Meral Ozsoyoglu, Sunita Sarawagi, Min Wang, and Gerhard Weikum.

SIGMOD Information Director:
Curtis Dyreson, Utah State University, <curtis.dyreson AT usu.edu>

Associate Information Directors:
Manfred Jeusfeld, Georgia Koutrika, Michael Ley, Wim Martens, Mirella Moro, Rachel Pottinger, Altigran Soares da Silva, and Jun Yang.

SIGMOD Record Editor-in-Chief:
Ioana Manolescu, Inria Saclay—Île-de-France, <ioana.manolescu AT inria.fr>

SIGMOD Record Associate Editors:
Denilson Barbosa, Pablo Barceló, Vanessa Braganholo, Marco Brambilla, Chee Yong Chan, Rada Chirkova, Anish Das Sarma, Glenn Paulley, Alkis Simitsis, Nesime Tatbul, and Marianne Winslett.

SIGMOD Conference Coordinator:
K. Selçuk Candan, Arizona State University <candan AT asu.edu>

PODS Executive Committee: Rick Hull (chair), <hull AT research.ibm.com>, Michael Benedikt, Wenfei Fan, Maurizio Lenzerini, Jan Paradaens, and Thomas Schwentick.

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

Awards Committee:
Masaru Kitsuregawa (chair, University of Tokyo, <kitsure AT tk1.iis.u-tokyo.ac.jp>), Rakesh Agrawal, Elisa Bertino, Umesh Dayal, and Maurizio Lenzerini.

Jim Gray Doctoral Dissertation Award Committee:
Johannes Gehrke (Co-chair), Cornell Univ.; Beng Chin Ooi (Co-chair), National Univ. of Singapore, Alfons Kemper, Hank Korth, Alberto Laender, Boon Thau Loo, Timos Sellis, and Kyu-Young Whang.

[Last updated : September 24th, 2013]

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. Formerly known as the "SIGMOD Innovations Award", it now honors 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)
Umeshwar Dayal (2010)	Surajit Chaudhuri (2011)	Bruce Lindsay (2012)
Stefano Ceri (2013)		

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)
David Lomet (2010)	Gerhard Weikum (2011)	Marianne Winslett (2012)
H.V. Jagadish (2013)		

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.* Recipients of the award are the following:

- **2006** *Winner*: Gerome Miklau, University of Washington. *Runners-up*: Marcelo Arenas and Yanlei Diao.
- **2007** *Winner*: Boon Thau Loo, University of California at Berkeley. *Honorable Mentions*: Xifeng Yan and Martin Theobald.
- **2008** *Winner*: Ariel Fuxman, University of Toronto. *Honorable Mentions*: Cong Yu and Nilesh Dalvi.
- **2009** *Winner*: Daniel Abadi, MIT. *Honorable Mentions*: Bee-Chung Chen and Ashwin Machanavajjhala.
- **2010** *Winner:* Christopher Ré, University of Washington. *Honorable Mentions*: Soumyadeb Mitra and Fabian Suchanek.
- **2011** *Winner*: Stratos Idreos, Centrum Wiskunde & Informatica. *Honorable Mentions*: Todd Green and Karl Schnaitterz.
- **2012** *Winner*: Ryan Johnson, Carnegie Mellon University. *Honorable Mention*: Bogdan Alexe.
- **2013** *Winner*: Sudipto Das, University of California – Santa Barbara. *Honorable Mention*: Herodotos Herodotou and Wenchao Zhou.

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

[Last updated : September 24th, 2013]

Editor's Notes

Welcome to the September 2013 issue of the ACM SIGMOD Record! After the highly successful SIGMOD and PODS 2013 conferences in New York, our community leadership is renewed through the election of the new SIGMOD officers: Donald Kossmann (chair), Anastasia Ailamaki (vice-chair) and Magdalena Balazinska (secretary and treasurer). Congratulations to the newly elected well-known members of our community, in particular to the former Record editor Magda! A word from the chairs follows these notes.

The issue opens with a Database Principles article by Chomicki, Ciaccia, and Meneghetti on skyline queries. Since their original inception in 2001, skyline queries (and algorithms) have enjoyed an enormous popularity, and numerous applications from Web commerce to recommendation (possibly in a social context) and more generally to decision making. This sharp and clear article recalls the basic formal notions behind skyline queries, in particular the notion of Pareto dominance, reviews the main classes of queries and algorithms, discuss skyline query result size, and point to areas of further research, including handling uncertainty and integrating some form of users' preferences to inform the skyline choice.

The paper by Ceri, Della Valle, Freytag, Palpanas, Pedreschi, and Trasarti illustrates how the mega-modeling approach previously proposed by the authors can be used to support a set of apparently very different data analysis scenarios. Mega-modeling is introduced as a theory and technology of model construction, model search, model composition, reuse and evolution; a design based on mega-modeling consists of an assembly of mega-modules, each of which consists of the successive stages of data preparation, analysis, and evaluation (preparation of the output). The target applications are inspired from the mobile and stream processing contexts; they include individual and collective profiling (through mobility trajectories), outlier detection, sentiment shifting (in micro-post streams) and more.

High-value Web data tables are the focus of the work by Lautert, Scheidt, and Dorneles. The (Web) data management community has taken recent notice of the wealth of well-structured data to be found in table format on the Web, and the authors propose a taxonomy of such tables according to how information is organized within them, including all the "hard-to-handle" aspects such as merged cells etc. The purpose of the taxonomy is to improve the effectiveness of Web table data processing, since each class of table may better handled by specific algorithms. Finally, the authors present a neural network-based Web table classifier of Web tables into these categories.

Andrew Eisenberg reports to us on the new XQuery 3.0 candidate recommendation published by the W3C in January 2013. If you think you missed 2.0, do not worry! 3.0 is the first update after the 2007 1.0 version. Among the novel features, XQuery 3.0 provides support for grouping, a window operator, more general FLWOR expressions, try-catch and higher-order functions. General ("relaxed") FLWOR expressions are interesting in their resemblance to the "sequential, all in a script" data processing framework of (say) MapReduce or PigLatin. Experience will tell how widely used and adopted will these languages be in the short and medium run, but XQuery (and this has been the case since its early days) certainly does not lack expressive power!

The Systems and Prototype columns features two system descriptions. PAIRSE, by Benslimane, Barhamgi, Cuppens, Morvan, Defude, and Nageba (and many additional authors) is concerned with privacy-preserving composition of services and algorithms in a peer-to-peer context. SYSEO, by Chabane, d'Orazio, Gruenwald, Mohamad and Rey details two components of a software built for endoscopic image processing by physicians. These papers share the characteristic of being about the results of three-years French R&D programs funded by the French national research agency (ANR).

The Profiles columns are basically all remarkable, as they provide the rare opportunity for respected colleagues to open up on their life, career, expectations, hopes and previously (in)famous quotes. Andreas Reuter's interview featured in this issue still stands out as one of the most lively and informative. The discussion of the weather and its impact on the writing of the classical Transaction Processing book together with Jim Gray is something I feel compelled to give to all my Greek students and post-docs to read, every time they complain about Parisian weather[1]! Andreas tells us that yes, there is a life outside transactions (something that the abovementioned students and post-docs typically do not need explaining). Further, he shares thoughts about his experience of starting the first full English-language, private university in Germany, and also on how he had to withdraw from managing it due to the complexity of handling sponsors, PR, political debates around the project and much more.

Zhou and Sadiq present the data centric research at the University of Queensland, spanning both the the timeline of the department's history and the breadth of today's research topics, including trajectory computing, multimedia search, data quality, IT governance, and numerous applications to science and humanities.
Last but not least, two workshop reports close the issue: BIRTE 2012 (Business Intelligence for the Real Time Enterprise) by Castellanos, Dayal and Rundensteiner, and the PhD workshop next to the CIKM conference 2012, by Kacimi, Suchanek and Varde.

As you may have noticed, the RECESS submission Web site has moved from the University of Pittsburgh where it had been installed and maintained during the tenure of the previous Editor-in-Chief, Alex Labrinidis, to an ACM-hosted Web site. I take the opportunity to thank Alex, the ACM IT support staff, Curtis Dyreson and especially my former PhD student Asterios Katsifodimos for ensuring a smooth transition! Your submissions to the Record are now welcome via the submission site:

http://sigmod.hosting.acm.org/record

Prior to submitting, be sure to peruse the Editorial Policy on the SIGMOD Record's Web site (http://www.sigmod.org/publications/sigmod-record/sigmod-record-editorial-policy).

Ioana Manolescu

September 2013

Past SIGMOD Record Editors:

Harrison R. Morse (1969)
Daniel O'Connell (1971 – 1973)
Randall Rustin (1974-1975)
Douglas S. Kerr (1976-1978)
Thomas J. Cook (1981 – 1983)
Jon D. Clark (1984 – 1985)
Margaret H. Dunham (1986 – 1988)
Arie Segev (1989 – 1995)
Jennifer Widom (1995 – 1996)
Michael Franklin (1996 – 2000)
Ling Liu (2000 – 2004)
Mario Nascimento (2005 – 2007)
Alexandros Labrinidis (2007 – 2009)

[1] Which is to say, basically every day, except for August holidays, when they're back home…

Letter from the new SIGMOD Officers

Dear SIGMOD members:

Earlier this year, you elected new SIGMOD officers. We are humbled that you chose us and are looking forward to working with you.

Firstly, **we would like to thank our predecessors**, Yannis Ioannidis, Christian Jensen, and Alex Labrinidis who have been running SIGMOD so smoothly in the last four years. They have done an amazing job, left a clean field behind, and we will try hard to follow their path and fit into their shoes.

Secondly, **we would like to ask you for your input**, help, and continuous support. SIGMOD is run primarily by volunteers; some in formal roles, such as the executive committee and the organizers of the various conferences which are sponsored by SIGMOD. Ultimately, however, all your actions every day make a difference and make this community such a great place to be in.

One area your immediate action is needed is in **bridging with other disciplines**. We live in exciting times and we believe that these times require particular efforts. Given the impressive growth and success of our community in the industrial and in the academic sectors, one may argue that we could lean back and relax because we have it all figured out. Our community has evangalized declarative programming, automatic optimization and parallelization, we have formalized concurrency and consistency, and we always strive to develop systems that scale with the data, the workload, and the number of machines. All of these innovations, as well as our general philosophy, become increasingly important in the Big Data era. Unfortunately, however, it is not sufficient to point to our old papers and trust that people will read them and implement our ideas. We need to become involved, adapt our ideas to the emerging aplications, and collaborate with other scientific communities and disciplines (within and outside of computer science) in order to have real impact. If we do not communicate, others will start to reinvent the wheel, making it increasingly difficult for us to even explain, let alone apply, our ideas.

Another area in which we need your help is in **improving the way our research community works**. Ultimately, it must be fun to be a member of our community. Fortunately, there is a great deal of fun already, but there is also a great deal of frustration. There seems to be a general perception in almost all scientific disciplines (not only computer science or databases) that the reviewing and publishing system is broken. Globalization and the Internet have made it easier to do science and have accelerated innovation, but they have also put new challenges on all scientific communities. Our community can play a decisive role and become a model for other communities. We have strong conferences and great demographics with many young people with fresh ideas joining our community every year. Furthermore, we have an open-minded culture of experimenting with new ideas such as the reviewing and publishing scheme pioneered by PVLDB or the „repeatability of experiments" which has started at SIGMOD 2008.

To meet the challenges of the new data-driven era and continue to evolve (or maybe even revolutionize) our system, it is critical that we hear your ideas and concerns. So, please, send us e-mail whenever you have a thought or call us on the phone or talk to us at a conference in person or interact with us in whatever way you like and whenever you like. We will collect all ideas and try to moderate the discussion within the community. As a freshly elected committee, we believe that we can improve things so that our ideas have even bigger impact than they have now and have had in the past.

Donald Kossmann	Anastasia Ailamaki	Magda Balazinska
donaldk@ethz.ch	natassa@epfl.ch	magda@cs.washington.edu

Skyline Queries, Front and Back*

Jan Chomicki
University at Buffalo
Buffalo, NY 14260
chomicki@buffalo.edu

Paolo Ciaccia
University of Bologna
40136 Bologna, Italy
paolo.ciaccia@unibo.it

Niccolo' Meneghetti
University at Buffalo
Buffalo, NY 14260
niccolom@buffalo.edu

ABSTRACT

Skyline queries are a popular way to obtain preferred answers from the database by providing only the orderings of attribute values. The result of a skyline query consists of those input tuples for which there is no input tuple having better or equal values in all the attributes and a better value in at least one attribute. In this article, we summarize the basic notions and properties of skyline queries, and discuss their extensions and generalizations. In particular, we consider skyline algorithms and skyline cardinality issues.

1. INTRODUCTION

Multi-criteria analysis [16] is a common approach to address the needs of decision-making applications. In some such settings a set of dimensions and a set of alternatives, both finite, are given. For example, a car shopper considers the price, make, model and mileage of the car, as well as the vehicles currently in stock. The analysis identifies the best, *most preferred* alternatives, which are obtained by eliminating those that are *dominated* by other alternatives. Under *Pareto* efficiency, the dominance relationship has a particularly simple structure: an alternative o_1 dominates an alternative o_2 if o_1 is better than or equal to o_2 in all dimensions and better than o_2 in at least one dimension. (An equivalent formulation requires that o_1 and o_2 be different, and o_1 be better than or equal to o_2 in all dimensions.)

In the database context, the concepts of multi-criteria decision analysis have a very natural interpretation. The dimensions correspond to attributes, and the alternatives, to the objects present in the database. Pareto dominance leads to *skyline* queries.

Example 1.1. *A prospective student, choosing a school, takes into account the financial support promised by the school, as well as its percentile rank among all schools. We represent the relevant data using a relation* `School(Id,Rank,Support)`*, which is depicted in Table 1 and plotted in Figure 1.*

id	rank	support	id	rank	support
t_1	96	5,000	t_8	96	3,000
t_2	95	6,000	t_9	93	3,500
t_3	89	8,000	t_{10}	92	2,500
t_4	87	9,000	t_{11}	88	4,500
t_5	86	10,000	t_{12}	85	7,000
t_6	84	14,000	t_{13}	83	6,500
t_7	81	14,500	t_{14}	80	11,000

Table 1: Student database

Suppose the student does not have a scoring function that assigns a numeric score to each school: such functions are difficult to construct. However, the student can still determine whether a school is dominated by some other school providing better support and having the same or higher rank, or having a higher rank and providing the same or better support. The dominated schools represent inferior choices and can be eliminated. The remaining schools are nondominated and form the skyline *of the input. The skyline contains all the best choices, in a precise sense. In this case, the skyline consists of the tuples $t_1, t_2, t_3, t_4, t_5, t_6$, and t_7 (the black dots).*

The notion of skyline queries was pioneered in [8]. Subsequently, the interest in this area has exploded: [8] has garnered over 1200 citations (Google Scholar, May 2013). The research has uncovered interesting properties of skylines, designed efficient algorithms for computing skyline queries, and produced numerous generalizations and extensions of the basic framework. Unfortunately, the sheer volume of publications on skylines and related topics prevents us from covering here all interesting issues

*__Database Principles Column__. Column editor: Pablo Barceló, Department of Computer Science, Universidad de Chile, Santiago, Chile. E-mail: pbarcelo@dcc.uchile.cl.

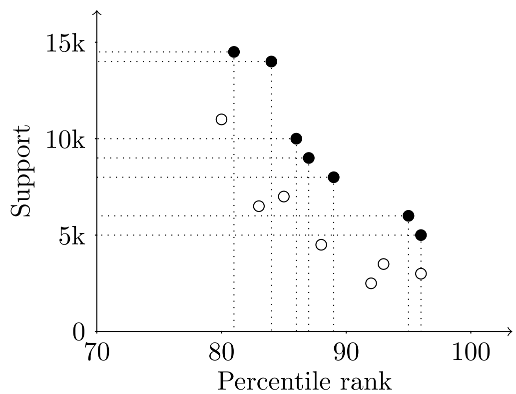


Figure 1: Student database (plotted)

studied in this area.

The needs of decision-making applications have been calling for designing new classes of queries that go beyond skyline queries. Richer classes of preferences and forms of query results have been studied. The research is motivated not only by semantic concerns but also by the observation, confirmed both analytically and experimentally, that skyline sizes may get impractically large.

It is important to keep in mind the distinctions between (1) Pareto dominance, (2) skyline queries that return all the non-Pareto-dominated objects, (3) skylines: the results of skyline queries, and (4) skyline algorithms that compute skylines.

The plan of the paper is as follows. In Section 2, we introduce the basic notions of Pareto dominance and study its basic properties. In Section 3, we consider skyline queries and their variants. Section 4 contains a survey of skyline algorithms. In Section 5, we address the issue of skyline cardinality. Section 6 shows generalizations and adaptations of Pareto dominance to various contexts. In Section 7, we consider some issues that we believe will be of importance in future research on skylines: uncertainty and elicitation of dominance relations. We conclude in Section 8.

2. DOMINANCE RELATIONS

2.1 Pareto dominance

We present here first a basic formal model of *Pareto dominance* (also called *Pareto preference*). Later, we will show how Pareto dominance can be generalized and extended.

Let $\mathcal{A} = \{A_1, ..., A_d\}$ be a finite set of attributes (a relation schema). The number d is the *dimension* of Pareto dominance. Every attribute $A_i \in \mathcal{A}$ is associated with an *infinite domain* $\mathcal{D}_{A_i}$ (here: real numbers). We work with the *universe of tuples* $\mathcal{U} = \prod_{A_i \in \mathcal{A}} \mathcal{D}_{A_i}$. A *dominance* (preference) relation is a subset of $\mathcal{U} \times \mathcal{U}$. Given a tuple $t \in \mathcal{U}$, we denote the value of its attribute A_i by $t[A_i]$. This notation can be easily generalized to sets of attributes.

Every attribute $A_i \in \mathcal{A}$ is associated with one of the standard orderings of the reals, $>$ or $<$, denoted by $>_{A_i}$. For simplicity, we assume in this paper that $>_{A_i}$ is $>$ (*larger is better*). The Pareto dominance relation $\succ^{pto}$ is defined as

$$t \succ^{pto} s \equiv t \neq s \wedge \bigwedge_{A_i \in \mathcal{A}} t[A_i] \geq_{A_i} s[A_i],$$

or, equivalently, as

$$t \succ^{pto} s \equiv \bigvee_{A_i \in \mathcal{A}} t[A_i] >_{A_i} s[A_i] \wedge \bigwedge_{A_i \in \mathcal{A}} t[A_i] \geq_{A_i} s[A_i].$$

These are *strict* versions; there is also a *non-strict* version obtained in the standard way:

$$t \succeq^{pto} s \equiv \bigwedge_{A_i \in \mathcal{A}} t[A_i] \geq_{A_i} s[A_i].$$

Strictly speaking, $\succ^{pto}$ denotes a different relation for different schemas $\mathcal{A}$. In the cases where it is necessary to disambiguate the set of attributes over which dominance is defined, we will add an appropriate subscript. In practical applications, some relation attributes may be irrelevant for Pareto dominance: see the attribute *id* in Example 1.1. For simplicity, we do not consider such attributes.

From an algebraic perspective, Pareto dominance can be defined using a binary accumulation operator $\otimes$ (also called Pareto) combining the attribute orders $>_{A_i}, i = 1, \ldots, d$:

$$\succ^{pto}_{\mathcal{A}} = >_{A_1} \otimes >_{A_2} \otimes \cdots \otimes >_{A_d}$$

where $\succ^{pto}_{A_i} = >_{A_i}$ and $\succ^{pto}_{XY} = \succ^{pto}_{X} \otimes \succ^{pto}_{Y}$ is defined as

$$\begin{aligned} t[XY] \succ^{pto}_{XY} s[XY] \ \equiv & \\ & t[X] \succ^{pto}_{X} s[X] \wedge t[Y] \succeq^{pto}_{Y} s[Y] \\ \vee \ & t[X] \succeq^{pto}_{X} s[X] \wedge t[Y] \succ^{pto}_{Y} s[Y] \end{aligned}$$

for $XY \subseteq \mathcal{A}$ and $X \cap Y = \emptyset$.

Pareto accumulation is associative and commutative.

2.2 Properties

Representation. Clearly, $\succ^{pto}$ is irreflexive and transitive. However, it is not negatively transitive (its complement does not have to be transitive), and thus $\succ^{pto}$ fails to be a weak order, as in the example below:

$$(2,0) \not\succ^{pto} (0,2), (0,2) \not\succ^{pto} (1,0), (2,0) \succ^{pto} (1,0).$$

The lack of the weak order property implies that $\succ^{pto}$ *cannot be represented using any scoring function.* Formally, a scoring function f represents the dominance relation $\succ_f$ such that

$$t \succ_f s \equiv f(t) > f(s).$$

To show that $\succ_f$ is a weak order, suppose $t \succ_f s$. Then $f(t) > f(s)$. So for every w, $f(t) > f(w)$ or $f(w) > f(s)$, and thus $t \succ_f w$ or $w \succ_f s$. Therefore, $t \not\succ_f w$ and $w \not\succ_f s$ imply $t \not\succ_f s$.

Robustness. One way of characterizing the robustness of $\succ^{pto}$ is through *invariance* with respect to the transformations of the underlying space. A function g is *Pareto-invariant* if for all tuples t and s, $t \succ^{pto} s$ implies $g(t) \succ^{pto} g(s)$.

A useful related notion is that of *monotonicity.* A function f mapping $\mathcal{U}$ into some Cartesian product of domains $\mathcal{V}$ is *monotone* if for all tuples t and s, $t \succeq^{pto} s$ implies $f(t) \succeq^{pto} f(s)$.

One can define special classes of monotone functions, for example *shifting* (addition of a vector of constants) and *scaling* (multiplication by a vector of constants). Shifting is Pareto-invariant. Scaling by an all-positive vector is also Pareto-invariant. However, scaling by a vector that has a zero component is not necessarily Pareto-invariant, as Example 2.1 shows. So monotonicity does not imply Pareto-invariance.

Example 2.1. *Consider the tuples $t = (3,1)$ and $s = (2,1)$, and the scaling vector $(c_1, c_2) = (0,1)$. Clearly, $t \succ^{pto} s$ but*

$$(c_1 \cdot t[A_1], c_2 \cdot t[A_2]) \not\succ^{pto} (c_1 \cdot s[A_1], c_2 \cdot s[A_2]).$$

3. SKYLINE QUERIES

3.1 Skyline using winnow

There is more than one way to incorporate the notion of Pareto dominance into a relational query language. We call all such queries *Pareto queries.* Arguably, the simplest and the most widely used kind of a Pareto query is the relational operator with the meaning *"Retrieve all the nondominated tuples in the input relation."* We use the term *winnow* for this operator, and the notation $\omega_\succ(R)$ where $\succ$ is a dominance relation and R a database relation schema [10]. Typically $\succ$ is defined by a logic formula. The semantics of winnow is as follows:

$$\omega_\succ(r) = \{t \in r \mid \neg\exists t' \in r.\ t' \succ t\}$$

where r is an instance of R. A *skyline query* is a special instance of winnow in which $\succ$ is $\succ^{pto}$, and is written as $sky_{\mathcal{A}}(R)$ (where $\mathcal{A}$ is the set of all attributes of R). The result $sky_{\mathcal{A}}(r)$ of a skyline query for an instance r of R is called a *skyline*: $sky_{\mathcal{A}}(r) = \omega_{\succ^{pto}}(r)$. The size of the skyline is denoted by $\ell = |sky_{\mathcal{A}}(r)|$. Intuitively, the Pareto-dominated tuples are inferior and should not be of interest to the user.

3.2 Maxima of scoring functions

As Theorem 3.1 below shows, a skyline consists of the maxima of monotone scoring functions. Thus, even if the scoring function is not known, the highest-scoring tuple is guaranteed to be in the skyline. This result was already mentioned in the original skyline paper [8]. A proof appeared in the full version of [13]. We present a different proof here.

Theorem 3.1.

$$\forall r.\ \forall t \in r.\ t \in \text{sky}_A(r) \text{ iff } \exists f \in \mathcal{M}.\quad \forall t' \in r.\ t' \neq t \Rightarrow f(t) > f(t'),$$

where $\mathcal{M}$ is the set of all monotone scalar-valued functions over $\mathcal{U}$.

Proof. $\Leftarrow$ Assume RHS holds and $t \notin sky_A(r)$. Then there is $t' \neq t$ such that $t' \succ^{pto} t$. By the monotonicity of f, $f(t') \geq f(t)$. A contradiction.

$\Rightarrow$ Assume $t \in sky_A(r)$. Without lack of generality, we can assume all attribute values are strictly positive: a simple shifting operation can make all the tuples strictly positive; the composition of a translation with a monotone function is again a monotone function. It is easy to see that for every $t' \neq t$, there is an attribute A_i such that $t[A_i] > t'[A_i]$ (otherwise t would not be in the skyline). The function f is defined for every t' as

$$f(t') = \min_i \{\frac{t'[A_i]}{t[A_i]} \mid i = 1, \ldots, d\}.$$

Thus $f(t) = 1$ and $f(t') < 1$ for $t' \neq t$. Also, f is monotone. $\square$

3.3 Algebraic laws

Algebraic laws relating query operators serve as a foundation of query optimization. Such laws can be used as rewrite rules to produce different, presumably more efficient formulations of queries. One of the basic insights of this field is that it is useful to push low-cost, size-reducing operators like relational algebra selection below high-cost, size-increasing operators like join. Such a transformation usually reduces the size of intermediate query results and the overall cost of evaluating the query.

The above intuition applies to skyline queries as well. A skyline query (winnow) is a high-cost operation, so it pays to push a selection below it.

THEOREM 3.2. *[10] If for every t and s such that $t \succ^{pto} s$ $\alpha(s)$ implies $\alpha(t)$, then for every relation instance r*

$$\sigma_\alpha(\text{sky}_\mathcal{A}(r)) = \text{sky}_\mathcal{A}(\sigma_\alpha(r)).$$

In Example 1.1, the selection $\sigma_{Rank>80}$ commutes with the skyline query but the queries $\sigma_{Rank<80}$ and $\sigma_{Rank=80}$ do not.

Other algebraic laws involving skyline and more general queries are studied in [10, 11, 26, 31].

3.4 Subspace skylines

In addition to the full-space skyline $sky_A(R)$, subspace skylines $sky_\mathcal{B}(R)$ for $\mathcal{B} \subset \mathcal{A}$ [17] are also considered in order to extract more information from the input dataset. For example, subspace skylines show which points excel in specific dimensions and which have all-around strength.

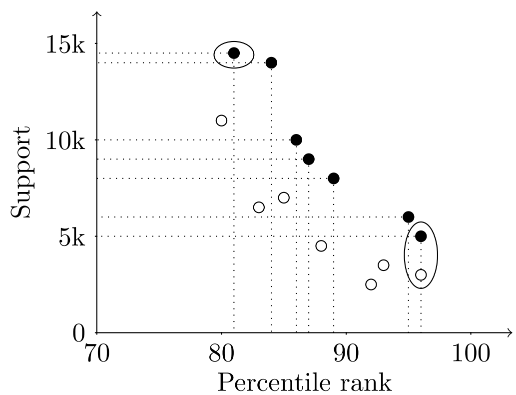


Figure 2: Subspace skylines

EXAMPLE 3.1. *We use the setting of Example 1.1. In Figure 2, the two one-dimensional subspace skylines are circled with ovals, and the two-dimensional skyline elements are black. Interestingly, a subspace skyline does not have to be contained in higher-dimensional skylines, unless no attribute values can be shared by different objects [17].*

Subspace skylines lead to new, interesting classes of queries. The *skyline membership query* determines the subspaces where a given object is in the subspace skyline. The *skycube query* returns the skylines in all the nonempty subspaces.

3.5 Other query classes

It is remarkable that Pareto dominance, a single kind of dominance relation, has been used to define many new classes of queries [41]. We briefly survey some of those classes here. *Constrained skyline queries* are skyline queries applied to the result of a selection. *Ranked skyline queries* are skyline queries whose output is ranked using a scoring function. When the scoring function returns the number of objects dominated by the given object, we have *enumerating queries*; when only top-k objects are returned, the queries are k-*dominating*. Finally, k-*skyband queries* return the objects dominated by at most k other objects. Clearly, skyline queries are 0-skyband.

4. SKYLINE ALGORITHMS

The skyline of a relation instance r with n tuples and attributes $\mathcal{A} = \{A_1, ..., A_d\}$ can be naïvely computed in $\mathcal{O}(n^2 d)$ time, which is $\mathcal{O}(n^2)$ if the relation schema is fixed. The quadratic bound holds since a standard nested-loops (NL) join algorithm, which compares a tuple t to each other tuple s, clearly suffices to determine if $t \in sky_\mathcal{A}(r)$.

The NL algorithm always requires $\Theta(n^2)$ time. This is also the worst-case performance of any algorithm that is applicable to arbitrary dominance relations $\succ$ and oblivious of the d-dimensional Euclidean space in which the tuples lie (as NL indeed is): If $\forall t, s \in r$ it is the case that $t \not\succ s$ and $s \not\succ t$, thus $\omega_\succ(r) = r$, then all pairs of tuples will be compared. Based on this observation, it can be argued that there are (at least) two features that can be used to escape the quadratic lower bound (or at least to reduce costs on the average) and that NL ignores: transitivity of the dominance relation and the attribute orders $>_{A_i}, i = 1, \ldots, d$. Further, it is advisable to look at *output-sensitive* algorithms, for which the performance depends on the skyline size, since the case of small-to-medium skylines is the most interesting one from a practical viewpoint.

In the following we provide a concise picture of the major approaches adopting such ideas and others as well. It is important to bear in mind that the problem of computing the skyline is equivalent to the *maximal vectors* problem studied in computational geometry, for which several algorithms are known. Here we do not cover them in detail, both for space reasons and for the absence of experimental validation on large databases.

4.1 Using transitivity

BNL. The BNL (block nested-loops) algorithm [8] is a practical way of implementing the NL approach in a database system. BNL allocates a buffer (window) W in main memory, and sequentially scans r. For each newly read tuple t, it compares t to the tuples in W. If $\exists s \in W. s \succ^{pto} t$, then t is discarded, otherwise t is inserted into W and the tuples in W dominated by t are removed. In case W saturates, a temporary file F is used to store those tuples that overflow. At the end of this process, all tuples that were inserted in W when F was still empty can be output. Another pass then starts in which F is used as the input, and the process repeats until no overflow occurs.

Empirical evaluation of BNL shows its high dependency on how the tuples are distributed, the worst case being when they are negatively correlated. BNL effectiveness also largely depends on the order in which the tuples are processed and on the window size. Indeed, for an unlimited window size, $|W| \geq n$, BNL might still require $\Theta(n^2)$ time *regardless of the skyline size*. This can be seen by considering the case in which tuples are ordered as $\langle t_1, t_2, \ldots, t_n \rangle$, $t_n \succ^{pto} t_i, i = 1, \ldots, n-1$, and no other dominance relationship exists (thus $sky_{\mathcal{A}}(r) = \{t_n\}$). At the other extreme, $|W| = 1$ guarantees that at most $\mathcal{O}(\ell n)$ comparisons will be performed, where $\ell = |sky_{\mathcal{A}}(r)|$. However the actual running time might be negatively influenced by having a small window, since I/O costs will increase. Notice that, although BNL has been empirically analyzed and its performance contrasted to that of several other algorithms, a complete understanding of this algorithm's behavior is still lacking. For instance, the scenario in which a large amount of space in main memory is available *and* a small window is used, with the rest of available memory used to cache F, has not been considered yet.

SFS. The SFS (Sort-Filter-Skyline) algorithm [12, 13] is similar to BNL, but it first topologically sorts r using a monotone function f. In the resulting order $\langle t_1, t_2, \ldots, t_n \rangle$ it is therefore guaranteed that $i < j \Rightarrow t_j \not\succ^{pto} t_i$. This property leads to three major improvements with respect to BNL: 1) SFS is *progressive*, since $t \in W$ implies $t \in sky_{\mathcal{A}}(r)$, and therefore t can be immediately output; 2) the number of passes is optimal, $\lceil |\ell|/|W| \rceil$; and, 3) if t and s are both dominated tuples, then they will not be compared to each other. As a consequence of 3), SFS will not execute more than $\mathcal{O}(\ell n + n \log n)$ comparisons ($n \log n$ being paid for sorting r).

LESS and SaLSa. There are two orthogonal directions along which SFS can be improved, and these yield the LESS and SaLSa algorithms, respectively. The basic idea of LESS [23] is to anticipate the dominance tests in the sorting phase, so as to discard some dominated tuples earlier, and consequently to reduce the sorting costs. SaLSa [4] extends SFS by avoiding to read the whole sorted input relation. Let $\langle t_1, t_2, \ldots, t_n \rangle$ be the order in which tuples are read, with t_i ($i < n$) being the last fetched tuple. Since the function f used to sort tuples is monotone, it is $\forall j > i. f(t_i) \geq f(t_j)$, thus all unread tuples correspond to points in a bounded region BR. (This requires that the attribute domains be bounded from below, which is always true if one considers *active* domains.) Therefore, if there exists t_j ($j \leq i$) such that t_j Pareto-dominates BR, that is, $\forall s \in BR. t_j \succ^{pto} s$, then the algorithm can be halted since no more tuples will enter the skyline. Theoretical analysis reveals an interesting fact about the *limiting* capability of SaLSa.

THEOREM 4.1. *[4] Let $m \leq n$ be the number of tuples that SaLSa reads. For any data distribution, the expected value of m/n monotonically decreases with n.*

Remark: Although none of the described algorithms can avoid a quadratic cost in the worst case in which the skyline has size $\Theta(n)$, their *average-case* behavior is indeed much better, as also confirmed by experimental observations and analytical results. For instance, the analysis in [24] proves that LESS has a *linear*, $\mathcal{O}(dn)$, complexity under the assumptions of independence of attributes, uniform distribution, and low probability of duplicate attribute values. A similar result is derived for BNL with either unlimited ($|W| \geq n$) or minimal ($|W| = 1$) window size, whereas for SFS it is shown that sorting is, in terms of average performance, equivalent to reducing the skyline dimensionality by one. Unfortunately, the simplifying assumptions on which results like these are based rarely hold together in real databases.

4.2 Using attribute orders

Since attribute domains are totally ordered, it is possible to partition them. This idea is at the heart of *divide & conquer* approaches, which have been pioneered in the computational geometry field. We first detail (Algorithm 1) the basic scheme of Kung et al. [33] and then discuss the D&C algorithm by Börzsönyi et al. [8] that was developed for dealing with large instances that do not fit in main memory.

After partitioning on 2 attributes A_i and A_j, the sets $S_{H_i,H_j}, S_{H_i,L_j}, S_{L_i,H_j}$, and S_{L_i,L_j} are obtained, with sets S_{H_i,L_j} and S_{L_i,H_j} that do not need to be compared. This observation, together with a rather sophisticated merging scheme, leads to the worst-case subquadratic bound $\mathcal{O}(n \log^{d-2} n)$ ($d \geq 3$). For $d = 2, 3$ this reduces to $\mathcal{O}(n \log n)$, which meets the theoretical lower bound established in [33].

The D&C algorithm shares with the above scheme the idea of recursive partitioning, but at each step it generates an *m-way partition* (rather than a 2-way one as in [33]), where m is chosen so that each

Algorithm 1 Basic divide & conquer [33]

Input: instance r with schema $\mathcal{A} = \{A_1, \ldots, A_d\}$
Output: $sky_\mathcal{A}(r)$
1: Partition r using the median m_i of some attribute A_i. Let $S_{H_i} = \{t \in r.t_i \geq m_i\}$ and $S_{L_i} = r \setminus S_{H_i}$;
2: Compute $sky_\mathcal{A}(S_{H_i})$ and $sky_\mathcal{A}(S_{L_i})$ by recursively applying step 1;
3: Merge $sky_\mathcal{A}(S_{H_i})$ and $sky_\mathcal{A}(S_{L_i})$, i.e., determine $T_{L_i} \subseteq sky_\mathcal{A}(S_{L_i})$ s.t. $sky_\mathcal{A}(r) = sky_\mathcal{A}(S_{H_i}) \cup T_{L_i}$.

of the resulting sets can be loaded in main memory. Although this makes D&C more amenable to a database scenario, its simplified merging scheme causes the worst-case complexity to rise back to $\mathcal{O}(n^2)$.

If one considers an external memory (EM) model, in which CPU is free and the cost is measured in terms of I/O operations, the currently best result is due to Sheng and Tao [45]. By developing a smart m-way merging technique, they are able to compute the skyline with $\mathcal{O}(n/B \log_{M/B}^{d-2}(n/B))$ I/Os ($d \geq 3$), where B (M) is the number of tuples in each disk block (main memory, respectively).

Remark: Although divide & conquer algorithms typically exhibit a subquadratic worst-case complexity, this does not imply their superiority over other approaches in terms of actual running time, to which many other factors contribute, for example the hidden constant factors in $\mathcal{O}()$ notation.

4.3 Low-cardinality domains

In many situations, some (or even all) attributes of interest can only assume values from a limited set of alternatives, i.e., domains have low cardinality. For instance, ratings of movies and hotels typically are integers in a small range, say $[1, 5]$. Also, *Boolean attributes* are typically used to describe the presence/absence of interesting object features.

Without loss of generality, we assume here that each combination of skyline attribute values can occur multiple times and $\forall A_i \in \mathcal{A}.|\mathcal{D}(A_i)| = V \ll n$. The LS-B algorithm [39] first builds the complete lattice of all the V^d value combinations ordered by Pareto dominance, and marks all elements as *not present* (`np`). It then sequentially reads the input relation r, and for each tuple t it marks as *present* (`p`) the corresponding lattice element. A simple level-wise analysis of the lattice is executed to determine which are the `p`-values that are also nondominated, and that consequently are in the skyline. Finally, with a second scan of r all skyline tuples are computed. (It is needed since tuples may have other attributes besides those on which the skyline is computed.) Overall, LS runs in $\mathcal{O}(dV^d + dn)$ time, which reduces to $\mathcal{O}(dn)$ if $V = \mathcal{O}(n^{1/d})$.

LS-B can also be adapted to work when (exactly) one attribute, say A_d, has *not* a low-cardinality domain. The idea of the extended algorithm, called LS, is to store in each lattice element also the *locally optimal value* (`lov`) of A_d for that element. Since $\succ_\mathcal{A}^{pto} = \succ_B^{pto} \otimes \succ_{A_d}^{pto}$, where $B = \mathcal{A} \setminus \{A_d\}$ and $\succ_{A_d}^{pto} = >_{A_d}$, for distinct tuples t and s it is:

$$t \succ_\mathcal{A}^{pto} s \equiv t[B] \succeq_B s[B] \wedge t[A_d] \geq_{A_d} s[A_d].$$

Consequently, a tuple s can be discarded if: (1) there exists a lattice element marked `p` that dominates the element of s and whose `lov` is at least as high as $s[A_d]$; or (2) $s[A_d]$ is strictly less than the `lov` of its element (implying that $\exists t.t[B] = s[B] \wedge t[A_d] >_{A_d} s[A_d]$). It is simple to show that the complexity of LS is the same as that of LS-B.

Unfortunately, no simple extension to the general case in which a mix of low- and high-cardinality domains coexist seems to be possible. Indeed, if two attributes have high cardinality, the *local skyline* with respect to these attributes should be computed for each lattice element, a fact that might nullify the advantages of using a lattice-based approach.

4.4 Index-based approaches

As with any other query type, processing of skyline queries can be accelerated if the input data have been indexed.

The BBS (Branch-and-Bound Skyline) algorithm [40] assumes that r is indexed by an R-tree, for which index regions are minimum bounding rectangles (MBRs), and that a *target (reference) point* p is available. Without loss of generality, let p be any point such that $\forall t.p[A_i] \geq t[A_i]$, $i = 1, \ldots, d$, so that the assumption *larger is better* still holds.

BBS performs an ordered scan of the index nodes based on their L_1 distance from p, i.e., for a node N whose region is $Reg(N) = [l_1, h_1] \times \ldots \times [l_d, h_d]$, it is $L_1(Reg(N), p) = \sum_i |p[A_i] - h_i|$. Notice that for each point $t \in Reg(N)$ it is guaranteed that $L_1(Reg(N), p) \leq L_1(t, p)$. The region descriptions of the nondominated index nodes that have not been accessed yet, as well as the currently nondominated points retrieved so far, are organized together in a priority queue PQ, which is kept ordered by increasing values of L_1 distances. Notice that a point t Pareto-dominates a node N, $t \succ^{pto} N$, if $\forall i.t[A_i] \geq h_i$. (If a bag semantics is assumed, then at least one inequality needs to be strict.)

Since L_1 (as well as any other L_p norm) is a monotone function, as soon as a point t becomes the top element of PQ it is guaranteed that t belongs to the skyline. Thus, BBS is *progressive*. Monotonicity of L_1 is also the key to show that BBS is *I/O-optimal*, that is, only the index nodes for which inspection of the points they contain is necessary to ensure the correctness of the result are accessed.

An index-based solution based on the *Z-order*, which maps multidimensional points to a linear address space, is introduced in [34]. The proposed 1-dimensional index structure, called ZBtree, is based on the B^+-tree principles, in that each node region is a 1-dimensional interval, i.e., a sequence of *Z-addresses*, and intervals do not overlap. Peculiar to the ZBtree is the policy according to which points and region descriptions are packed together (in leaf and non-leaf nodes, respectively), and which aims to facilitate the pruning of some regions, thus avoiding dominance tests.

4.5 Distribution and parallelism

Vertical fragmentation. Consider a scenario in which the d skyline attributes are distributed over multiple sites, each site thus providing only a partial view of the alternatives under examination. In the following we describe the basic case in which each site stores a single skyline attribute (thus, there are exactly d sites), the extension to arbitrary vertical decompositions having been recently analyzed in [48].

The BDS algorithm [3] is based on the framework that Fagin pioneered for the processing of top-k ranking queries [18, 19], and that since then has been widely used for retrieving data from multiple sources. According to Fagin's framework there are d sorted lists $L_i, i = 1, \ldots, d$, with the i-th list ordered by non-increasing values of attribute A_i, and all lists managing the same set of n objects $O_1, \ldots, O_n$. A d-way 1-1 join on the object identifiers then allows the instance r to be reconstructed. Lists can be accessed either requesting the next element in the order (*sorted access*) or by providing an identifier and requesting the associated attribute value (*random access*). The basic steps of BDS are summarized in Algorithm 2.

The condition that halts the first phase is based on monotonicity (if an object O has not been seen in any list, then it is $\forall i.O_s[A_i] \geq_{A_i} O[A_i]$, thus $O_s \succ^{pto} O$), and is already found in [8]. There, the above algorithm is considered for computing the skyline using d B^+-trees, i.e., a different scenario but with the same access model of BDS.

EXAMPLE 4.1. *Consider the example in Table 2, in which there are $d = 3$ sorted lists and* $\text{sky}_{\mathcal{A}}(r) = \{O_2, O_7\}$. *After retrieving the first 3 objects from each list BDS can halt, since all unseen objects, like O_1, are dominated by O_2.*

Algorithm 2 Basic distributed skyline algorithm

Input: instance r vertically partitioned in d sorted list $L_1, \ldots, L_d$
Output: $sky_{\mathcal{A}}(r)$
1: Cyclically perform sorted accesses on the d lists until (at least) one object, say O_s, is retrieved from *all* the lists;
2: For all objects O that have been fetched from *at least one* list, perform random accesses to retrieve the missing attribute values;
3: Perform the necessary dominance tests and return the nondominated objects.

oid	A_1
O_7	0.9
O_3	0.6
O_2	0.6
O_1	0.5
O_4	0.4

oid	A_2
O_2	0.9
O_3	0.7
O_4	0.6
O_1	0.5
O_7	0.5

oid	A_3
O_7	1.0
O_2	0.8
O_4	0.7
O_3	0.7
O_1	0.6

Table 2: A vertically-partitioned relation

The halt condition coincides with the one used in the A_O algorithm by Fagin [18] for computing the top-1 object according to an arbitrary monotone scoring function. In light of Theorem 3.1, this should not be surprising at all. This fact also implies that the analysis in [18] applies to skyline computation, which with arbitrarily high probability, and assuming attribute independence, will therefore require $\mathcal{O}(n^{1-1/d})$ accesses to the lists for any fixed value of d.

Horizontal fragmentation. When a relation r is horizontally fragmented over a cluster of P servers, $r = r_1 \cup \ldots \cup r_P$, the skyline can be computed by exploiting the identity

$$sky_{\mathcal{A}}(r) = sky_{\mathcal{A}}(sky_{\mathcal{A}}(r_1) \cup \ldots \cup sky_{\mathcal{A}}(r_p)),$$

i.e., by first computing the *local* skylines and then merging the results, as the divide & conquer approaches described in Section 4.2 do. As argued in [1], this simple scheme has the drawback of requiring $\log P$ communication steps, i.e., its *synchronization complexity* is high and might easily become the main performance bottleneck. The (first) algorithm described in [1] requires only 2 communication steps and is also perfectly load-balanced, in that each server has maximum load $\mathcal{O}(dn/P)$.

The algorithm starts from an arbitrary data allocation, in which each server s manages a local fragment r_s of n/P tuples. In the initial preprocessing phase the servers cooperate to build a grid of P^d cells, which requires a total of $dP(P+1)$ values (partition points) to be transmitted over the network (note that this is independent of n). A key property of the multidimensional grid, in which each coordinate is partitioned into P buckets, is that each bucket is guaranteed to contain $\mathcal{O}(n/P)$ points. Assuming J is the set of nonempty cells, the preprocessing phase also includes the broadcast of which cells are in J (there are P^{d+1} values overall). Knowing J immediately allows the points within cells that are *strictly dominated* by some other cell in J to be discarded, where cell C strictly dominates cell D if on each coordinate C has a value strictly better than D. All cells in J that are not strictly dominated form the so-called *relaxed skyline* of r, $S_r(J)$, and they are guaranteed to contain all the points in $sky_{\mathcal{A}}(r)$.

If C is a cell in the relaxed skyline, and $t \in C$, then t can only be dominated by points in the cells of $S_r(J)$ that dominate C. A key observation is that these are exactly those cells D that share with C at least one coordinate value (thus, on at least one dimension they belong to the same bucket), and have better values in the other coordinates. Based on this observation, it is derived that the total number of points that can dominate any point t in a cell C is $\mathcal{O}(dn/P) = \mathcal{O}(n/P)$, i.e., an amount of data that could be processed by a single server. However, since the number of cells in the relaxed skyline can be in the order of $\mathcal{O}(P^{d-1})$, it is unfeasible to look for skyline points on a cell-by-cell basis.

The first step of the algorithm in [1] assigns to each server s the task of computing the local skylines $sky_{\mathcal{A}}(r_{i,s})$, $i = 1, \dots, d$, where $r_{i,s} = \{t \in r | t \in C = (C_1, \dots, C_d) \wedge C \in S_r(J) \wedge C_i = s\}$ is the set of points mapped to those cells C of the relaxed skyline that on the i-th coordinate fall in bucket s. The identity

$$r_1 \cap r_2 \cap sky_{\mathcal{A}}(r_1 \cup r_2) = sky_{\mathcal{A}}(r_1) \cap sky_{\mathcal{A}}(r_2)$$

is then used to compute the result, the intuition being that if $t \in C$, then $t \in sky_{\mathcal{A}}(r)$ iff it is in the local skylines $sky_{\mathcal{A}}(r_{i,C_i})$, $i = 1, \dots, d$. The second step of the algorithm computes such intersections using a randomized load-balanced algorithm.

4.6 Further approaches

Several other algorithms have been developed for computing the skyline in distributed environments. In particular, both structured and unstructured peer-to-peer networks have been considered and specific techniques for processing skyline queries in such scenarios have been developed. A recent survey [27] enters into the details of these approaches.

5. SKYLINE CARDINALITY

5.1 Average skyline size

Estimating the expected skyline size for a given data distribution is a key issue for the design of good cost models for skyline queries. The problem has been addressed in several papers [7, 9, 22]; here we report some major results.

Denote by $\ell_{d,n}$ the expected size of the skyline for a randomly sampled dataset of n points in d dimensions. Under the assumption that (i) all the attributes are statistically independent of each others, and (ii) the probability of sampling the same attribute value twice is negligible, the following recurrence holds:

$$\ell_{d,n} = \ell_{d,n-1} + \frac{1}{n} \cdot \ell_{d-1,n} \tag{1}$$

where the base case is $\ell_{d,1} = 1$.
If we assume that any two different tuples in r cannot share the same value for the same attribute (i.e., no two tuples can be projected to a single point, in any dimension), then the skyline of r is fully determined by the order in which tuples appear in each possible projection over a skyline attribute, and so is the skyline size. Without lack of generality we can substitute each attribute value with its rank, as determined by the corresponding projection. Therefore, a random relation instance r that respects the conditions (i) and (ii) can be seen as a set of d statistically independent random orders. For any r we know there is exactly one tuple having the worst rank w.r.t. the first attribute A_1. Denote by t^* this tuple. Since t^* cannot dominate any other tuple in r, $\ell_{d,n}$ is given by the expected number of skyline points in $r - \{t^*\}$ (i.e., $\ell_{d,n-1}$) plus the probability that t^* is a skyline point itself. In order to be nondominated in r, t^* needs to be a skyline point w.r.t. attributes in $\mathcal{A} - \{A_1\}$. The expected number of such skyline points is $\ell_{d-1,n}$, and every tuple in r has the same probability of being one of those.

[7] showed that $\ell_{d,n}$ is $\mathcal{O}((\ln n)^{d-1})$; later [9] provided the closed form

$$\ell_{d,n} = \sum_{k=1}^{n} (-1)^{k+1} \cdot \binom{n}{k} \frac{1}{k^{d-1}} \tag{2}$$

and proved the tighter bound $\Theta((\ln n)^{d-1}/(d-1)!)$.

When the assumption (ii) above is dropped, i.e., the same attribute value can appear multiple times in r, the analysis becomes more complex and leads to some unexpected results [22]. A first observation is that the distribution of values over their domains now matters, with deviations from the uniform case that lead to reducing the skyline size. A second phenomenon can be observed when values are *binned* and all values within the same bin are considered to be equal when testing dominance (which is equivalent to changing the size of the attribute domains). If $t \succ^{pto} s$ *before* binning, then reducing the domain size can lead to $t \not\succ^{pto} s$ only if the two tuples come to share the same bin on *all* the coordinates, i.e., they become equal on all skyline attributes. On the other hand, if $t \not\succ^{pto} s$ before binning, it is possible to have $t \succ^{pto} s$ (or $s \succ^{pto} t$) if t and s fall in the same bin on all the coordinates in which t (s, respectively) was worse. The larger is the number of dimensions d, the more the second effect will prevail over the first, unless the number of bins is very small. Thus, limiting the size of attributes' domains generally reduces the skyline size (since more tuples will be dominated).

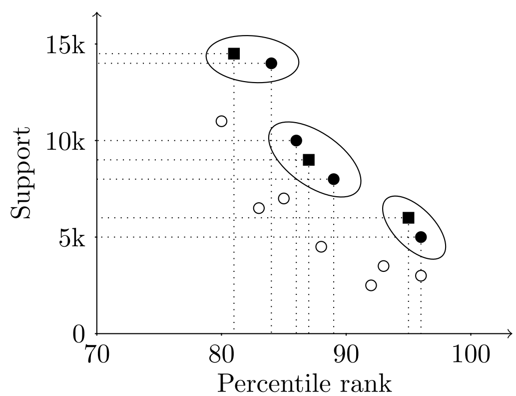


Figure 3: Diverse skyline. Most representative items are depicted as squares; ovals pair each skyline point with the closest representative item.

5.2 Representative skylines

As can be seen from the earlier discussion, skylines can be very large. In the worst case, a skyline can be as big as its originating dataset; in the average case, the size of the skyline grows with the number of its dimensions. There are several situations where dealing with a huge skyline may be impractical: the users may not be able or willing to browse through thousands of tuples; also, the time spent delivering the results may adversely affect the general user experience. Furthermore, a large skyline hardly provides any help to a user making a decision. Hence, in many real-life applications, we would like to return only a limited number κ of results that are representative of the whole skyline.

Diverse skylines. The problem of identifying the κ most representative items of the skyline has been addressed in several papers. Tao et al. [46] propose to model it as the κ-*center* problem: they suggest to select representative items by minimizing the *representation error*, i.e. the maximum Euclidean distance between a skyline point and its closest representative. The subset of points obtained this way is called a *diverse skyline*. Even if computing a diverse skyline with more than two dimensions is NP-hard, a simple greedy algorithm, like the one proposed by Gonzalez [25], can provide a 2-approximate solution in $\mathcal{O}(\kappa \cdot |sky_{\mathcal{A}}|)$. By using the Euclidean distance as a metric for measuring the similarity between skyline points, this approach implicitly assumes that all dimensions are equally important (for example: one unit of financial support is as important as one unit in the universities ranking scale). As a consequence, diverse skylines are not invariant w.r.t. scaling. Figure 3 shows the $\kappa = 3$ most representative items from the `School` relation of Example 1.1.

Top-κ RSP. An alternative approach is proposed by Lin et al. [36]: a set of κ representative items is selected in order to maximize the number of data points that are dominated by at least one of them. That is, we want to select the κ elements of the skyline that together cover the largest part of the dataset. The computation of top-κ representative skyline points (top-κ RSP) is NP-hard even in three dimensions; nevertheless, it can be easily translated into the *maximum coverage* problem and an $(1-\frac{1}{e})$-approximate solution can be obtained by applying the corresponding greedy algorithm. Top-κ RSP are invariant w.r.t. scaling and shifting operations, but they strongly depend on the points that are not part of the skyline.

Threshold-based preferences. Das Sarma et al. [43] exploit additional user preferences for selecting the most representative skyline points. They assume each user expresses her willingness to click on a particular result using threshold constraints; they also propose a probabilistic framework for modeling these preferences. In the simple case of deterministic preferences, a threshold can be represented as a point in the data space: for example, a student may be interested only in those universities that are in the 80th percentile and offer at least 20k dollars of financial support. It is easy to see that a skyline point satisfies a threshold if it dominates the corresponding point. Therefore, selecting κ skyline items that maximize the number of thresholds covered is a problem very similar to computing top-κ RSP. The same properties for invariance and complexity hold for both the approaches.

6. BEYOND PARETO DOMINANCE

6.1 Generalizations

The concept of Pareto dominance is quite restrictive. Presumably, users would like to have a richer language for formulating their preferences. Also, generalizing Pareto dominance should yield stronger dominance relations, and thus, fewer nondominated tuples and smaller query results. Below, we describe several such generalizations [41].

Grouped dominance. This variant of Pareto dominance requires that subspace dominance with respect to a set of attributes X_1 be applied only to the tuples with identical values in a set of attributes X_2, disjoint from X_1. This achieves the effect of grouping the tuples by X_2. Dominance and skylines are defined group-by-group.

k-dominance. Another variant of subspace dominance does not fix the subspace but rather considers all subspaces with cardinality k (usually $k < d$). For k-dominance of a tuple t over another tuple s it is sufficient that $t[A_{j_1} \cdots A_{j_k}]$ Pareto-dominates $s[A_{j_1} \cdots A_{j_k}]$ over *some* attributes $A_{j_1} \cdots A_{j_k}$. This concept of dominance is especially suited to applications with a very large number of dimensions, as in recommender systems (each user is a separate dimension). There, Pareto dominance (over all attributes) may occur rarely, while dominance over only some k attributes may be more common and thus more useful.

p-dominance [31, 38]. The notion of p-dominance (where "p" stands for "prioritized") builds on the algebraic definition of Pareto dominance. A different binary accumulation operator, &, is proposed. It is supposed to capture the relative importance of different attributes. p-dominance relations can now be defined using an arbitrary nesting of $\otimes$ and &. Let $\succ_X$ (resp. $\succ_Y$) be a p-dominance relation over a set of attributes X (resp. Y). Then their prioritized accumulation $\succ^{pr}_{XY} = (\succ_X \;\&\; \succ_Y)$ is defined as

$$t[XY] \succ^{pr}_{XY} s[XY] \;\; \equiv \;\; t[X] \succ_X s[X] \;\; \vee \;\; t[X] = s[X] \wedge t[Y] \succ_Y s[Y]$$

for $XY \subseteq \mathcal{A}$ and $X \cap Y = \emptyset$. Note that

$$\succ^{pr}_{\mathcal{A}} = >_{A_1} \& >_{A_2} \& \cdots \& >_{A_d}$$

is a *lexicographic* order over $\mathcal{U}$.

Properties. Clearly, subspace dominance, grouped dominance and p-dominance are irreflexive and transitive. However, k-dominance, being also irreflexive, is not transitive in general.

EXAMPLE 6.1. *Assuming larger values are better, the tuple $t_1 = (1, 2)$ 1-dominates the tuple $t_2 = (2, 1)$, which in turn 1-dominates $t_3 = (1, 3)$. However, t_1 does not 1-dominate t_3.*

6.2 Dominance in other spaces

Pareto dominance is often used to define dominance in a different space. We consider here dynamic and aggregate skyline queries.

Dynamic skyline queries [41]. Assume there are m functions $f_1, \ldots, f_m$, each defined over some subset of $\mathcal{A}$. We can construct a transformed tuple $f(t) = (f_1(t), \ldots f_m(t))$ for every tuple t. Now t dominates s if $f(t)$ Pareto-dominates $f(s)$. A common application involves functions capturing the 2D distance of a moving point from some fixed locations: a point x dominates another point y if for every given location z, x is not farther from z than y is.

Aggregate skyline queries. Tuples of function values (profiles) provide also a succinct way to represent aggregate properties of sets, e.g. cardinality or minimum value. The functions map sets to scalar values. The space of sets may consist of all k-element subsets of a given set of tuples [28, 35, 52], or all the tuple groups in a set of tuples [2]. Now a set T dominates another set S if the profile of T Pareto-dominates the profile of S. This approach to set dominance can be generalized to arbitrary dominance relations [52].

7. FURTHER DIRECTIONS

7.1 Skylines for uncertain data

Generalizations of the concepts of Pareto dominance and skyline to the case of uncertain databases have recently been attempted according to three different approaches, which we briefly describe here. Common to all of them is the underlying model of uncertainty, based on *probabilistic tuples* and *possible world semantics*. In short, each tuple $t \in r$ has an associated existence probability, $p(t)$, and correlations among tuples are captured by a set $\mathcal{G}$ of *generation rules*. The probabilistic relation r is seen as representing a set of standard relations, each of them termed a possible world. A possible world W is a subset of tuples from r, that respect all the generation rules in $\mathcal{G}$, and its probability $\Pr(W)$ is the probability that all and only the tuples in W indeed exist.

Both [42] and [50] view r as representing a set of uncertain objects, $O_1, \ldots, O_m$, and for each object O_i there is a mutual exclusion rule $G_i \in \mathcal{G}$ stating which are the tuples representing O_i's distribution. Given a possible world W and a tuple $t \in W$, one can determine whether $t \in sky_{\mathcal{A}}(W)$ or not. Consequently, [42] defines the *skyline probability* of t as

$$\Pr_{sky}(t) = \sum_{W:t \in sky_{\mathcal{A}}(W)} \Pr(W),$$

and that of an object O as the sum of the probabilities of its tuples, $\Pr_{sky}(O) = \sum_{t_j \in O} \Pr_{sky}(t_j)$. The p-skyline of r, where p is a threshold probability, is the set of objects O such that $\Pr_{sky}(O) \geq p$. This approach has also been adopted by [6] in the context of recommender systems based on collaborative filtering.

The approach of [50] is based on the concept of *usual stochastic order*: given two random variables O_1 and O_2, O_1 stochastically dominates O_2 if for each point x in the domain of definition, the cumulative distribution of O_1 at x, $O_1.cdf(x)$, satisfies $O_1.cdf(x) \geq O_2.cdf(x)$, with strict inequality for at least one x. In the multidimensional case and discrete distributions, the cumulative distribution of object O at point x is the sum of probabilities of the instances (i.e., tuples) of O that dominate x (or coincide with it). [50] also describes an alternative approach based on the lower orthant order.

Finally, [5] introduces the concept of P-domination, i.e., domination among probabilistic tuples, and consequently defines the skyline of a probabilistic relation r as the set of those tuples that are not P-dominated. The idea builds on works dealing with the ranking of probabilistic tuples for answering top-k queries, e.g., [51]. There, one considers that the tuples are ordered by a monotone scoring function f and a specific ranking semantics is adopted to properly combine scores and probabilities. For a given ranking semantics (that is a parameter for P-domination) one stipulates that t P-dominates s if t is ranked not worse than s whatever the monotone scoring function f is.

7.2 Elicitation

A skyline query requires minimal user input: a designation of the skyline space $\mathcal{A}$ and the associated attribute domains. The orderings associated with the attributes are standard. If a user provides more information, however, it is possible to construct queries that more completely reflect her intentions.

[29] propose to use superior and inferior examples to determine missing attribute orderings in an incomplete Pareto dominance specification. The superior examples POS are the tuples that have to be in the skyline, and the inferior examples NEG, those that should not be in the skyline because of being dominated by a skyline element. The authors show that determining the existence of a strict partial order (resp. minimal strict partial order) on an attribute domain that satisfies the above POS/NEG conditions is NP-complete (resp. NP-hard). They provide greedy heuristic algorithms, without any guarantees on their performance.

[38] also use superior and inferior examples, but for a different purpose. Assuming that attribute orderings are given, their approach seeks to find a maximal p-dominance relation that satisfies the above POS/NEG conditions. They show that the associated existence problem is NP-complete but becomes polynomial if only superior examples are given.

It remains to be seen if richer forms of user input, for example dominance relationships (*does tuple t dominate tuple s?*), could be used for eliciting complete specification of dominance relations. This is a problem similar to *learning orders from examples* [14].

8. CONCLUSIONS

We believe that skyline queries provide a useful, practical, and flexible query framework for decision-making applications dealing with large amounts of data. In this paper, we have tried to showcase some of the many technical results obtained in this area. Clearly, many further research opportunities still exist or remain to be identified. For example, the connections to decision theory largely remain to be explored. Although it was not the main focus of the paper, skyline queries are a specific yet important case of preference queries, on which a large body of interdisciplinary literature exists [10, 20, 30, 31].

Among the many topics that we were unable to cover here are reverse skylines [15], skylines over joins [49], stream skylines [47], spatial skylines over moving objects [44], skylines with trade-offs [37], approximate skylines [32], and skylines in metric spaces [21]. Together with the topics discussed in this paper, they bear witness to the continuing vitality of the research on skyline queries.

9. ACKNOWLEDGMENT

The detailed comments of the anonymous reviewer are gratefully acknowledged.

10. REFERENCES

[1] F. Afrati, P. Koutris, D. Suciu, and J.D. Ullman. Parallel Skyline Queries. In *International Conference on Database Theory (ICDT)*, pages 274–284, 2012.

[2] S. Antony, P. W, D. Agrawal, and A. El Abbadi. Aggregate Skyline: Analysis for Online Users. In *Symposium on Applications and the Internet (SAINT)*, pages 50–56. IEEE, 2009.

[3] W-T. Balke, U. Güntzer, and J. X. Zhang. Efficient Distributed Skylining for Web Information Systems. In *International Conference on Extending Database Technology (EDBT)*, pages 256–273, 2004.

[4] I. Bartolini, P. Ciaccia, and M. Patella. Efficient Sort-Based Skyline Evaluation. *ACM Transactions on Database Systems*, 33(4), 2008.

[5] I. Bartolini, P. Ciaccia, and M. Patella. The Skyline of a Probabilistic Relation. *IEEE Transactions on Knowledge and Data Engineering*, 25(7):1656–1669, 2013.

[6] I. Bartolini, Z. Zhang, and D. Papadias. Collaborative Filtering with Personalized Skylines. *IEEE Transactions on Knowledge and Data Engineering*, 23(2):190–203, 2011.

[7] J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. On the Average Number of Maxima in a Set of Vectors and Applications. *Journal of the ACM*, 25(4):536–543, 1978.

[8] S. Börzsönyi, D. Kossmann, and K. Stocker. The Skyline Operator. In *IEEE International Conference on Data Engineering (ICDE)*, pages 421–430, 2001.

[9] C. Buchta. On the Average Number of Maxima in a Set of Vectors. *Information Processing Letters*, 33:63–65, 1989.

[10] J. Chomicki. Preference Formulas in Relational Queries. *ACM Transactions on Database Systems*, 28(4):427–466, December 2003.

[11] J. Chomicki. Semantic Optimization Techniques for Preference Queries. *Information Systems*, 32(5):660–674, 2007.

[12] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with Presorting. In *IEEE International Conference on Data Engineering (ICDE)*, pages 717–719, 2003.

[13] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with Presorting: Theory and Optimizations. In *Intelligent Information Systems*, pages 595–604. Springer, Advances in Soft Computing, 2005. Full version with proofs: Technical Report CS-2002-04, October 2002, Department of Computer Science, York University.

[14] W. W. Cohen, R. E. Schapire, and Y. Freund. Learning to Order Things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

[15] R. Dellis and B. Seeger. Efficient Computation of Reverse Skyline Queries. In *International Conference on Very Large Data Bases (VLDB)*, pages 291–302, 2007.

[16] M. Ehrgott. *Multicriteria Optimization*. Springer, 2nd edition, 2005.

[17] J. Pei et al. Towards Multidimensional Subspace Skyline Analysis. *ACM Transactions on Database Systems*, 31(4):1335–1381, 2006.

[18] R. Fagin. Combining Fuzzy Information from Multiple Systems. *Journal of Computer and System Sciences*, 58(1):83–99, 1999.

[19] R. Fagin, A. Lotem, and M. Naor. Optimal Aggregation Algorithms for Middleware. *Journal of Computer and System Sciences*, 66(4):614–656, 2003.

[20] P. C. Fishburn. Preference Structures and their Numerical Representations. *Theoretical Computer Science*, 217:359–383, 1999.

[21] D. Fuhry, R. Jin, and D. Zhang. Efficient Skyline Computation in Metric Space. In *International Conference on Extending Database Technology (EDBT)*, pages 1042–1051, 2009.

[22] P. Godfrey. Skyline Cardinality for Relational Processing. In *International Symposium on Foundations of Information and Knowledge Systems (FOIKS)*, pages 78–97. Springer-Verlag 2942, 2004.

[23] P. Godfrey, R. Shipley, and J. Gryz. Maximal Vector Computation in Large Data Sets. In *International Conference on Very Large Data Bases (VLDB)*, pages 229–240, 2005.

[24] P. Godfrey, R. Shipley, and J. Gryz. Algorithms and Analyses for Maximal Vector Computation. *VLDB Journal*, 16:5–28, 2007.

[25] T. F. Gonzalez. Clustering to Minimize the Maximum Intercluster Distance. *Theoretical Computer Science*, 38:293–306, 1985.

[26] B. Hafenrichter and W. Kießling. Optimization of Relational Preference Queries. In *Australasian Database Conference (ADC)*, pages 175–184, 2005.

[27] K. Hose and A. Vlachou. A Survey of Skyline Processing in Highly Distributed Environments. *The VLDB Journal*, 21(3):359–384, 2012.

[28] H. Im and S. Park. Group Skyline Computation . *Information Sciences*, 188(0):151 – 169, 2012.
[29] B. Jiang, J. Pei, X. Lin, D. W. Cheung, and J. Han. Mining Preferences from Superior and Inferior Examples. In *KDD*, pages 390–398, 2008.
[30] S. Kaci. *Working with Preferences: Less is More*. Springer, 2011.
[31] W. Kießling. Foundations of Preferences in Database Systems. In *International Conference on Very Large Data Bases (VLDB)*, pages 311–322, 2002.
[32] V. Koltun and C. H. Papadimitriou. Approximately Dominating Representatives. *Theoretical Computer Science*, 371(3):148 – 154, 2007.
[33] H. T. Kung, F. Luccio, and F. P. Preparata. On Finding the Maxima of a Set of Vectors. *Journal of the ACM*, 22(4):469–476, 1975.
[34] K.C.K. Lee, B. Zheng, H. Li, and W.-C. Lee. Approaching the Skyline in Z Order. In *International Conference on Very Large Data Bases (VLDB)*, pages 279–290, 2007.
[35] C. Li, N. Zhang, N. Hassan, S. Rajasekaran, and G. Das. On Skyline Groups. In *International Conference on Information and Knowledge Management (CIKM)*, pages 2119–2123, 2012.
[36] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang. Selecting Stars: the k Most Representative Skyline Operator. In *IEEE International Conference on Data Engineering (ICDE)*, pages 86–95, 2007.
[37] C. Lofi, U. Güntzer, and W-T. Balke. Efficient Computation of Trade-Off Skylines. In *International Conference on Extending Database Technology (EDBT)*, pages 597–608, 2010.
[38] D. Mindolin and J. Chomicki. Preference Elicitation in Prioritized Skyline Queries. *VLDB Journal*, 20(2):157–182, 2011. Special issue: selected papers from VLDB 2009.
[39] M.D. Morse, J.M. Patel, and H.V. Jagadish. Efficient Skyline Computation over Low-Cardinality Domains. In *International Conference on Very Large Data Bases (VLDB)*, pages 267–278, 2007.
[40] D. Papadias, Y. Tao, G. Fu, and B. Seeger. An Optimal and Progressive Algorithm for Skyline Queries. In *ACM SIGMOD International Conference on Management of Data*, pages 467–478, 2003.
[41] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive Skyline Computation in Database Systems. *ACM Transactions on Database Systems*, 30(1):41–82, 2005.
[42] J. Pei, B. Jiang, X. Lin, and Y. Yuan. Probabilistic Skylines on Uncertain Data. In *VLDB*, pages 15–26, 2007.
[43] A. Das Sarma, A. Lall, D. Nanongkai, R. J. Lipton, and J. Xu. Representative Skylines Using Threshold-Based Preference Distributions. In *IEEE International Conference on Data Engineering (ICDE)*, pages 387–398, 2011.
[44] M. Sharifzadeh and C. Shahabi. The Spatial Skyline Queries. In *International Conference on Very Large Data Bases (VLDB)*, pages 751–762, 2006.
[45] C. Sheng and Y. Tao. Finding Skylines in External Memory. In *ACM Symposium on Principles of Database Systems (PODS)*, pages 107–116, 2011.
[46] Y. Tao, L. Ding, X. Lin, and J. Pei. Distance-Based Representative Skyline. In *IEEE International Conference on Data Engineering (ICDE)*, pages 892–903, 2009.
[47] Y. Tao and D. Papadias. Maintaining Sliding Window Skylines on Data Streams. *IEEE Transactions on Knowledge and Data Engineering*, 18(3):377–391, 2006.
[48] G. Trimponias, I. Bartolini, D. Papadias, and Y. Yang. Skyline Processing on Distributed Vertical Decompositions. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):850–862, 2013.
[49] A. Vlachou, Ch. Doulkeridis, and N. Polyzotis. Skyline Query Processing over Joins. In *ACM SIGMOD International Conference on Management of Data*, pages 73–84, 2011.
[50] W. Zhang, X. Lin, Y. Zhang, M. A. Cheema, and Q. Zhang. Stochastic Skylines. *ACM Transactions on Database Systems*, 37(2), 2012.
[51] X. Zhang and J. Chomicki. Semantics and Evaluation of Top-k Queries in Probabilistic Databases. *Distributed and Parallel Databases*, 26(1):67–126, 2009.
[52] X. Zhang and J. Chomicki. Preference Queries over Sets. In *IEEE International Conference on Data Engineering (ICDE)*, pages 1019–1030, April 2011.

Towards Mega-Modeling: A Walk through Data Analysis Experiences

Stefano Ceri
DEIB – Politecnico di Milano
stefano.ceri@polimi.it

Emanuele Della Valle
DEIB – Politecnico di Milano
emanuele.dellavalle@polimi.it

Johann-Christoph Freytag
DBIS - Humboldt-Universität zu Berlin
freytag@informatik.hu-berlin.de

Themis Palpanas
DISI – University of Trento
themis@disi.unitn.eu

Dino Pedreschi
KDD Lab - Università di Pisa
pedre@di.unipi.it

Roberto Trasarti
KDD Lab – ISTI,CNR Pisa
roberto.trasarti@isti.cnr.it

1. INTRODUCTION

Big data is perceived as a fundamental ingredient for fostering the progress of science in a variety of disciplines. However, we believe that the current ICT solutions are not adequate for this challenge. Abstractions and languages for big data management are tailored to vertical domains and influenced by underlying ICT platforms, hence unsuitable for supporting "computational interdisciplinarity", as it is required if one wants to use the best of, e.g., analytical, inductive, and simulation techniques, all at work on the same data. In other words, "our society is data-rich, but it lacks the conceptual tools to handle it" [1].

In previous work [2], we advocate the need for a new approach to data analysis, based on mega-modeling as a new holistic data and model management system for the acquisition, composition, integration, management, querying and mining of data and models, capable of mastering the co-evolution of data and models and of supporting the creation of what-if analyses, predictive analytics and scenario explorations.

In this paper, we provide some evidence that mega-modeling is a viable approach to data analysis by using a bottom-up, inductive method. We consider several experiences of data analysis research performed at our home institutions and examine them in retrospective, inducing their mega-modularization a-posteriori. This exercise convinces us that the mega-modeling approach could be highly beneficial.

2. MEGA-MODELING

In this section, we provide a historical perspective on the development of the mega-modeling concept and a self-contained summary of our previous work on Mega-modeling [2].

2.1 Unifying Data-centric Disciplines

Big Data analytics calls for *a comprehensive theory and technology that blend simulation, analytical, ontological and data-driven models into one picture*. Modeling, as we know it today, is required to scale up to a higher level, that we call **mega-modeling** [2].

The database/data mining community has been investigating approaches that unify data analysis and mining, since the seminal paper on *inductive databases* and *data mining as a querying process* by Imielinski and Mannila [11]. A fundamental aspect of [11] is the representation of data mining activities through patterns, whereas patterns can be seamlessly integrated with data and can therefore be the subject of queries; such view attempts a conceptualization and generalization of data mining. Yet, this idea has found concrete realizations only lately and partially; e.g., it is used in [12], where extracted data patterns are defined as views on top of data tables, and as such can be composed with domain-specific data representing spatio-temporal trajectories expressing human mobility [12].

The roots of mega-modeling can be traced to an article appeared in 1992 on "Mega-programming" [13] – *large, autonomous computing systems whose interfaces are described through a data-centric approach and whose execution behaviour can be inspected* – and to another one that appeared about one decade later on "mega-models" [14] – models of which at least some elements represent and/or refer to other models or meta-models – that paved the road to the school on Model-Driven Engineering (MDE) [15].

However, the innovative aspects of mega-modeling go beyond classical model-driven software generation. Mega-modeling aims at defining a comprehensive theory and technology of model construction (with an emphasis on incremental bottom-up approaches), model search, model fitness evaluation, model composition, model reuse and model evolution. We need entirely new *models of models*, namely algebras of objects representing patterns, rules, laws, equations, etc., which are either mined/induced from data, or based on deep mathematical findings or agent-based reasoning – an overarching algebra of data and models that allow us to devise a new holistic system for integrated data and model acquisition, integration, querying and mining, capable of mastering the complexity of the knowledge discovery process.

2.2 Mega-Modules

The building block of mega-modeling is the **mega-module** – a software component capable of processing "big data" for analytical purposes. Every mega-module performs a well identified computation, which can be considered a unitary transformation from inputs to outputs. Inputs and outputs take the form of data and of patterns, where data are domain-specific both in terms of their schema end instances, while patterns are forms of data regularity or rules whose schema is domain-independent and whose content typically reflects collective or aggregated data properties; patterns may be extracted by data analysis algorithms, which may in turn be embodied within mega modules. Every mega-module can internally use data and patterns that are considered as invariant in the context of the computation, whose extension can be either local (e.g., organization-specific) or global (e.g., stored in public databases or ontologies).

Every mega-module exhibits a format that consists of three distinct phases of information processing, although such phases can vary significantly for their internal organization: data preparation, analysis, and evaluation.

- The first phase, **data preparation**, consists of the processing of input data and patterns for the purpose of assembling input objects that will be the subjects of the analysis. The distinction between data and object is of semantic nature: data preparation typically assembles several elementary data in the input to generate a single object for the purpose of analysis. The aggregative process that builds a object can be driven by a variety of purposes – abstracting irrelevant differences, recognizing common features, aggregating over elementary items which satisfy given predicates – thus, semantically interpreting and reconstructing data. The keywords for the preparation phase are: data sensing, acquisition, integration, transformation, semantic enrichment.
- The second phase, **data analysis**, consists of extracting computed objects from input data, possibly using the input patterns as references. Data analysis produces the response to a specific problem by performing the core scientific processing, and uses a variety of methods, ranging from mathematical to statistical models, from data mining to machine learning, from simulation to prediction, including crowd-sourcing as a way for asking social responses. The keywords for the analysis phase are: mining, learning, modeling, simulation, forecast.
- The third phase, **data evaluation**, consists of preparing the output objects, which may in turn be presented as data and/or patterns. This phase consists of filtering or ranking computed objects based on their relevance, and possibly of a post-processing so as to observe the result in the most suitable way for the mega-module enclosing environment or user. The keywords for the evaluation phase are: quality assessment, filtering, significance measurements, presentation, delivery, visualization.

In Fig. 1, we propose a mega-module graphical element, which visually captures the characteristics of a mega-module as described above.

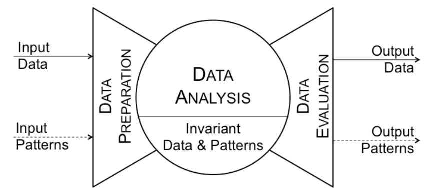


Figure 1: Visual presentation of a generic mega-module.

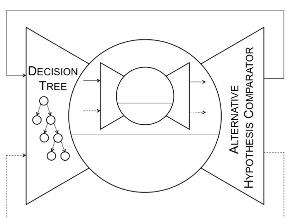
The presence of the three phases allows us to define two standard **inspection points** within a mega-module, used for asynchronous control and feedback that mega-modules should provide to their enclosing environment. The first one, after preparation, provides a view on objects abstracted/reconstructed from data; the second one, after analysis, provides a view of the objects resulting from the analysis.

A mega-module inspection consists in extracting its controls asynchronously, during its execution; this in turn allows the enclosing environment to trace mega-module execution, to estimate completion time, and to anticipate the quality of its results. We regard the data and patterns that may be exchanged by a mega-module during its execution as the **mega-module controls**. A mega-module should expose commands to the enclosing environment that may alter its behavior, for instance by rising or by lowering confidence levels during analysis based on the quality of intermediate results or on the expected completion time. It should also be possible to suspend, resume, and terminate the mega-module computation.

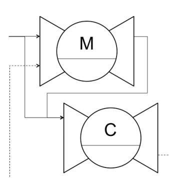
Wrapping up, we associate to each mega-module the potential of expressing classes of computations on top of big data, thereby highlighting the computational nature of the modules and the support of dynamic aspects related to inspection, adaptation, and integration. In the design or reverse engineering of Mega-Modules, data come first: clarifying their input and output data by using known pattern types is the key aspect for guaranteeing module interoperability and reuse. Emphasizing the role of data transformers for Mega-Modules opens up to using/inventing algebraic languages for data-driven orchestrations and optimizations. Moreover, the possibility of declaring the streaming or ordered nature of data opens up possibilities for a different class of optimizations that emphasize recent and ordered data.

2.3 Mega-modularizing Big Data Analysis

In mega-modeling, a big data analysis problem is modeled as a data-driven workflow involving several Mega-Modules. Composition abstractions are the means of combining mega-modules to the purpose of creating sophisticated analytical processes. Composition abstractions reflect the classical ways of assembling modules into higher order computations. Every abstraction induces a hierarchical decomposition, singling out an enclosing mega-module and one or more enclosed mega-modules; our goal is to describe computations over big data as top-down recursive applications of a well-designed collection of abstractions. In [2], we present an initial set of abstractions; they are orthogonal, but most likely incomplete, and further investigation is needed to consolidate them. The set includes traditional pipeline and parallel composition, but also typical data mining ones such as what-if control, drift control and component-based graph decomposition.

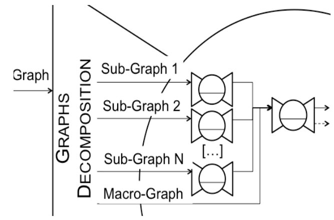


What-if control is a classical way of mining big data by exploring many alternative solutions that would occur for different choices of initial setting of models and/or parameters. Essentially, this control abstraction is a form of iteration driven by an analytical goal, allowing to repeat a mega-module under different parameterizations of input data and patterns, until a final analytical result is obtained, which possesses a desired level of, e.g., quality, precision or statistical significance; the preparation phase can be modeled by a decision tree. Many possible instances of this "what-if" iteration control may be envisage, pertaining to many existing alternatives for exploring a space of patterns/models studied, e.g., in machine learning, data mining, statistical physics and (agent-based) simulation.



Many mega-module computations are based upon the validity of underlying assumptions. Thus, if the assumptions cease to be valid, the mega-module itself must be invalidated, and then either corrected or abandoned. For instance, a credit risk predictor used by a bank for granting mortgages may become obsolete as an effect of an economic crisis that impact household incomes. The phenomenon of "drifting" describes the progressive invalidation of assumptions under which a model has been learned from data. A mega-module M, which is potentially subject to drifting, should be paired to an associated *drift-control* mega-module C, which assumes the output of M as input. The controller normally has no output, however if it perceives that the drift has occurred, then it interacts with M, by providing suitable controls.

Many mega-module computations apply to input data representing (large) networks and graphs thus making parallelization more difficult; if instead a graph has modular structure of components (namely sub-networks) with high intra-module connectivity and relatively low inter-module connectivity, then a natural parallelization can be achieved by mapping each sub-network to an internal mega-module before integrating the results using one additional combining mega-module. Such an approach enables a *component-based graph decomposition*.



Data-driven mega-modularization should also facilitate dynamic adaptation, performed by the invoking environment in the context of a mega-module orchestration. The presence of mega-module inspection points allows for asynchronously extracting parameters describing data analysis execution, while the execution is ongoing; in this way, the enclosing environment might include a **Mega-Module controller** which traces Mega-Module execution, estimates the completion time of data analysis, and anticipates the quality of its results. The controller may adaptively alter the behavior of a module, for instance by raising or by lowering the confidence levels that control the output production.

3. DATA ANALYSIS PROBLEMS

We reviewed seven recent research experiences, three in the mobility data context, and four in the data stream context.

- Problem P1: INDIVIDUAL PROFILING. Given spatio-temporal information, reconstruct trajectories and find trajectory clusters that correspond to routine daily commutes of individual citizens [3].
- Problem P2: COLLECTIVE PROFILING. Given spatio-temporal information, reconstruct trajectories and aggregate them to find typical traffic routes, each of them consisting of sequences of regions [4].
- Problem P3: REGION IDENTIFICATION. Given spatio-temporal information, reconstruct trajectories and map them to edges among cells of a spatial tessellation, then partition the resulting network of cells so as to recognize regions with high connectivity [5].
- Problem P4: TRACKING OF CROWD MOVES. Given streams of geo-tagged micro-posts (e.g., geo-tagged tweets, foursquare check-ins) from a geographic area, detect where crowds are assembling and show how they are moving using a stream of heatmaps [6].
- Problem P5: BURST OF INTEREST DETECTION. Given streams of micro-posts (e.g., tweets, facebook status), enrich them with semantic entities they talk about, before detecting bursts of interest w.r.t. the described entities [7].
- Problem P6: SENTIMENT SHIFTING. Given streams of micro-posts, extract topic(s) and the sentiment

relative to the topic(s) and monitor topic-sentiment pairs to detect sentiment shifts [8].

- Problem P7: DATA OUTLIER DETECTION. Given streams of data representing measures, summarize them by their probability density functions and then detect different kinds of outliers [9].

4. MOBILITY USE CASES

The common aspects of problems P1-P3 is the presence of trajectories as fundamental data pattern, and of trajectory reconstruction and clustering as fundamental computational steps.

4.1 Common Pattern Types

Our approach to the modeling of data analysis problems starts with the definition of the **pattern types,** i.e. generic types used for representing mobility [2]. Considering the three mobility problems, they all are based on a big dataset of observations of mobile objects, associated to their positions in space and time. Observations are assembled into trajectories that are sequences of observations of the same object, further characterized by the length and average speed. Given a set of trajectories, a medoid is the result of a statistical process that defines their median trajectory and variance. The corresponding pattern types are shown below using a simple formalism, where square brackets denote tuples, curly brackets denote sets, and the "<" ">" symbols denote lists.

Observation: [oid, position[latitude, longitude], time]
Trajectory: [tid, <Observation>, length, avgSpeed]
Medoid: [oid, <Observation>, Variance]

In addition to moving points, mobile applications also describe geographic regions, typically characterized by their geometry, which is a sequence of positions describing the region's border. Regions are typically related to each other in a network, which is a collection of nodes and arcs, where nodes are associated to regions and arcs connect two regions and are further characterized by a weight. The corresponding pattern types are below; note that pattern type denote minimal information and can be extended in each different application, e.g. regions may have a name and additional properties such as size and population.

Region: [rid, <position[latitude, longitude]>]
Network: [nid, {Region},{[Region, Region, weight]}]

All the problems essentially deal with trajectory assembling and managing. Similar trajectories can be grouped into clusters, and trajectories can be aggregated in space so as to connect regions rather than individual positions; such trajectories may be further characterized by their minimum and maximum time.

T-Cluster: [cid, {Trajectory}]
T- Pattern: [tid, <Region>, min-time, max-time>]

4.2 Models of Mobility Cases

We considered the three applications and observed that they share many aspects that can be modeled as chains of Mega-Module applications that progressively produce the relevant data.

4.2.1 Individual Profiling

Problem P1 is concerned with aggregating the trajectories of a single individual to capture her usual commutes; thus it requires combining Mega-Modules for reconstructing trajectories from observations, then to cluster the trajectories of each individual users, and then compute the medoid of each cluster and associate it with labels in order to extract opportunities for car-pooling. This chain of transformations is described in Fig. 2, which visually shows trajectory reconstruction, clustering, and profiling.

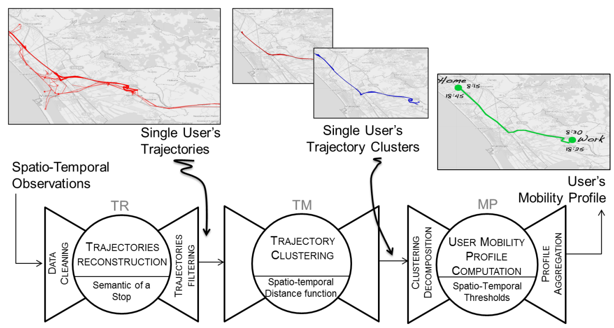


Figure 2: Problem P1

We next describe the Mega-Modules:

- **TRAJECTORY RECONSTRUCTION** (TR) – builds trajectories from observation of moving points, using a semantic description of a "stop". In the pre-processing, spatio-temporal observations received as input are cleaned, and the history of each user's movement is obtained by ordering their observations according to their time. At this point, the data analysis component processes each history by dividing it into several sub-sequences representing trajectories; each trajectory represents an individual user's trip. Data analysis is parametric and uses as input the definition of the "semantics of a stop", defined as conjunction of two spatio-temporal constraints: a minimum time span and a maximum distance between two consecutive points. Finally, the post-processing consists of filtering those trips that are not meaningful or contain outlier and anomalies that can be detected only at this level of abstraction, e.g., one-point or out-of-region trips.
- **TRAJECTORY CLUSTERING** (TM) – performs a density-based clustering of trajectory data. The data analysis is parametric, it uses as input a spatio-temporal function for computing distances between different trajectories. The result is a set of clusters of homogeneous (i.e., similar) trajectories. Trajectories may be associated with application-specific labels concerning their initial and final observations; in the example, we obtain two

groups of trajectories, tone moving from north to south and the other one moving from south to north.

- **MOBILITY PROFILE** (MP) – computes the medoids of each cluster. The pre-processing is used to partition the analysis process into separate threads and to filter those sets whose cardinality is below a given threshold. At this point, the data analysis component extracts the medoids according to a parametric distance function; each medoid is associated with a variance describing its statistical representativeness. Post-processing gathers all the results from the different execution threads, filters low-quality medoids and then constructs user's mobility profiles. In Fig. 2, two profiles are associated with a given user: (Home) 8:15 → (Work) 8:30 and (Work) 18:25 → (Work) 18:45, where "Home" and "Work" are two labels assigned to the points where their medoid trips begin and end, considering the hour of the day in which the trips occurs.

4.2.2 Collective Profiling
Problem P2 is concerned with understanding, in a broad sense, how people move across small-scale regions, thus building "important" trajectories that are relevant for further analysis (e.g. real-time traffic monitoring getting to specific locations). The first two steps for solving this problem turn out to be identical to the previous use case. However, the third step is quite different, as it involves a transformation of the most relevant trajectories from sequences of positions to sequences of regions.

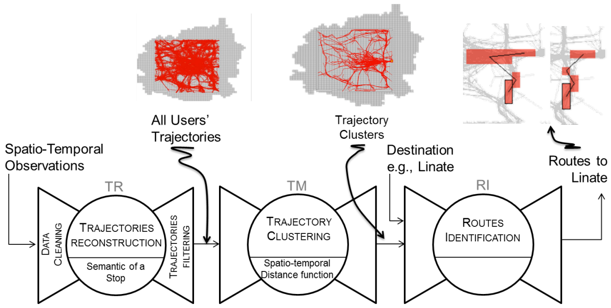


Figure 3: Problem P2

Thus, the second problem reuses the modules TR and TC and adds to them the Route Identification Mega-Module.

- **ROUTE IDENTIFICATION** (RI) - uses the trajectory clusters for mining the typical routes, represented as T-patterns. The pre-processing filters the clusters by keeping only those containing trajectories ending in a specific place specified by the application, e.g. Linate airport. Then, the data analysis component applies the T-Pattern discovery algorithm over each set of trajectories and extracts typical routes.

In the example illustrated in Fig. 3, two T-Patterns are found: the first one (on the left) represents the people arriving at Linate from the south exiting the highway at "Via Mecenate" and turning into "Viale Forlanini", while the second (on the right) represents the people arriving at Linate existing directly in "Viale Forlanini".

The former is often a smarter choice as the "Viale Forlanini" exit is often congested.

4.2.3 Region Identification
The third problem focuses on recognizing macro-regions consisting of regions that are strongly connected, i.e. such that most of traffic occurs inside the macro-region, while

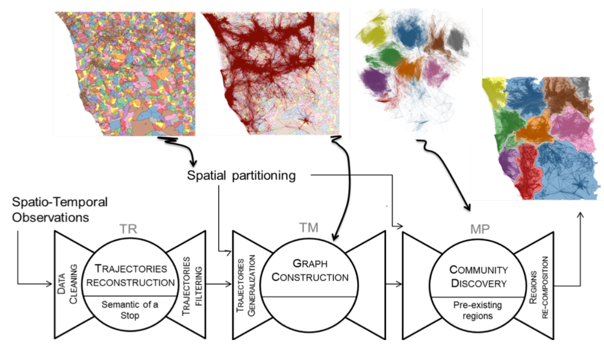


Figure 4: Problem P3

only a small amount of traffic moves between them. This problem also uses the Mega-Module for trajectory reconstruction, but it then characterizes trajectories as T-patterns traversing regions, and then describe the induced network of region connections in order to discover the macro-regions (see Fig. 4). Two new Mega-Modules are introduced:

- **GRAPH CONSTRUCTION** (GC) – transforms the trajectories into a set of sequences using the spatial tessellation as input. This is done by intersecting all the points of the trajectories with the cells and removing all the consecutive repetitions obtained. Thus, a trajectory is represented as the sequence of traversed cells without the temporal component. Then, each cell is mapped into a distinct node of the graph, and trajectories connecting two cells are mapped to edges; the weight of each edge is proportional to the number of trajectories.
- **COMMUNITY DISCOVERY** (CD) – uses Infomap algorithm, which is based on a combination of information-theoretic techniques and random walks. The result is a set of communities, each represented by a set of nodes. These sets are then remapped to the spatial dimension and joined obtaining the spatial borders of the communities.

5. TEXT STREAM USE CASES
Problems P4, P5, and P6 are based on independent experiences of the authors on text stream analytics. By studying them, we realized that they could be effectively modeled by an initial common sequence of Mega-Modules.

5.1 Common Pattern Types

The text items constituting the datasets consist of short texts associated with their authors and publication times, potentially containing a set of tags, and possibly annotated with a geo-position. Each short text is enriched by a set of topics – e.g. hashtags or semantic entities extracted from a KBs - and then annotated with the author's sentiment about each topic. The corresponding pattern types are shown below.

ShortText: [sid, {word}, user, {tag}, position[latitude, longitude},time]
Topic: [tid, {word}]
EnrichedShortText: [eid, ShortText, {Topic, Sentiment}]

Short texts are produced, enriched and analyzed on the fly. They are managed as continuous flows of textual information, i.e., text streams. The pattern types for streams of texts (of both cases) are timestamped sequences:

TextStream: [stid, <ShortText,timestamp>] or
[stid, <EnrichedShortText,timestamp>]

Further analysis of these text streams may require the introduction of other pattern types: heatmap streams, which carry the aggregated information about geographic areas from where most of the short text come from, topic popularity streams, which carry the aggregated information about the most trendy topic under discussion, and contradictions, which are the time points when positive and negative sentiments have been simultaneously recorded with respect to a specific topic, or the time points when a sentiment shift has occurred (i.e., positive sentiments on a specific topic have turned into negative, or vice versa).

HeatmapStream: [stid, <Region, value, timestamp>]
TopicPopularityStreams: [stid, <Topic,value,timestamp>]
Contradiction Stream: [stid, <topic, time, timestamp>]

5.2 Models of Text Stream Cases

In casting problems P4, P5 and P6 as a mega-modeling process, we identified a pipeline for short text enrichment that is common to all three problems.

5.2.1 Short Text Enrichment Pipeline

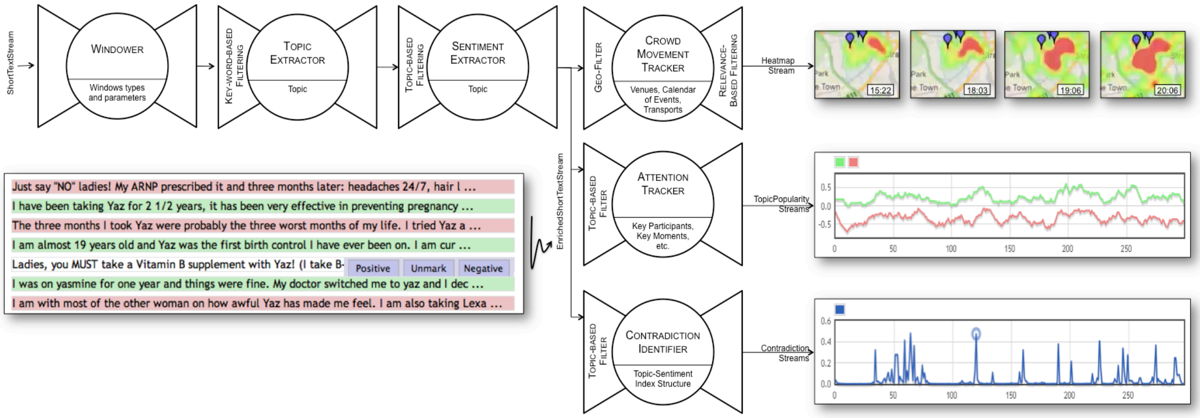


Figure 5: Problem P4, P5 and P6.

In the pipeline, the (infinite) incoming text stream is chunked in manageable blocks of short texts using windows, then each short text is parsed in order to detect the topics that are mentioned, and subsequently, the sentiment expressed in the short text for each one of these topics is extracted. Three new Mega-Modules are introduced:

- **Windowing** (W) – transforms a portion of a (by definition infinite) stream in a (finite) window – a block of processable information. Several types of windows exist, the most frequently used are: physical windows, which can hold a fixed number of data items, logical windows, which contain all the data elements received in a given time period, tumbling windows, whose content does not overlap, and sliding windows whose content overlaps. We use physical tumbling windows.
- **Topic Extractor** (TE) – extracts the topic, or topics that are mentioned in a short text. Evidently, when the short text is large enough, more than one topic may be mentioned. The topic extraction Mega-Module may look in the short text for a pre-defined set of topics (e.g., a list of topics that has been constructed based on domain knowledge, or a preceding processing step), or it may discover ad-hoc topics.
- **Sentiment Extractor** (SE) – extracts the sentiment corresponding to each of the topics mentioned in the working block. The sentiment value expresses the opinion represented as a discrete (e.g., negative/neutral/positive) or continuous (e.g., in the interval [-1,1]) value, of the subject towards a specific topic. In Fig. 5, sentiments are represented as real values between -1 and 1.

5.2.2 Text Stream Analytics Mega-Modules

The blocks of enriched short texts delivered by the reusable pipeline can then be further processed. In modeling our past text analytics experiences, three new Mega-Modules are introduced:

- **CROWD MOVEMENT TRACKER** (CMT) – tracks the movement of the crowds using geo-tagged user statements, and produces a time series of heat maps. The analysis can be limited to a given area (e.g., in [6] the London Olympic stadium and the train and metro stations around it, in [7] a tourist district of Seoul) and can be aware of the position (and shape) of the venues, the calendar of the events, and other background information. For instance, the series of heat maps in the upper-right corner of Fig. 5 show a crowd entering the Olympic stadium for the London 2012 Olympic Games opening ceremony.
- **ATTENTION TRACKER** (AT) – tracks the attention of the crowds using sentiment about a topic in the enriched short text and outputs a topic popularity stream.
- **CONTRADICTION IDENTIFIER** (CI) – organizes the information in the enriched short texts in an (incrementally maintainable) index structure that is used for efficiently managing the information on the sentiments expressed on various topics over time. This index structure is then used for identifying the time intervals and topics for which a sentiment-based contradiction occurs. The contradictions can be either very different sentiments expressed on the same topic or sentiment shifts (i.e., change of polarity of the sentiments expressed on some topic); they can be identified by examining the index structure at different time granularities 0. The final result is a stream of such contradictions, which are represented as peaks in the data stream shown in the bottom right of Fig. 5.

6. DATA STREAM USE CASE

Problem P7 is concerned with streams of data representing measures, which need to be first chunked into working sets using windows, then summarized, and then analyzed for identifying outliers.

6.1 Common Pattern Types

The datasets to be analyzed consist of streams of several measures produced by different sensors. The corresponding pattern types are shown below.

Measure: [sid, sensor,{oid, variable, value}]
MeasureStream:[stid,<Measure, timestamp>]

6.2 Model of P7 Case

Problem P7 is solved by a pipeline of Mega-Modules: the first one is the windowing Mega-Module, which chunks the stream using a sliding window, the second one computes a summary of the values in the window, and the third one performs outlier detection (see Fig. 6).

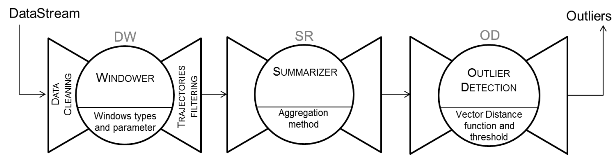


Figure 6: Problem P7

- **SUMMARIZER** (SR) – builds a concise summary of the multidimensional features associated to a set of given objects (e.g., based on histograms, or kernel density estimators).
- **OUTLIER DETECTOR** (OD) – given a population of objects described by multidimensional features and a notion of distance in the feature space, this module identifies an object as an outlier when the vector of the object differs significantly from the median vector of the population, or when the local neighborhood of the object is significantly less dense than its extended neighborhood.

After modeling P7, we realized that the outlier detection module performs a task that is very similar to the post-processing of the Trajectory Reconstruction (TR) module of P1. Indeed, once a trajectory is defined as an appropriate vector of features, a trajectory is outlier when its features significantly diverge from the features of the median. Then a remodeling of P1 takes place, by eliminating the post-processing from TR and chaining OD between TR and TM. This experience gives an indication of how we expect mega-modeling to evolve, with both top-down problem decompositions and bottom-up identification of reusable components across problems.

7. MEGA-MODULES IN THE CLOUD

In the following we show how to transform Mega-Modules into executable programs that use the Map/Reduce (M/R) paradigm, and could be executed on systems such as Hadoop, Dryad, or Stratosphere [10]. In particular, we use the Stratosphere platform, which provides two different programming models:

1. The PACT programming model, which supports flow programs based on second order functions (such as Map and Reduce) and User Defined Functions (UDFs). For example, the relational SELECTION operator is modeled by a MAP together with an UDF removing the input tuples that do not satisfy a filter predicate.
2. For higher level programming, Stratosphere provides a programming framework called SOPREMO which allows programmers to define custom packages, the respective operators and their instantiation.
3. The METEOR language allows programmer to write programs by specifying sequences of Jason-like statements using built-in operators or operators from one or more SOPREMO packages. Each METEOR statement may refer to output variables of previous statements or to input variables.

Once a METEOR program has been fully specified including data sources and data sinks, a compiler compiles it into a PACT program by:

- replacing library specific operators by their corresponding instantiations (part of the library);
- "chaining together" all partial PACT programs according to the variables of the Meteor program.

The result of this compilation is an executable PACT program which then may be optimized and compiled further for execution in a cluster based environment. Similar transformations are possible for target execution systems such as Hadoop (using the languages PIG or Hive) or Dryad (using the SCOPE language).

Fig. 7 shows a fragment of METEOR program which implements the Mega-Module TR (TRAJECTORY RECONSTRUCTION), which uses functions of previously defined SOPREMO packages. The program reads observations from the input, then cleans the observations that are considered as outlier or inconsistent, then builds histories as sequences of observations, then reconstructs trajectories using the StartStopFunction that is provided as local input to the TrajectoryReconstruction function, and finally writes trajectories to the output; note that outlier filtering is omitted from TR and associated with the Mega-Module OD. Mega-Modules OUTLIER DETECTION (OD), TRAJECTORY CLUSTERING (TM) and MOBILITY PROFILE (MP) can be encoded by similar METEOR programs.

```
MegaModule TR {  // Trajectory Reconstruction
using Trajectory ; //library used

input $ObservationData ;
input function $TStartStopDef  from Trajectory.StartStopFuntion;
output $TrajectoryData ;


$Observations = read from input $ObservationData;
$CleansedObservations = Cleanse $Observations
$Histories = BuildHistory $CleansedObservations
$Trajectories = TrajectoryReconstruction $Histories
                using function $TStartStopDef;
write $Trajectories to output $TrajctoryData ;

} //end MegaModule TR
```

Figure 7: METEOR Program Fragment for TR

Problem P1 was used in an application context where similar profiles of citizen are matched so as to propose car-pooling options to them. Such an extension can be supported as follows. First, observation data are extended with the identity of citizens. Then the high-level computation is a METEOR program that, for each user, invokes the function ComputeProfile that embodies the Mega-Modules TR, OD, TC and MP, and then calls a match function that produces pairs of candidate commuters. A simplified version of the corresponding METEOR program fragment is:

```
$CitizenData = read from input $ObservationData
$CitizenCommutes =
   Group CitizenData by CitizenID into
      {(CitizenID, Profile:
            ComputeProfile (CitizenData.Observations)};
$Candidates = Match $CitizenCommutes;
write $Candidates to output CandidateCommuters.
```

Finally, note that current M/R frameworks, including Stratosphere, do not support streams; however, windows allow gathering large collections of information during suitably long periods of time; thus, each window can be processes as a separate batch. In this way, it is possible to manage stream problems, such as P4-P7.

ACKNOWLEDGMENT. This work is supported by the Search Computing (SeCo) Project, sponsored by ERC, and by the FET-Open project DATASIM.

8. REFERENCES

[1] P. Ball: Why Society is a Complex Matter, Springer-Verlag,2012.

[2] S. Ceri, E. Della Valle, D. Pedreschi, R. Trasarti: Mega-Modeling for Big Data Analytics, *ER-2012*, LNCS 7352, Springer, pp. 1-15, October 2012.

[3] R. Trasarti, F. Giannotti, M. Nanni, F. Pinelli: Mining mobility user profiles for car pooling. In ACM International Conference on Knowledge Discovery and Data Mining, KDD 2011.

[4] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli: Trajectory Pattern Mining. In ACM International Conference on Knowledge Discovery and Data Mining, KDD 2007.

[5] S. Rinzivillo, S. Mainardi, F. Pezzoni, M. Coscia, F. Giannotti, D. Pedreschi: Discovering the Geographical Borders of Human Mobility. KI - Künstliche Intelligenz, 2012.

[6] M. Balduini and E. Della Valle: Tracking Movements and Attention of Crowds in Real Time Analysing Social Streams: The case of the Open Ceremony of London 2012. Semantic Web Challenge, 2012.

[7] M. Balduini, I. Celino, D. Dell'Aglio, E. Della Valle, Yi Huang, T. K. Lee, S. H. Kim, V. Tresp: BOTTARI: An augmented reality mobile application to deliver personalized and location-based recommendations by continuous analysis of social media streams. J. Web Sem. 16: 33-41, 2012.

[8] M. Tsytsarau, T. Palpanas, K. Denecke: Scalable Discovery of Contradictions on the Web. International World Wide Web Conference (WWW), Raleigh, NC, USA, April 2010.

[9] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, D. Gunopulos: Online Outlier Detection in Sensor Data Using Non-Parametric Models. Proc. VLDB, Seoul, Korea, September 2006.

[10] A. Heise, A. Rheinlander, M. Leich, U. Leser, F. Naumann, Meteor/Sopremo: An Extensible Query Language and Operator Model, Workshop on End-to-End Management of Big Data, Istanbul, Turkey (2012).

[11] Tomasz Imielinski, Heikki Mannila: A Database Perspective on Knowledge Discovery. Communication of the ACM 39 (11): 58-64, 1996.

[12] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, C. Renso, S. Rinzivillo and R. Trasarti: Unveiling the Complexity of Human

Mobility by Querying and Mining Massive Trajectory Data. *The VLDB Journal* 20(5): 695-719, 2011.
[13] G. Wiederhold, P. Wegner, S. Ceri: Towards Mega-Programming, ACM Communications, 35:11, 1992.
[14] J. Bezivin, F. Journault, P. Valduriez: On the need for Megamodels, OOPSLA2004/GPCE Workshop.
[15] J-M. Favre, T. Nguyen: Towards a Megamodel to Model Software Evolution Through Transformations. Electr. Notes Theor. Comput. Sci. 127(3), 59-74, 2005.

Web Table Taxonomy and Formalization

Larissa R. Lautert
Universidade Federal de Santa Catarina
Florianópolis, SC, Brazil
llautert@inf.ufsc.br

Marcelo M. Scheidt
Universidade Federal de Santa Catarina
Florianópolis, SC, Brazil
marcelo.scheidt@inf.ufsc.br

Carina F. Dorneles
Universidade Federal de Santa Catarina
Florianópolis, SC, Brazil
dorneles@inf.ufsc.br

ABSTRACT

The Web is the largest repository of data available, with over 150 million high-quality tables. Several works have combined efforts to allow queries on these tables, but there are still challenges, like the various different types of structures found on the Web. In this paper, we propose a taxonomy for the tabular structures and formalize the ones used with relational data and show, through an experimental evaluation, that WTCLASSIFIER, our supervised framework, classifies Web tables with high accuracy. Additionally, we use WTCLASSIFIER to categorize more than 300 thousand Web tables into our taxonomy and found that 82.25% are not formatted similarly to relational structure.

1. INTRODUCTION

According to Cafarella et al. [2], there are over 150 million HTML tables on the Web with high-quality relational data. The main incentive for working with this information is enabling analysis and integration of data on the Web, since it is the larger corpus of table ever seen. Several works have combined their efforts to properly identify [15, 16, 8], extract [9, 3, 12] and label [7, 14] this data, so it can be used as basis for structured queries. However, most of them have formatting different from relational databases. Hence, they should be individually analyzed to understand their structures.

Fig. 1 shows a Web table with high-quality relational data found in Wikipedia. Its structure addresses some challenges faced in table understanding. First of all, attribute names are disposed vertically, and not in a row, as in relational tables. Besides, there are two tables nested and attributes *Developed by* and *Language(s)* are multivalued. Data interpretation within these characteristics is easily understood by humans, but not by machines.

General information	
Title	Game of Thrones
Developed by	David Benioff D. B. Weiss
Language(s)	English, Dothraki
Broadcast	
Original channel	HBO
Picture format	1080i (HDTV)
Audio format	Dolby Digital 5.1

Figure 1: Web table with heterogeneous formatting.

In face of the many structure types found in tables on the Web, it is necessary to catalog and interpret most used categories. Thus, we can simplify understanding and processing of Web structured data. Crestan et al. [6] have already proposed a taxonomy with nine heterogeneous structures, but we argue that some important types were ignored.

In this paper, we propose Relational Knowledge Web table categories and present WTCLASSIFIER, an Artificial Neural Network-Based Classifier, which learns by analyzing features from each category. In summary, this paper makes the following contributions: (i) definition of a taxonomy for heterogeneous Web tables; (ii) formalization of its Relational Knowledge categories; and (iii) a framework for classification of heterogeneous Web tables.

The remainder of this paper is organized as follows. In Section 2, we propose a Web table taxonomy and formalize Relational Knowledge categories. Later, in Section 3, our classification framework is described. Its experimental evaluation is shown in Section 4, along with a classification of 342,795 tables collected from the Web. Section 5 discusses related work and compares our framework with a similar one. Finally, Section 6 presents conclusions e future work.

2. WEB TABLES TAXONOMY AND FORMALIZATION

The Relational Model [5] considers *relations* as data structures. Given n sets of domains $S_1, \cdots, S_n$, R is a relation on these n sets if it is a set of n-tuples each of which has its first element from S_1, its second element from S_2, and so on [5]. In view of this definition, we propose the subsequent definitions to formalize Web tables, having the relational model definition as base. As these Web tables have relational purpose, i.e., are composed of relational data, they should also be treated as relations.

The early part of this section brings general concepts of Web tables. They are important to properly define each Web table category later, in Subsection 2.2. Web tables used without relational purposes are briefly described in Subsection 2.3.

2.1 General Concepts

Structure presented on Table 1, with x rows and y columns, will be used to illustrate some concepts.

Table 1: Structure of a Web table.

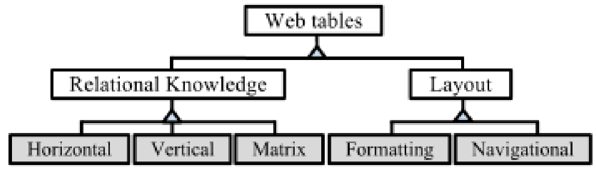
v_{11}	v_{12}	$\cdots$	v_{1y}
v_{21}	v_{22}	$\cdots$	v_{2y}
$\vdots$	$\vdots$		$\vdots$
v_{x1}	v_{x2}	$\cdots$	v_{xy}

Definition 2.1 (Web Table)**.** *We call* WEB TABLES WT *tabular structures found in Web pages, composed of an ordered set of x rows and y columns.*

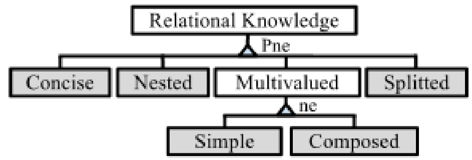
Definition 2.2 (Cell, Label, Data *and* Multivalued Data)**.** *Let* WT *be a* WEB TABLE *with x rows and y columns, represented on Table 1. Each intersection between a row and a column determines a* CELL c_{ij}*, which has a value v_{ij}, where $1 \leqslant i \leqslant x$ and $1 \leqslant j \leqslant y$. Values v_{ij} come from set $L = \{l_1, ..., l_y\}$, composed of* LABELS*, or from set $D = \{d_1, ..., d_{(x-1) \cdot y}\}$, composed of* DATA*. The elements domain in L is string, while the elements domain in D might be strings,* WEB TABLES*, null values or a set of atomic values. A value v_{ij} is said to be* MULTIVALUED DATA *iff it is composed of a set of* DATA *values.*

Comparing with Relational Model [5], LABELS are equivalent to domain names, while DATA values corresponds to elements of these domains. According to notation presented in [1], LABELS correspond to *attributes*. The major difference between Relational Tables and WEB TABLES is that the first one has a unique structure, in which the first row corresponds to attribute names and the others, to tuples. Fig. 3a shows an example of WEB TABLE, where LABELS (*Title* and *Year*), located in the first row, correspond to attribute names for DATA below.

In order to better understand some WEB TABLES characteristics, it is important to introduce the concept of SEQUENTIAL CELLS. It will be used in the definition of MERGED CELLS, found in some categories presented later.



(a) Main classification.



(b) Secondary classification.

Figure 2: Classification of Web table types.

Definition 2.3 (Sequential Cells)**.** *Let c_{ab} and c_{de} be two* CELLS *of a* WEB TABLE WT*, where a and d are indices for the rows and b and e, for the columns.* CELLS *c_{ab} and c_{de} are said to be* SEQUENTIAL CELLS *iff only one of the following conditions is true: $d = a + 1$ or $e = b + 1$.*

Let m, n and p be three arbitrary CELLS. If m is sequential to n and n is sequential to p, then m, n and p compose a set of SEQUENTIAL CELLS. For instance, in Fig. 5a, observe CELLS in the first ROW. The one containing value *PLANT* is sequential to CELL with *COLOR*, and this second is sequential to CELL with *HEIGHT*. Together, all the three compose a set of SEQUENTIAL CELLS.

Definition 2.4 (Merged Cell)**.** *Let* MC *be a set of* SEQUENTIAL CELLS *of a* WEB TABLE WT*.* MC *is said to be* MERGED CELL *iff all their elements are associated to the same value v, with $v \in L$ or $v \in D$.*

An example can be seen in Fig. 5a, where values v_{21}, v_{22} and v_{23} are associated to *SHRUBS*. As they have the same value, they are presented in one condensed CELL. It is important to note that only SEQUENTIAL CELLS can form a MERGED CELL.

2.2 Relational Knowledge Web Tables Categorization

Considering all these definitions presented, we will now introduce and categorize WEB TABLE structures, whose hierarchical classification is presented in Fig. 2. Note that, as we are interested on tables with relational purpose, only Relational Knowledge ones, which can generate data in Relational Model, are formalized. The others are informally described in Subsection 2.3.

Definition 2.5 (Horizontal Web Table)**.** *A* WEB TABLE WT *is said to be* HORIZONTAL *iff values v_{ij}*

($b \leqslant i \leqslant x$; $j = 1$) come from the domain S_1, values v_{ij} ($b \leqslant i \leqslant x$; $j = 2$), from domain S_2, and so on. b corresponds to index of the first row where $\exists\ v_{ij} \in D$ ($\forall\ i$; $\forall\ j$). If $\exists\ v_{ij} \in L$ ($i = 1$; $\forall\ j$), b assumes value 2 and $v_{ij} \in D$ ($b \leqslant i \leqslant x$; $\forall\ j$). Otherwise, $b = 1$ and values $v_{ij} \in D$ ($\forall\ i$; $\forall\ j$).

In Codd's definition [5] for the Relational Model, LABELS correspond to domain names on which the WEB TABLE WT is defined. The LABELS set is located on the first row, and the values of the remaining rows are DATA, likewise in a relational table. Fig. 3a exemplifies this type of structure.

Definition 2.6 (Vertical Web Table)**.** *A* WEB TABLE WT *is said to be* VERTICAL *iff values v_{ij} ($i = 1$; $b \leqslant j \leqslant y$) come from the domain S_1, values v_{ij} ($i = 2$; $b \leqslant i \leqslant y$), from domain S_2, and so on. b corresponds to index of the first column where $\exists\ v_{ij} \in D$ ($\forall\ i$; $\forall\ j$). If $\exists\ v_{ij} \in L$ ($\forall\ i$; $j = 1$), b assumes value 2 and $v_{ij} \in D$ ($\forall\ i$; $b \leqslant j \leqslant y$). Otherwise, $b = 1$ and values $v_{ij} \in D$ ($\forall\ i$; $\forall\ j$).*

In other words, WT is said to be VERTICAL if its tuples are disposed vertically. LABELS set, when present, is located in first column. In the structure shown on Table 1, LABEL represented by v_{11} would be associated to DATA located on the first row, i.e., from v_{12} until v_{1n}. This structure was informally defined as *Horizontal Listing* in [6] and as *Vertical Table* in [14]. Fig. 3b shows an example of this WEB TABLE, where LABELS (*Born*, *Residence*, *Nationality*, etc.) are located in the first column.

Definition 2.7 (Matrix Web Table)**.** *Let S_1, S_2 and S_3 be three different domains. A* WEB TABLE WT *is said to be* MATRIX *iff values $v_{ij} \in S_1$ ($i = 1$; $2 \leqslant j \leqslant y$), $v_{ij} \in S_2$ ($2 \leqslant i \leqslant x$; $j = 1$) and $v_{ij} \in S_3$ ($2 \leqslant i \leqslant x$; $2 \leqslant j \leqslant y$). Let V be a set with all values $v_{ij} \in$ WT. $v_{11} \in L$ and $(V - v_{11}) \in D$. Each value v_{ij} ($2 \leqslant i \leqslant x$; $2 \leqslant j \leqslant y$) belongs to the same tuple as v_{mj} ($m = 1$) and v_{in} ($n = 1$).*

In other words, each value that are not in the first row/column is associated to a value in row header and to another in column header. Fig. 4 shows statistics for car accident, where domain S_1 is *decades*, S_2 is *causes* and S_3 is the *number of accidents*. A human observer can easily note that CELL c_{22} content, the number 26, is not an instance of row header (*1980s*) neither column header (*Pilot error*), as they are no LABELS. It corresponds to the *Number of accidents* that happened on *1980s* by *Pilot error*. Value in CELL c_{11} is the only LABEL in this WEB TABLE, and in this case, is associated to the first column DATA values. There are no LABELS for *Decades* and *Number of accidents* values.

Year	Title
1925	The Freshman
1931	Maker of Men
1932	Horse Feathers

(a) Horizontal Web table.

Robert De Niro	
Born	August 17, 1943 New York, NY
Nationality	American
Occupation	Actor and director

(b) Vertical Web table.

Figure 3: Examples of Web tables

Cause	1980s	1990s	2000s
Pilot Error	26	27	30
Weather	14	10	8
Mechanical Failure	20	18	24

Figure 4: Example of Matrix Web table.

Definition 2.8 (Concise Web Table)**.** *Let* MC *be any* MERGED CELL. *A* WEB TABLE WT *is said to be* CONCISE *iff* WT $\supset$ MC.

In a CONCISE WEB TABLE, there is occurrence of MERGED CELLS to avoid repetition of values, so then it becomes more compact, i.e., concise. Challenges on interpreting this structure where already mentioned in recent work [14], where MERGED CELL is considered a sub-header for the rows below it. We see the problem in a most general way. In the case illustrated in Fig. 5a, MERGED CELL values represent common DATA for rows below and can be seen as sub-headers. It could be also represented as a new attribute posed in another column, with value *shrubs* for plants *azalea* and *buddleia*; and *cultivated annuals* for plant *alyssum*.

However, in the situation of Fig. 5b, CELLS disposed vertically were merged and do not act as sub-headers. In order not to repeat equal year values, which are the same for the films *Death at a Funeral*, *I Love You Too* and *Pete Smalls is Dead* (*2010*), original CELLS were merged into one.

Definition 2.9 (Nested Web Table)**.** *A* WEB TABLE WT *is said to be* NESTED *iff $\exists\ v_{ij} \in$* WT *that is another* WEB TABLE.

The WEB TABLE presented in Fig. 1 is classified as NESTED. It can be observed that there are two WEB TABLES nested in one, separated for MERGED CELLS containing each WEB TABLE title.

PLANT	COLOR	HEIGHT
SHRUBS		
Azalea	variable	shrub
Buddleia	blue, pink, white	shrub
CULTIVATED ANNUALS		
Alyssum	violet, white	4 inches

(a)

Year	Title
2010	Death at a Funeral
	I Love You Too
	Pete Smalls Is Dead
2011	A Little Bit of Heaven

(b)

Figure 5: Examples of Concise Web tables

Definition 2.10 (Splitted Web Table)**.** *A* WEB TABLE WT *is said to be* SPLITTED *iff its* LABELS *present sequential ordered repetitions in the row/column header. Let s be the number of these repetitions. Hence, each* LABEL *is repeated every* $\frac{z}{s+1}$ CELL(S)*, where $z = y$ if the* WT *is* HORIZONTAL*; and $z = x$ if it is* VERTICAL*.*

Comparing with Relational Model, we can say that each DATA row of a SPLITTED WEB TABLE is composed of $s+1$ tuples. This fact can be observed in Fig. 6, where the SPLITTED WEB TABLE has $s = 1$, i.e., it was horizontally splitted once. Thus, the set of LABELS consisting of *Rank*, *City name* and *Pop.* appears repeated once. In analogy with the Relational Model, it can be said that rows from 2 to 6 are composed of two tuples.

Rank	City name	Pop.	Rank	City name	Pop.
1	São Paulo	11,316,149	6	Belo Horizonte	2,385,639
2	Rio de Janeiro	6,355,949	7	Manaus	1,832,423
3	Salvador	3,093,605	8	Curitiba	1,764,540
4	Brasília	2,609,997	9	Recife	1,536,934
5	Fortaleza	2,476,589	10	Porto Alegre	1,413,094

Figure 6: Example of Splitted Web table.

Definition 2.11 (Multivalued Web Table)**.** *Let v_{ij} be any value of* WEB TABLE WT*.* WT *is said to be* MULTIVALUED *iff $\exists\ v_{ij} \in$* WT *that is a* MULTIVALUED DATA*, composed of a set of k* DATA *values $\{m_1, ..., m_k\}$, which come from the domains $\{S_1, ..., S_k\}$, respectively.*

In this case, some DATA values are sets of other DATA values, as we will see in subsequent definitions.

Definition 2.12 (Simple Multivalued Web Table)**.** *Let v_{ij} be a* MULTIVALUED DATA *of* WEB TABLE WT *with k* DATA *values.* WT *is said to be* SIMPLE MULTIVALUED *iff $S_1 = ... = S_k$, i.e., all k* DATA *values of v_{ij} come from the same domain.*

The WEB TABLE in Fig. 1 has MULTIVALUED DATA in two situations. The first one is in DATA value for *Developed by*, which is composed of two names (*David Benioff* and *D. B. Weiss*). The other case occurs in *Language(s)*, where the DATA value is a set of two strings (*English* and *Dothraki*).

Definition 2.13 (Composed Multivalued Web Table)**.** *Let v_{ij} be a* MULTIVALUED DATA *of* WEB TABLE WT*.* WT *is said to be* COMPOSED MULTIVALUED *iff $S_1 \neq ... \neq S_k$, i.e., all k* DATA *values of v_{ij} are from different domains.*

The WEB TABLE in Fig. 3b has COMPOSED MULTIVALUED DATA in *Born* value, which consists of information about date and city of birth.

2.3 Layout Web Tables

Most of the HTML tables found on the Web are used only for layout purpose. According to [6], they can be divided in FORMATTING and NAVIGATIONAL. The first one is used in order to organize elements (text, images, videos, tables, etc.) in the page, while the second category disposes items of menus containing hyperlinks for navigating purpose.

3. WTCLASSIFIER, A NEURAL NETWORK-BASED CLASSIFIER

In order to automate the Web tables classification process, we have developed WTCLASSIFIER, a supervised Neural Network classifier using Neuroph[1]. Our framework learns from analyzing each category patterns, represented by a list of layout, HTML and lexical features, likewise approach used in [6]. Along with 20 features they described, we added 5 new ones: position of inner HTML tables and ratio of cells containing unordered lists, ordered lists, commas and brackets. The first feature helps on identification of Formatting Web table, while the other four often characterize MULTIVALUED DATA.

For the training phase, we have provided a list of 25 features extracted from 4,000 Web tables and their categories (golden collection). As output, we have 1 neuron per category, with value ranging from 0 to 1. We have used Multilayer Perceptron Network, with 1 hidden layer and resilient propagation. Classification process steps are described below.

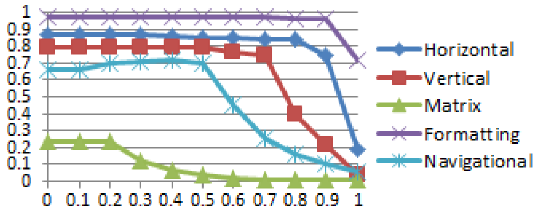
Main Classification separates Web tables in five categories: HORIZONTAL, VERTICAL, MATRIX, FORMATTING and NAVIGATIONAL. These classes are mutually exclusive, i.e., a Web table cannot be in more than one of these classes at a time.

Secondary Classification categorizes HORIZONTAL, VERTICAL and MATRIX in CONCISE, NESTED, SPLITTED, SIMPLE MULTIVALUED and/or COMPOSED MULTIVALUED. As input, we consider 25 features mentioned before plus the category obtained in Main Classification. Output categories are not mutually exclusive.

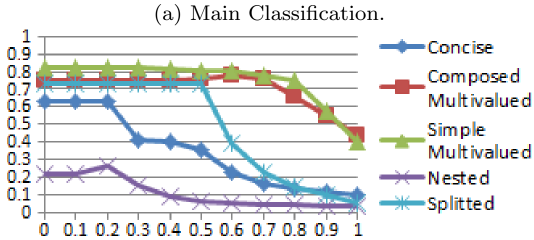
4. EXPERIMENTS

We have developed a crawler focused on extracting Web tables. As seeds, we have used Wikipedia, e-commerce, news and university sites, visiting a total of 174,927 pages, in which 104,261 contained Web tables. From these pages, we have extracted 631,382 HTML tables. Discarding repeated ones, 342,795 were left.

[1]Neural Network Framework (neuroph.sourceforge.net)

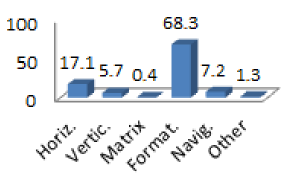


(a) Main Classification.

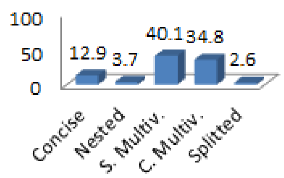


(b) Secondary Classification.

Figure 7: Recall vs precision graphs.



(a) Main Classific.



(b) Secondary Classific.

Figure 8: Distribution of Web tables.

4.1 Web Tables Distribution

First, we have ran WTCLASSIFIER to categorize the set with 342,795 Web tables in Main Classification. Latter, the ones first categorized as *Relational Knowledge* (75,233), went through Secondary Classification. Observing Fig. 8a, we note that the most used structure to represent relational data is HORIZONTAL (17.1%), followed by VERTICAL (5.7%). A total of 1.3% of our set was not classified in any of our categories. Observing Secondary Classification distribution, it can be seen that MULTIVALUED is the most often type, occuring in 74.9% of Relational Knowledge Web tables (40.1% INCLUSIVE plus 34.8% COMPOSED).

In order to verify how many Relational Knowledge Web tables were formatted similarly to relational structure, we have counted the number of HORIZONTAL occurences that do not fit in any category of Secondary Classification. We found that only 17.75% present this homogeneous structure.

4.2 WTClassifier Quality Evaluation

For each classification phase, there were generated 10 non-overlapping sets from our golden collection (90% of examples used for training and 10% for testing purpose). Using these values, through a 10-fold cross-validation process, we obtained precision and recall values represented in Fig. 7. MATRIX, the category with the lowest representation (Fig. 8a), is the one with worst precision values. This happens because there are too few cases of matrix in the training set (14). In Secondary Classification, it can be seen that WTCLASSIFIER is presenting difficulties on distinguishing categories where MERGED CELLS are more often (CONCISE and NESTED), and therefore, they present lower precision values. On the other hand, SIMPLE and COMPOSED MULTIVALUED have the best results due to their representatively in training set, plus full cover of their main patterns in features list described in Section 3.

Comparing F-measure values in Table 2 for overlapping categories, we can see that WTCLASSIFIER outperforms TabEx [6] in HORIZONTAL, VERTICAL, FORMATTING and NAVIGATIONAL categories.

	TabEx	WTClassifier
Horizontal	0.72	0.83
Vertical	0.24	0.71
Matrix	0.30	0.22
Formatting	0.86	0.95
Navigational	0.45	0.60

Table 2: F-measure for TabEx and WTClassifier.

5. RELATED WORK

Since Cafarella [2] attested the wealth of knowledge present in HTML tables, some studies have focused on them [7, 10, 14, 3, 13, 11]. The main issue with this data structure comes from the fact of them being made for human consumption, and therefore, machines have difficulty on interpreting some kinds of formatting. Heterogeneous structures for HTML tables, like vertical tables and lists presented in two dimensions were already noticed and dismissed in previous work [4, 14]. While most of them only worried about detecting HTML tables similar to relational tables [2, 4, 16], Yoshida et al. [17] came up with a method to integrate tables of the Web and proposed nine categories. However, these categories do not reflect the most common cases found on the Web today. A more complete taxonomy was proposed by Crestan et al. [6], where categories range from tables used only for layout purpose to structures similar to relational tables.

The most recent taxonomy proposed [6] has 7 categories of Relational Knowledge Tables. Even though this classification is useful, it presents some conceptual issues. One should note that, planning algorithms for bringing heterogeneous Web tables to a unique structure, *Attribute/Value* (Web table with only one entity) and *Enumeration* (Web table with only one attribute) categories should be simply classified in *Vertical* or *Horizontal* categories; while *Form* type should not be classified as *Relational Knowledge Table*, as they have no high-quality relational data. Furthermore, they consider only inner HTML tables in classification process, assuming all ones are FORMATTING; while we add this characteristic as input on Neural Network and let it decide how important this feature is for each category. Another point is that since we are dealing with tables, which are, formally, *relations* [5], it would be appropriate to formally define them according to the Set Theory.

Comparing results in Fig. 8a with the one reported by Crestan et. al [6], we note that *Layout* categories (FORMATTING and NAVIGATIONAL) do not present similar values for Web table distribution. We suppose this is due to different seeds used for crawling. As we are most interested in analyzing Relational Knowledge structures, many of our Web tables come from Wikipedia, known for its high-quality data. Thereby, we obtained higher occurrences of HORIZONTAL and VERTICAL WEB TABLES. Moreover, tables classified as *Enumeration* and *Attribute/value* in *TabEx* are classified as *Horizontal* or *Vertical* in WTCLASSIFIER.

6. CONCLUSIONS AND FUTURE WORK

We have presented formalizations for Relational Knowledge Web tables, essential for defining algorithms to deal with them. Besides, we propose a taxonomy for Web tables categories, with five not mentioned in previous work. Comparing F-measure values of WTCLASSIFIER with reported results of TabEx [6], we note that our framework outperforms on identifying four, in a total of five categories in common. As future work, we highlight the importance of defining algorithms for bringing all categories to a unique structure. Thus, applications which deal with Web table data would not worry about their heterogeneous characteristics. Another issue consists of Web tables which subject is in surrounding text, and not within its structure.

7. REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Add.-Wesley, 1995.

[2] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. Webtables: exploring the power of tables on the web. *Proc. VLDB Endow.*, 1(1):538–549, Aug. 2008.

[3] M. J. Cafarella, J. Madhavan, and A. Halevy. Web-scale extraction of structured data. *SIGMOD Records*, 37(4):55–61, Mar. 2009.

[4] M. J. Cafarella and E. Wu. Uncovering the relational web. In *WebDB*, 2008.

[5] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[6] E. Crestan and P. Pantel. Web-scale table census and classification. In *WSDM '11*, pages 545–554, New York, NY, USA, 2011. ACM.

[7] A. S. da Silva, D. Barbosa, J. M. B. Cavalcanti, and M. A. S. Sevalho. Labeling data extracted from the web. In *ODBASE*, pages 1099–1116. Springer, 2007.

[8] D. W. Embley, C. Tao, and S. W. Liddle. Automating the extraction of data from html tables with unknown structure. *Data Knowl. Eng.*, 54(1):3–28, July 2005.

[9] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain-independent information extraction from web tables. In *WWW*, pages 71–80, New York, NY, USA, 2007. ACM.

[10] G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and searching web tables using entities, types and relationships. *Proc. VLDB Endow.*, 3(1-2):1338–1347, Sept. 2010.

[11] S. Mergen, J. Freire, and C. A. Heuser. Querying structured information sources on the web. In *iiWAS*, pages 470–476, New York, NY, USA, 2008. ACM.

[12] G. Nagy, S. C. Seth, D. Jin, D. W. Embley, S. Machado, and M. S. Krishnamoorthy. Data extraction from web tables: The devil is in the details. In *ICDAR*, pages 242–246, 2011.

[13] S. L. Sardi Mergen, J. Freire, and C. A. Heuser. Indexing relations on the web. In *EDBT*, pages 430–440, USA, 2010. ACM.

[14] P. Venetis, A. Halevy, J. Madhavan, M. Paşca, W. Shen, F. Wu, G. Miao, and C. Wu. Recovering semantics of tables on the web. *Proc. VLDB*, 4(9):528–538, June 2011.

[15] Y. Wang and J. Hu. Detecting tables in html documents. In *DAS*, pages 249–260, London, UK, UK, 2002. Springer-Verlag.

[16] Y. Wang and J. Hu. A machine learning based approach for table detection on the web. In *WWW*, pages 242–250, New York, NY, USA, 2002. ACM.

[17] M. Yoshida and K. Torisawa. A method to integrate tables of the world wide web. In *WDA*, pages 31–34, 2001.

XQuery 3.0 is Nearing Completion

Andrew Eisenberg
IBM, Westford, MA 01886
andrew.eisenberg.ma@verizon.net

Introduction

XQuery 3.0 was published in January as a W3C Candidate Recommendation [1]. It adds a number of new features to XQuery 1.0, which was published as a W3C Recommendation in Jan. 2007 and revised in Dec. 2010 [2]. In this article, I will describe some of these new features.

The XML Query Working Group is responsible for these specifications:

- XQuery 3.0: An XML Query Language
- XML Syntax for XQuery 3.0

The XML Query Working Group and the XSL Working Group are jointly responsible for these specifications:

- XPath 3.0
- XQuery and XPath Data Model (XDM) 3.0
- XQuery and XPath Functions and Operators 3.0
- XSL and XQuery Serialization 3.0

XQuery 3.0 was initially called XQuery 1.1, but the Working Groups later decided to align the names of these new versions of XPath and XQuery. The relationship between the two languages is unchanged; XPath 3.0 is largely a subset of XQuery 3.0.

The Working Groups decided not to publish a new version of XQuery 1.0 and XPath 2.0 Formal Semantics. The rules for static type inferencing are now implementation-defined.

Publication as a Candidate Recommendation means that XQuery 3.0 is largely complete and has been widely reviewed. The Working Groups are gathering implementation experience before requesting that the specification advance further.

New Features in XQuery 3.0

Immediately after XQuery 1.0 was completed, the Working Groups published a set of requirements for XQuery 1.1. Positional grouping, windowing, error processing and recovery, and higher order functions were among these requirements. Usage scenarios were created for some of these features in the XQuery 3.0 Use Cases document [3]. As our work progressed, additional features were requested. Among these were the relaxed FLWOR expression, the simple map operator, and the string concatenation operator.

Before discussing these new features, I need to introduce a bit of terminology. The FLWOR expression is made up of `for`, `let`, `where`, `order by`, and `return` causes. All but the `return` clause can be thought of as generating ordered streams of tuples. Each tuple contains one or more bindings of variables to values. Consider this FLWOR fragment:

```
for $name in doc('employees.xml')
             //employee/data(name)
let $len := string-length($name)
let $flet := substring($name, 1, 1)
```

This fragment would produce the following tuple stream:

```
($name := "Jones",  $len := 5, $flet := "J")
($name := "Barnes", $len := 6, $flet := "B")
```

Group By

The `group by` clause will be familiar to anyone that has used SQL. Tuples are grouped, and these groups can be acted upon by subsequent clauses.

This new clause in the FLWOR expression allows a user to specify one or more grouping variables.

```
let $emp := doc('employees.xml')
return
   for $e in $emp//employee
   let $state := $e/address/state
   let $name := $e/name
   group by $state
   return <state name="{$state}">
             {$name}
          </state>
→
<state name='NY'>
   <name>Jones</name>
   <name>Barnes</name>
</state>
.
.
.
```

In this example, `$state` is a grouping variable, while `$e` and `$name` are non-grouping variables.

This clause produces a post-grouping tuple from one or more pre-grouping tuples – tuples provided by previous FLWOR clauses. The variable names in the post-grouping tuples will be the same as those in the pre-grouping tuples, but they will be bound to new values.

Each grouping variable generates a grouping key, which is the atomized value of the grouping variable. The atomized value of an atomic value is the value itself, while the atomized value of a node is the typed value of that node. A type error is raised if any grouping key is a sequence of more than one item.

Each pre-grouping tuple is assigned to a group, based on the value of its grouping keys. The `fn:deep-equal` function is used to compare key values. This means that an empty sequence is equivalent to another empty sequence, a `NaN` is equivalent to another `NaN`, two `untypedAtomic` values are compared as strings, and that two values for which `eq` is not defined, such as integer and date, are not equivalent.

The tuples produced by the `group by` clause have bindings for all of the grouping and non-grouping variables. A tuple is produced for each group of input tuples. For this reason, the number of post-grouping tuples will be equal to or less than the number of pre-grouping tuples.

Each post-grouping tuple has bindings for each grouping variable to one of its corresponding grouping keys. I say "one of" because the grouping keys for a grouping variable must all be equivalent according to `fn:deep-equal`, but they may not be identical. Some collations consider `"Andrew"` and `"andrew"` to be equal, but they are certainly not identical.

The post-grouping tuple has bindings for each non-grouping variable to the concatenation of all of the values of the non-grouping variable for the tuples in the partition. When ordering mode is set to ordered, the concatenation preserves the order of the values in the pre-grouping tuple stream.

In the example above, the first post-grouping tuple might be:

```
($state := "NY",
 $e := <employee><name>Jones</name> ...,
       <employee><name>Barnes</name> ...,
 $name := <name>Jones</name>,
          <name>Barnes</name>
)
```

Grouping variables can be bound within the `group by` clause. Also, a `collation` clause can be specified for the comparison of string values:

```
group by $state := $e/address/state
            collation
            "us-english-case-insensitive",
         $city := $e/address/city
            collation
            "us-english-case-insensitive"
```

SQL's `HAVING` clause allows a user to filter out some of the groups created by the `GROUP BY` clause. In XQuery 3.0, this is accomplished with a `where` clause following the `group by` clause.

Count

The `count` clause has been added to the FLWOR expression to identify the position of tuples in the tuple stream.

```
for $x in (1, 2)
for $y in (10, 20)
count $c
return concat ($c, ": ", $x + $y)
```

→

```
"1: 11", "2: 21", "3: 12", "4: 22"
```

The count clause adds a binding of `$c` to each tuple in the tuple steam. `$c` is bound to the position of the tuple in the tuple stream.

Windowing

The `window` clause operates on a sequence and derives zero or more windows from that sequence. Each window is a sequence of consecutive items taken from the input sequence. Windows can contain different numbers of items. Each window can be identified by its start and end position in the input sequence.

Each `window` clause specifies either `tumbling` or `sliding`. A tumbling window clause generates windows that do not overlap one another. A sliding window clause allows the windows to overlap, but it does not allow two windows to start at the same position.

The `window` clause is similar to the `for` clause. It operates on an input sequence and binds its variable to successive windows, starting from the beginning of the input sequence. The `window` clause allows a user to bind additional variables and then use these variables to define the windows that are generated. These additional variables may be bound in the `start` clause and `end` clause:

- start/end item (`$v`)
- start/end item position (`at $v`)
- item previous to the start/end item (`previous $v`)
- item following the start/end item (`next $v`)

A user must pick distinct names for the variables defined within a single `window` clause.

Let's look at a simple example:

```
for sliding window $w in
  ("a","b","c","d","e","f","g","h","i","j")
start $s at $spos when $spos mod 2 eq 1
end $e at $epos when $epos eq $spos + 2
return <w start="{$s}" end="{$e}">{$w}</w>
```

→

```
<w start="a" end="c">a b c</w>
<w start="c" end="e">c d e</w>
<w start="e" end="g">e f g</w>
<w start="g" end="i">g h i</w>
<w start="i" end="j">i j</w>
```

For each window that is generated, `$s` is bound to the start item, `$spos` is bound to the position of the start item, `$e` is bound to the end item, and `$epos` is bound to the position of the end item. We start our windows at odd numbered positions and end them two positions later.

The final result element, starting at position 9, reflects a window with only two items. When no item satisfied the end condition, the last item in the input sequence was taken to be the end of this window. This last window can be skipped by starting the `end` clause with `only`. This prefix requires that each window have an end condition that is true.

Let's look at two, more realistic, examples. We'll start with a document that records trades, with each trade having a stock name, a price, and a timestamp. `trades.xml` contains the trades in time order, as follows:

```
<trades>
   <trade stock="ACO"
          price="200" time="12:00:00.1"/>
   <trade stock="ACO"
          price="202" time="12:00:00.2"/>
   <trade stock="BCO"
          price="200" time="12:00:00.3"/>
   <trade stock="CCO"
          price="300" time="12:00:00.4"/>
   <trade stock="CCO"
          price="299" time="12:00:00.5"/>
   <trade stock="CCO"
          price="290" time="12:00:00.6"/>
   <trade stock="ACO"
          price="200" time="12:00:00.7"/>
   <trade stock="ACO"
          price="205" time="12:00:03.1"/>
</trades>
```

The following query finds one second windows after the trade of a specific stock and then filters the windows, keeping those that contain more than one trade for the stock.

```
let $onesec := xs:dayTimeDuration('PT1S')
for sliding window $w
   in doc("trades.xml")//trade
start $s when $s/@stock eq "ACO"
end next $n
    when $n/@time > ($s/@time + $onesec)
let $occurrences
   := count ($w[@stock eq $s/@stock])
where $occurrences gt 1
return <run stock="{$s/@stock}"
            occurrences="{$occurrences}"
            time="{$s/@time}"
       />
```

→

```
<run stock="ACO"
     occurrences="3" time="12:00:00.1"/>
<run stock="ACO"
     occurrences="2" time="12:00:00.2"/>
```

The following query finds runs of trades on the same stock:

```
for tumbling window $w
   in doc("trades.xml")//trade
start $s when true ()
end next $n when $n/@stock ne $s/@stock
where count ($w) gt 1
return <run stock="{$s/@stock}"
            length="{count ($w)}"
            max="{max($w/@price)}"/>
```

→

```
<run stock="ACO" length="2" max="202"/>
<run stock="CCO" length="3" max="300"/>
<run stock="ACO" length="2" max="205"/>
```

Saying "`start $s when true()`" causes a window to start at the beginning of the input sequence and additional windows to start as soon as the previous window has ended. The end condition for the `window` clause is that the stock name of the next trade differs from the stock name of the initial trade. This means that every trade in the window will have the same stock name. We filter out those windows that have just a single trade, and then construct an element for each window that contains the stock name, the number of trades, and the maximum price.

A tumbling window can omit its `end` clause. This will end a window just before the start of the window that follows it. The example above can be rewritten as:

```
for tumbling window $w
   in doc("trades.xml")//trade
start $s previous $p
   when not ($p/@stock eq $s/@stock)
where count ($w) gt 1
return <run stock="{$s/@stock}"
            length="{count ($w)}"
            max="{max($w/@price)}"/>
```

Relaxed FLWOR Expression

The FLWOR expression in XQuery 1.0 consisted of one or more `for` clauses and `let` clauses, followed by an optional `where` clause, an optional `order by` clause, and a `return` clause.

As we considered the `group by` clause, the `window` clause, and the `count` clause, we saw that each clause operates on a tuple stream and generates a new tuple stream. The rigid order of the clauses could be relaxed.

The XQuery 3.0 FLWOR expression begins with a `for` clause, a `let` clause, or a `window` clause. Any number of `for`, `let`, `window`, `where`, `group by`, `order by`, and `count` clauses can occur between the initial clause and the final `return` clause, and they can occur in any order.

```
let $emp := doc("employees.xml")
for $e in $emp//employee
let $state := $e/address/state
where starts-with ($state, 'M')
group by $state
order by count ($e) descending
count $c
where $c le 5
return <state
          name="{$state}"
          position="{$c}"
          emps="{count ($e)}">
       </state>
```

This query finds employees with a state that starts with "M", groups them by state, orders the groups based on the number of employees they contain, numbers the groups, keeps only the top 5 groups, and then reports the position and number of employees for each state.

Try/Catch

In XQuery 1.0, an expression that raises a dynamic error causes the evaluation of the query that contains the expression to fail. The `castable` expression was provided to allow users to avoid errors when casting values, which is a common occurrence:

```
let $additionalCharge
       := if ($sale/weight
              castable as xs:decimal)
          then if ($sale/weight
                   cast as xs:decimal gt 10)
               then 1.50
               else 0
          else 0
```

If a value of weight is not castable to `xs:decimal`, `"light"` for example, then 0 is returned.

We've added the `try/catch` expression to XQuery 3.0. It is a more general expression than the `castable` expression. The example above could be rewritten as:

```
let $additionalCharge
       := try {
            if ($sale/weight
                cast as xs:decimal gt 10)
            then 1.50
            else 0
          }
          catch err:FORG0001 {0}
```

`err:FORG0001` is raised when an input value cannot be converted to a value in the value space of the target datatype. XQuery defines many errors and XQuery implementations may define additional errors.

If the `try` clause evaluates without error, then it provides the result of the `try/catch` expression.

The `catch` clause allows a "`|`" separated list of QNames to identify specific dynamic errors that should be caught. Wildcards such as `*`, `myco:*`, and `*:FORG0001` can be specified as well.

Multiple `catch` clauses may be specified. If an error occurred in the `try` clause, then the first `catch` clause that matches the dynamic error is evaluated and produces the result for the `try/catch` expression.

Within a `catch` clause, the following variables are bound and available to the user:

- `$err:code`
- `$err:description`
- `$err:value`
- `$err:module`
- `$err:line-number`
- `$err:column-number`
- `$err:additional`

If the error is not matched by any `catch` clause, then the `try/catch` expression raises the error.

The `try/catch` expression does not catch static errors. Static errors are detected during the analysis phase and prevent the query from executing.

```
try {
   substring ($description, 1, 7, "orange")
}
catch * {"oops"}
```

This query raises static error `err:XPST0017`, because no substring function with 3 parameters exists in the static context.

Higher-Order Functions

The XPath and XQuery data model has been extended, allowing function as a new primitive type. This allows functions to return functions and it allows functions to accept functions as arguments.

Function types can be written as:

```
function(*)
```

```
function(xs:string, xs:integer) as xs:string
```

The first of these identifies a function with any signature. The second identifies a function that accepts a string and an integer and returns a string.

An inline function expression allows a user to create an anonymous function item.

```
let $capitalize :=
       function ($s)
          {let $leading := substring($s,1,1)
           let $trailing := substring($s,2)
           return
              concat (upper-case($leading),
                      lower-case($trailing))
          }
return $capitalize ("association")
```

→

```
"Association"
```

This example binds a function to `$capitalize` and then returns an invocation of this function. Because a type was not specified for the `$s` parameter and the function return, these types default to `item()*`. The user could have been more precise and defined the inline function with:

```
function ($s as xs:string) as xs:string
{...}
```

A function returned by an inline function expression carries with it the static context and variable bindings that existed when it was created.

```
let $x := 7
let $f := function($i) {$x + $i}
return
   let $x := 12
   return $f(100)
```

→

```
107
```

A named function reference can be used to return functions by naming statically defined functions, including XQuery's built-in functions. Both the function name and the number of parameters must be specified, using *name#arity*.

The `fn:string-join` function, used in the examples below, accepts a sequence of strings and a string separator. It concatenates the strings in the sequence, adding the separator string between pair of strings.

```
let $f := fn:string-join#2
return $f(("a", "z"), " to ")
```

→

```
"a to z"
```

A function is partially applied when it is invoked with one or more argument placeholders, "?". This partial application of a function item returns a new function item, with as many parameters as there are argument placeholders.

```
let $dash-join := fn:string-join(?, "--")
let $ducks := ("huey", "duey", "louie")
return $dash-join($ducks)
```

→

```
"huey--duey--louie"
```

Functions can be defined that accept functions as their arguments and return functions as their results. A "top" function might accept a sequence, a ranking function and an integer n, returning the top n items in the sequence:

```
declare function local:top
   ($seq as xs:string*,
    $rank as
       function (xs:string) as xs:integer,
    $n as xs:integer)
{
   for $i in $seq
   order by $rank($i) descending
   count $c
   where $c le $n
   return $i
};

local:top (("red", "green", "blue"),
           string-length#1,
           2)
```

→

```
"green", "blue"
```

This returns the top 2 colors, ordered by the number of characters in each value.

XQuery provides several useful higher-order functions:

`fn:for-each ($seq, $f)`
: Applies the function item `$f` to every item from the sequence `$seq` in turn, returning the concatenation of the resulting sequences in order.

`fn:filter ($seq, $f)`
: Returns those items from the sequence `$seq` for which the supplied function `$f` returns true.

`fn:fold-left ($seq, $zero, $f)`
: Processes the supplied sequence from left to right, applying the supplied function repeatedly to each item in turn, together with an accumulated result value.

`fn:fold-right ($seq, $zero, $f)`
: Processes the supplied sequence from right to left, applying the supplied function repeatedly to each item in turn, together with an accumulated result value.

`fn:for-each-pair ($seq1, $seq2, $f)`

Applies the function item $f to successive pairs of items taken one from `$seq1` and one from `$seq2`, returning the concatenation of the resulting sequences in order.

A simple example using `fn:for-each` is:

```
let $ctof :=
       function($c) {9 * $c div 5 + 32}
return fn:for-each ((0, 100), $ctof)
```

→

```
32, 212
```

Annotations

Annotations allow properties to be associated with both variables and functions when they are being declared.

```
declare namespace myXQuery
   ="http://www.example.com/performance" ;

declare %private variable $stack := ... ;

declare %myXQuery:optimize("max")
   function local:creditScore ... ;
```

Annotations are QName, value pairs introduced with "`%`". XQuery defines some annotations in the `http://www.w3.org/2012/xquery` namespace.

`%private` does not specify a namespace prefix, and so is considered to be in the `http://www.w3.org/2012/xquery` namespace. Private variables and functions are not visible to the modules that import the library module that contains them.

Implementations are free to define annotations in their own namespaces. The behavior of these annotations is completely up to the implementation. One might guess that the user intends to spend extra time and resources optimizing the `local:creditScore` function. An implementation that doesn't recognize that annotation is free to ignore it.

Some Additional Features

XQuery 3.0 has many new features that are smaller and simpler than those shown above. In this section, I will describe some of them.

Simple Map Operator

The simple map operator is similar to the path expression. It consists of a series of expressions, separated by "`!`". Each operation E1 ! E2 is evaluated by first evaluating E1, using each item in E1 as the inner focus for evaluating E2, and then concatenating each evaluation of E2. Unlike the path expression, there are no restrictions on the type of items in either E1 or E2, duplicate elimination does not take place, and reordering does not take place.

Some examples of this operator are:

```
(1 to 5) ! (. * .)
```

→

```
1, 4, 9, 16, 25
```

```
let $emp := doc('employee.xml')//employee
return $emp
       !string-join((name, dept), ", ")
       !upper-case(.)
```

→

```
"JONES, A00", "BARNES, B01"
```

String Concatenation Expression

The string concatenation expression has been introduced as a short-hand for the `fn:concat` function. It uses "`||`" to identify the strings that will be concatenated.

```
$emp!(./name || " works for " || ./dept)
```

→

```
"Jones works for A00"
```

URI Qualified Name

An expanded QName can be specified by with a URI qualified name, which is made up of a namespace URI and a local name:

```
doc("employees.xml")
//Q{http://www.example.com/emp}jobTitle
```

In XQuery 1.0, this would have been expressed as:

```
declare namespace e =
   "http://www.example.com/emp";

doc("employees.xml")//e:jobTitle
```

These URI qualified names can be used almost everywhere that a QName is allowed. They cannot be used in element names and attribute names in element constructors.

Switch expression

The `switch` expression is similar to the switch expressions provided by most programming languages.

The following example uses the `switch` expression to normalize an element value that contains a customer's gender:

```
let $input
   := lower-case (
         normalize-space ($customer/gender)
      )
let $gender
       := switch ($input)
              case "man" return "M"
              case "boy" return "M"
              case "woman" return "F"
              case "girl" return "F"
              default return ()
```

fn:analyze-string function

XQuery 1.0 provides several string functions that make use of regular expressions. `fn:matches` determines whether a string is matched by a regular expression. `fn:replace` replaces matching substrings with replacement strings. `fn:tokenize` breaks a string into multiple pieces, with separators identified by regular expressions.

`fn:analyze-string` takes `fn:match` a step further. Instead of just returning whether or not a string is matched by a regular expression, `fn:analyze-string` identifies which parts of the string did and did not match. It does so by returning an XML element that identifies the matching and non-matching parts of the regular expression.

```
analyze-string
   ("000-11-2222 is a good fellow",
    "(\d\d\d)-(\d\d)-(\d\d\d\d)")
```

→

```
<analyze-string-result
   xmlns=
   "http://www.w3.org/2005/xpath-functions">
   <match>
      <group nr="1">000</group>
      -
      <group nr="2">11</group>
      -
      <group nr="3">2222</group>
   </match>
   <non-match> is a good fellow</non-match>
</analyze-string-result>
```

Indentation has been added to this result to improve its readability.

Each character of the input string appears in either a `match` or a `non-match` element. This means that the string value of the result of `fn:analyze-string` will equal the input string.

Still More Features

There isn't room to describe all of the features that have been added to XQuery 3.0. I'll mention several more of them, without providing detailed descriptions or examples. I encourage you to go directly to the XQuery 3.0 specification for more information on these features – it is quite readable.

External variables can now have default values that will be used if values are not provided by the query's host environment.

The context item can be declared. Both a type and a default value can be specified.

A computed namespace constructor allows a user to create a namespace node. This will most often be used to add to the in-scope namespaces when a user is creating element nodes.

A number of formatting functions have been defined that accept a "picture" argument to generate string representations of numbers, dates, etc.

Serialization declarations can be included to determine how XML results will be serialized.

A type name can be specified in a `validate` expression, requiring that a node, when validated, must be of that type.

Common trigonometric and exponential functions have been defined.

The `fn:parse-xml` function parses its string argument and returns a document node. The `fn:serialize` function serializes its argument and returns a string value.

The `fn:unparsed-text` function uses its URI argument to retrieve a resource and returns a string representation of the resource.

The `fn:available-environment-variables` and `fn:environment-variable` functions provide access to variables in the host environment.

XQuery 3.0 Conformance

Minimal Conformance is the lowest level of conformance that can be claimed for XQuery 3.0. Minimal Conformance encompasses all XQuery 3.0 functionality, with the exception of the following optional features:

- Schema Aware Feature – allows the use of `import schema` in the prolog and allows the use of the `validate` expression.
- Static Typing Feature – requires XQuery to detect and report type errors during the static analysis phase. Some queries that might run successfully without static typing will return an error during static analysis.
- Module Feature – allows the use of `import module` in the prolog and allows library modules to be created.
- Serialization Feature – requires that an implementation provide a way to produce an XML serialization of the result of a query.

- Higher-Order Function Feature – allows dynamic function calls, inline functions, named function references, and partial application of functions.

Completing XQuery 3.0

The XML Query and XSL WGs have published the XQuery 3.0 Test Suite [5] and asked for results from implementers of XQuery 3.0. Positive results from impementors will give us confidence that the specifications are complete and unambiguous. We will then be allowed to publish XQuery 3.0 as a W3C Recommendation.

Work on XQuery 3.1 has already started. The XML Query WG will make this work visible in the near future by publishing a Requirements and Use Cases document.

Jim Melton and I were co-chairing the XML Query WG when this work began. Several years ago I had to resign this position, leaving Jim to complete this work. He has done an excellent job and will deserve hearty congratulations when XQuery 3.0 is published as a Recommendation.

Congratulations will also be due to the editors of our specifications, and to the many members of the XML Query and XSL Working Groups that contributed proposals, raised issues, attended meetings, and wrote test cases.

References

[1] XQuery 3.0: An XML Query Language, W3C Candidate Recommendation, Jonathan Robie, Don Chamberlin, Michael Dyck, John Snelson, 08 January 2013, http://www.w3.org/TR/2013/CR-xquery-30-20130108/.

[2] XQuery 1.0: An XML Query Language (Second Edition), W3C Recommendation, Scott Boag, Don Chamberlin, Mary F. Fernández, Daniela Florescu, Jonathan Robie, Jérôme Siméon, 14 December 2010, http://www.w3.org/TR/xquery/.

[3] XQuery 3.0 Use Cases, W3C Working Draft, Jonathan Robie, Tim Kraska, 08 January 2013, http://www.w3.org/TR/xquery-30-use-cases/.

[4] XQuery 1.0 is Nearing Completion, Andrew Eisenberg and Jim Melton, ACM SIGMOD Record, Vol. 34, No. 4, Dec. 2005, http://www.sigmod.org/publications/sigmod-record/0512/p78-column-eisenberg-melton.pdf.

[5] XQuery/XPath/XSLT 3.* Test Suite, http://www.w3.org/XML/Query/QT3-test-suite/.

[6] Query - W3C, http://www.w3.org/standards/xml/query.

PAIRSE: A Privacy-Preserving Service-Oriented Data Integration System

Djamal Benslimane
LIRIS, Lyon 1 University
69622 Villeurbanne, France
djamal.benslimane@univ-lyon1.fr

Mahmoud Barhamgi
LIRIS, Lyon 1 University
69622 Villeurbanne, France
mahmoud.barhamgi@univ-lyon1.fr

Frederic Cuppens
TELECOM Bretagne
35576 Rennes, France
frederic.cuppens@telecom-bretagne.eu

Franck Morvan
IRIT, Paul Sabatier University
31062 Toulouse, France
franck.morvan@irit.fr

Bruno Defude
TELECOM SudParis
91011 Evry, France
bruno.defude@it-sudparis.eu

Ebrahim Nageba
Claude Bernard University
69622 Villeurbanne, France
ebrahim.nageba@univ-lyon1.fr

ABSTRACT

Privacy is among the key challenges to data integration in many sectors, including healthcare, e-government, etc. The PAIRSE project aims at providing a flexible, loosely-coupled and privacy-preserving data integration system in P2P environments. The project exploits recent Web standards and technologies such as Web services and ontologies to export data from autonomous data providers as reusable services, and proposes the use of service composition as a viable solution to answer data integration needs on the fly. The project proposed new composition algorithms and service/composition execution models that preserve privacy of data manipulated by services and compositions. The proposed integration system was demonstrated at EDBT 2013 and VLDB 2011.

1. INTRODUCTION

Data integration has been a long-standing challenge for the database community. This is motivated by the number of contexts in which the need for a flexible data integration mechanism has become critical, including Web and enterprise data integration, scientific data exploration, data exchange in government agencies, etc.

Much of the literature on data integration across autonomous data sources has tacitly assumed that data on the side of each data source can be revealed and shared with other sources. In practice, however, data integration scenarios are often hampered by legitimate and widespread data privacy concerns. In the healthcare application domain for example, medical data are subject to many legislations (e.g., [2, 1]) around the world that restrict collection, processing, and disclosure of personal data, and hold data holders accountable for any unintended data disclosure or misuse.

The PAIRSE project addresses the challenge of flexible and privacy-preserving data integration in peer-to-peer environments. Driven by the recent trends of using SOA-oriented architectures for data integration in modern enterprises, PAIRSE assumes that data sources are exposed to the data sharing environment as Web services. This type of services is commonly known as data services [11], where data services provide a well documented, platform (and source) independent, interoperable method of interacting with data. PAIRSE proposes a service composition-based approach for on-demand data integration; i.e., heterogeneous data services from autonomous service providers are selected and composed on the fly to answer users' queries. Data privacy preservation is a key objective of PAIRSE. Users in PAIRSE are allowed only to access the information they are entitled to for a given purpose.

PAIRSE focuses on modeling, discovering, selecting and composing data services to efficiently answer users' queries. The contributions of PAIRSE, which was demonstrated at EDBT 2013 [5] and VLDB 2011 [9], are summarized as follows:

- *Semantic description model for data services*: The semantics of data services should be explicitly represented to automate their discovery, selection and composition. We modeled data services as "*RDF Views*" over domain ontologies to formally define their semantics [7]. The service description files (e.g., WSDLs) are annotated with these RDF views.
- *Query resolution by automatic service composition*: Queries in PAIRSE are resolved by *automatically* selecting and composing data services. We exploited mature query rewriting techniques to devise a novel service composition algorithm [7, 9]. The algorithm relieves users from having to manually select and com-

pose services, tasks that would generally require important programming skills. We proposed also an efficient algorithm to locate relevant services in a P2P environment [14].

- *Privacy preservation*: We proposed a privacy preserving composition model [5, 8, 16]. Our model allows services providers to locally enforce their privacy and security policies when their services are invoked. In addition, it prevents services in a composition from learning any information about the data that each other holds, beyond what is permitted.

The rest of the paper is organized as follows. Section 2 gives an overview of our integration system. Section 3 describes our semantic modeling of data services. Section 4 presents our composition approach. Section 5 presents our techniques to privacy preservation. Section 6 applies our work in two application domains, and summarizes obtained results. Section 7 concludes the paper.

2. PAIRSE'S ARCHITECTURE

The PAIRSE data integration system has a hybrid peer-to-peer infrastructure [14], where peers form communities of interest, called *Virtual Organizations VOs*. Each VO has a common domain ontology modeling its expertise, and peer members that may have relations with members from other VOs. Relations between peers exist only if there is a mapping between the ontologies of their respective VOs. PAIRSE does not impose any constraint on the topology graph formed by the ontologies and the different mappings. Peers export their (sharable) data sources as data services.

PAIRSE follows a *declarative* approach to compose data services (Figure 1). Data services in each peer are modeled as *RDF Views* over domain ontologies to explicitly define their semantics. Users formulate their queries on domain ontologies using SPARQL query language. Then, our system exploits the defined RDF views (added as annotations to service description files) to select and compose the relevant services using an RDF query rewriting algorithm that we have devised for that purpose. Queries may necessitate the use of remote data services, in which case an efficient P2P service discovery algorithm [14] is used to locate and retrieve the descriptions of relevant services from remote peers. The system generates then an execution plan for the composition and executes it to provide the user with the requested data. As data services may manipulate privacy-sensitive information, PAIRSE proposed new service and composition execution models to preserve privacy.

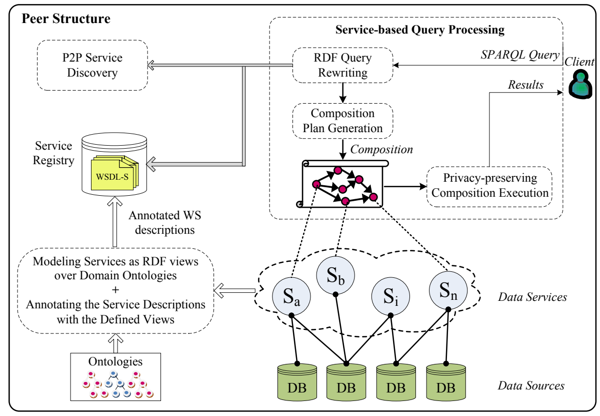


Figure 1: Peer Structure

3. SEMANTIC MODELING AND SERVICE QUERYING

In this section, we explain our service composition based approach to query resolution.

3.1 Semantic Modeling of Data Services

Modeling and explicitly specifying the semantics of data services are the first step towards the automation of service selection and composition.

In PAIRSE, we proposed in [7] to model data services as *RDF Parameterized Views* (RPVs) over domain ontologies. A parameterized RDF view uses *concepts* and *relations* whose meanings are formally defined in domain ontologies to define the semantic relationships between input and output parameters sets of a data service. A parameterized view is a technique that has been used to describe content and access methods in Global-as-View (GaV) integration architectures [13]. Figure 2 shows an RPV of a service returning the personal information (i.e., name and dates of birth) of patients admitted in a given medical center. Note that input parameters are prefixed with the symbol "$" and output parameters are prefixed with the symbol "?".

RDF views may also specify constraints to characterize the data manipulated by their corresponding services. These constraints may have different forms, including *simple interval constraints* (e.g., $X \in [a, b]$, where X is a variable used in an RDF view), and *fuzzy constraints* interpreted according to a fuzzy membership function (e.g., the medications returned by a service have "$High$" concentration of hydroxypropy1-β-cyclodextrin; i.e., X *is* $High$, where the fuzzy term "$High$" is interpreted by a membership function specifying for each value of X the degree to which it is high).

We adopted an approach similar to SAWSDL[1] to

[1] http://www.w3.org/2002/ws/sawsdl/

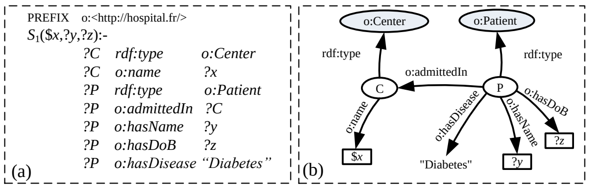


Figure 2: (a) a parameterized RDF view; (b) its graphical representation

associate data services with their RPVs. We exploited the extensibility feature of the WSDL standard to annotate the WSDL files with RPVs.

3.2 Service-based Query Resolution

In PAIRSE, users' queries are resolved by composing relevant data services on the fly. Each virtual organization in PAIRSE's hybrid P2P architecture has a DHT (Distributed Hash Table) to index its published services [14]. Services are indexed according to the ontological concepts used in their RPVs.

When a query is issued at a given peer, relevant services are first sought in the same VO where the query is posed, then the service discovery request is propagated to connected VOs. The descriptions of discovered services are then sent back to the initial peer, where the relevant services will be selected and composed. Furthermore, for each discovered service we return the mapping path between the ontologies associated with the expertise domains (i.e., VOs) of the discovered service and the initial peer. This mapping path allows the translation of RPV views.

We proposed a *query rewriting* based service composition algorithm to select and compose data services on the fly [7, 9]. The algorithm, given a SPARQL query, and a set of data services represented by their RPVs, rewrites the query in terms of calls to relevant services. Our algorithm extends earlier works on query rewriting and data integration [13] in the following aspects:

Compliance with the RDF/S data models: while most of previous work has focused on relational and XML data integration [13, 17], we considered the case of RDF/RDFS data integration. Specifically, our query rewriting algorithm takes into account RDF schema constraints such as *rdfs:subClassOf*, *rdfs:subPropertyOf*, *rdfs:domain*, and *rdfs:range* when comparing RPVs to queries. The consideration of RDFS constraints is important as allows our system to infer more results than the previous rewriting techniques. For example, suppose there is a statement in an RDFS ontology specifying that *:Medication rdfs:subClassOf :Drug*. Given a data service S returning the medications administered to a given patient, and a query Q for the drugs administered to a given patient, our algorithm automatically infers that S can be used to generate rewritings for Q.

Answering parameterized queries: while previous data integration systems have focused on answering specific queries, PAIRSE has focused on answering *parameterized queries*. The key focus was on constructing compositions of services (i.e., parameterized integration plans) that are independent of a particular input value. For example, assume a parameterized query $Q(\$x, ?y)$ for the medications y that may interact with a given medication x. Assume also two data services:

S_1($x,?y), where $x \in$ [1, 5] and $y \in$ [100,150],

S_2($x,?y), where $x \in$ [6,10] and $y \in$ [150,200]

If Q was a specific query ($Q_{x=2}$), then S_2 would not be considered in the rewriting (i.e., composition) as x=2 is not covered by S_2. In contrast, both of S_1 and S_2 are usable for Q, to cover as much as possible of the potential values of x. Our composition algorithm extends the previous ones with: (i) a probabilistic subsumption test to determine in a polynomial time the minimum number of services required to satisfy the value constraints that may be specified on query's parameters [6], and (ii) a mechanism to optimize the generated composition plans based on value constraints specified in service descriptions [7].

Inclusion of user's preferences: often the number of candidate compositions that may be used to answer the same query is very large. We proposed an approach [9] to compute the top-k compositions based on user preferences. In our approach, we modeled user's preferences using fuzzy sets. We match the (fuzzy) constraints of the relevant services to those of the query and determine their matching degrees using a set of matching methods from the fuzzy set theory. We then rank-order candidate services based on a fuzzification of *Pareto dominance* and compute the top-k compositions.

4. PRIVACY PRESERVATION IN PAIRSE

In this section, we briefly present our models to preserve the privacy of manipulated data at the service and the composition levels.

4.1 Privacy-preserving Service Execution Model

Data returned by a data service may be subject to different security and privacy concerns. For example, different people may have different access rights over the same data item; data subjects[2] may

[2]We use the term *data subject* to mean the individual whose information is manipulated by data services.

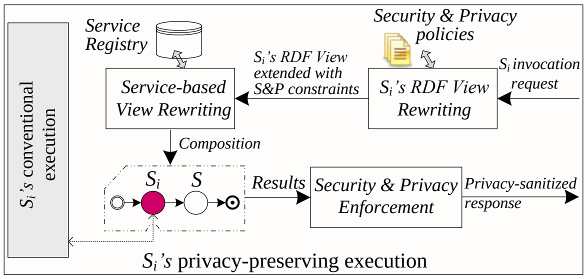


Figure 3: A Privacy-preserving Service Execution Process

have different preferences about the disclosure of their data, etc. A common approach in the database field to handle such concerns is to push them to the underlying DBMS by rewriting the query to include these constraints [15]. However, this may not be applicable to data services as the same service may access a multitude of heterogeneous data sources that may not be necessarily managed by a DBMS. An alternative approach is to enforce privacy and security policies at the application level [4], by modifying, in our case, the source code of data services. However, this may not always be applicable, as most of current data service creation platforms (e.g., *AquaLogic* [11]) provide data services as *black boxes* that cannot be modified. Even if the code was modifiable, this approach often leads to privacy leaks [15].

We proposed a secure, privacy-preserving execution model for data services allowing service providers to enforce their privacy and security policies without changing the implementation of their data services (i.e., services are seen as black boxes). Our model is inspired by the database approach to "*declaratively*" handle the security and privacy concerns. It involves the following steps (refer to Figure 3):
Step 1: *View rewriting to integrate security and privacy constraints.* When a data service is invoked, our model rewrites its corresponding RDF view to take into account applicable security and privacy rules from the service's associated policies, which are expressed using the OrBAC and PrivOrBAC models over domain ontologies and take into account the data *recipient* (i.e., service consumer), his *purpose* for requesting the data, and the *consents* of data subjects [16]. The soundness and correctness of our algorithm are demonstrated in [16, 8].
Step 2: *Rewriting the extended view in terms of data services.* The extended RDF view $v_{extended}$ may include additional data items (denoted by $\Delta v = v_{extended} - v_{original}$) required to enforce security and privacy constraints. In this step, we find the data services covering Δv, and rewrites $v_{extended}$ in terms of these services along with the initial service.
Step 3: *Enforcing security and privacy constraints.* Services selected in the previous step are composed and executed using the conventional service execution process. The composition returns (i) the data items returned by the invoked service along with (ii) the data items necessary to evaluate the security and privacy constraints. We defined a privacy filter that evaluates the privacy constraints of the different items that are subject to privacy constraints in the view. *Null* values will be returned for items whose privacy constraints evaluate to *False*.

We demonstrated the validity of our model by extending the architecture of the famous service container AXIS[3] 2.0 with a new module implementing our privacy-preserving service execution model.

4.2 Privacy-preserving Composition Execution Model

Executing compositions may disclose confidential information to component services. Assume, for example, a composition of two services: S_1 returns HIV patients in a given city, and S_2 checks whether a given patient has been treated for psychiatric disorders. Such composition could be needed (by a pharmaceutical researcher) to investigate the connection between a chemical component present in HIV medicines and the development of severe psychiatric disorders. Assume also *Bob* is a common patient to S_1 and S_2. If S_2 is invoked with Bob's identifier, and the provider of S_2 has an access to the composition plan (i.e., he knows that Bob was outputted by S_1), then he will infer that Bob is an HIV patient. On the other hand, if the data returned by S_1 were completely privacy-sanitized (e.g., by removing identifiers and sensitive information), then the composition could not be executed.

We proposed a privacy-preserving composition execution model in [5] that limits the information disclosed to services in a composition about the data that each other holds. Our model distinguishes between the following entities: (i) the services in the composition, (ii) the execution engine, and (iii) the recipient of final results. It relies on two key ideas: First, data services use the same order-preserving encryption scheme OPES [3] to encrypt the identifier attributes that are needed to connect data subjects across the different services. They are still free to protect non-identifier attributes with their own techniques (e.g., anonymization, etc.). This way the execution engine has only access to protected data and can still link data subjects across services using the encrypted identifier attributes (note that OPES allows for applying equality queries on encrypted

[3]http://axis.apache.org/axis2/java/core/

data). By the end of the composition execution, it removes from the final results the encrypted identifier attributes before returning them to the recipient, who will thus get only privacy-sanitized data. Second, we proposed a algorithm to allow the execution engine to generalize the encrypted value v_e received from a service S_i before proceeding with the invocation of the subsequent service S_j in the composition, such that the generalized value Gen(v_e) corresponds to k input encrypted values for which S_j has outputs; e.g., the identifier of Bob is generalized to cover k-1 other patients for which S_2 has an output (i.e., S_2 will not be able to distinguish between Bob and k-1 other patients).

5. IMPLEMENTATION AND EVALUATION

We evaluated our different techniques and algorithms in the healthcare and bioinformatics application domains. These domains have widely embraced Web standards, such as XML and Web services [12, 10], and are characterized by the need for a flexible and privacy-preserving data integration approach.

The cardiology hospital of Lyon provided us with access to two medical databases. The identities of patients in these databases were changed. We also generated synthetic medical data about the same patients. We implemented about /400/ data Web services on top of our real and synthetic data. Services were all deployed on an extended version of AXIS 2.0 implementing our service execution model. We built a medical ontology based on the building blocks and the data types defined in the HL7 standard, and used it for the annotation of service description files. To evaluate our techniques in the bioinformatics domain, we used a set of /300/ services from the *BioCatalogue registry*[4].

Figure 4 (part *a*) shows the query interface to PAIRSE. Users are assisted in formulating their SPARQL queries over domain ontologies. The figure shows (in part *b*) also the composition plan of a selected composition, along with the privacy-sanitized results (part *c*).

We conducted exhaustive experiments to evaluate the performance of our integration system. We summarize below obtained results[5]:

Composition construction and execution: Our experiments in [7] showed that our composition algorithm can handle hundreds of data services in a reasonable time. For example, for chain queries [13] and RPVs with a length of 3 or 4 object properties the algorithm was able to handle up to 400 services in less than 4 seconds.In the context of parameterized queries, our experiments in [6] showed that our algorithm to find the minimum set of services introduced only a small cost at the composition construction time (i.e., in all experiments the algorithm required less than 10% of the time needed to rewrite the query), and improved substantially the composition execution time (i.e., in all experiments the composition execution time was reduced to less than 0.75% of the time needed without optimization), as it removes redundant services. In the context of preferences queries, our experiments in [9] considered that services can be grouped in classes. The experiments showed that the top-k compositions can be computed efficiently. For instance, for classes containing about 400 services, the top-k compositions are computed in less than 4 seconds.

Security and privacy preservation: The conducted experiments in [8] showed that our secure and privacy preserving service execution model added only a small increase to the service execution time. In all experiments, the cost incurred in the enforcement of security and privacy constraints did not exceed 10% of the time required to execute the service with ignoring these security and privacy constraints altogether. The conducted experiments for the evaluation of our composition execution model [5] showed that the time required to execute the composition with privacy preservation is at most three orders of magnitude of the time required without privacy preservation (K_i was set to 4 in all tests). We were able to cut down that cost to two orders of magnitude by *reusing* the values of the protocol parameters that were computed in past invocations of the same services (and during the same composition execution).

[4]http://www.biocatalogue.org/

[5]For detailed information about the considered settings in each cited experiment, please refer to the corresponding paper.

6. CONCLUSION

The goal of the PAIRSE project was to develop new methods and techniques for flexible and privacy-preserving data integration. We have evaluated our composition-based approach in the healthcare and the bioinformatics domains. The obtained results [5, 9, 7, 14, 8, 16, 6] are promising.

7. ACKNOWLEDGMENTS

PAIRSE is funded by the French National Research Agency under the grant number ANR-09-SEGI-008

8. ADDITIONAL AUTHORS

Michael Mrissa (Lyon 1 University, `michael.mrissa@univ-lyon1.fr`), Francois Paulus (Semsoft Company, `francois.paulus@semsoft-corp.com`), Stephane Morucci (Swid Company, `stephane.morucci@swid.fr`), Nora Cuppens (Telecom-Bretagne, `nora.cuppens@telecombretagne`

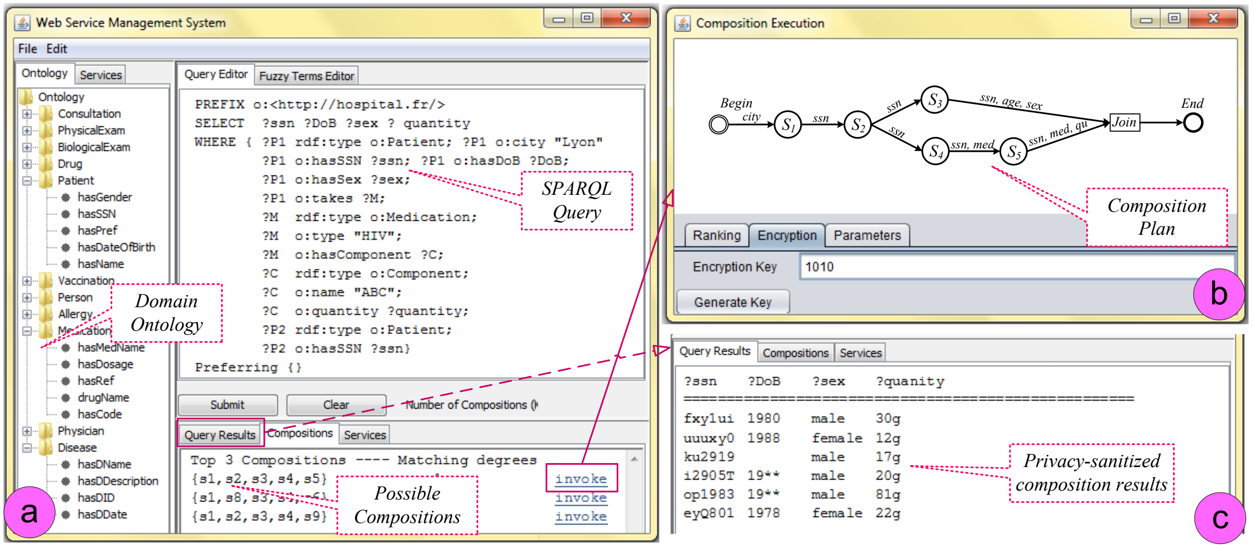


Figure 4: (a) Query interface; (b) The selected composition; (c) Obtained results

.eu), Chirine Ghedira (Lyon 3 University, chirine.ghedira-guegan@univ-lyon3.fr), Riad Mokadem (Paul Sabatier University, Riad.Mokadem@irit.fr), Said Oulmakhzoune (Telecom-Bretagne, said.oulma khzoune@swid.fr) and Jocelyne FAYN (INSA-Lyon, Jocelyne.Fayn@insa-lyon.fr).

9. REFERENCES

[1] E. u. directive on data protection, official journal of the european communities, 23 november 1995.

[2] Health insurance portability and accountability act of 1996, united states public law 104-191.

[3] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order-preserving encryption for numeric data. In *SIGMOD Conference*, pages 563–574, 2004.

[4] P. Ashley and D. Moore. Enforcing privacy within an enterprise using ibm tivoli privacy manager for e-business. In *VLDB*, pages 108–119, 2003.

[5] M. Barhamgi, D. Benslimane, Y. Amghar, N. Cuppens-Boulahia, and F. Cuppens. Privcomp: a privacy-aware data service composition system. In *EDBT*, pages 757–760, 2013.

[6] M. Barhamgi, D. Benslimane, C. Ghedira, S.-E. Tbahriti, and M. Mrissa. Optimizing daas web service based data mashups. In *IEEE SCC*, pages 464–471, 2011.

[7] M. Barhamgi, D. Benslimane, and B. Medjahed. A query rewriting approach for web service composition. *IEEE Transactions on Services Computing*, 3(3):206–222, 2010.

[8] M. Barhamgi, D. Benslimane, S. Oulmakhzoune, N. Cuppens-Boulahia, F. Cuppens, M. Mrissa, and H. Taktak. Secure and privacy-preserving execution model for data services. In *CAiSE*, pages 35–50, 2013.

[9] K. Benouaret, D. Benslimane, A. HadjAli, and M. Barhamgi. Fudocs: A web service composition system based on fuzzy dominance for preference query answering. *PVLDB*, 4(12):1430–1433, 2011.

[10] J. Bhagat, F. Tanoh, E. Nzuobontane, T. Laurent, J. Orlowski, and M. Roos. Biocatalogue: a universal catalogue of web services for the life sciences. *Nucleic Acids Research*, 38(5):689–694, 2010.

[11] M. J. Carey, N. Onose, and M. Petropoulos. Data services. *Commun. ACM*, 1(1):2–9, 2012.

[12] A. Dogac. Interoperability in ehealth systems. *PVLDB*, 5(12):2026–2027, 2012.

[13] A. Y. Halevy. Answering queries using views: A survey. *VLDB J.*, 10(4):270–294, 2001.

[14] I. Ketata, R. Mokadem, and F. Morvan. Resource discovery considering semantic properties in data grid environments. In *Globe*, pages 61–72, 2011.

[15] K. LeFevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, and Y. Xu. Limiting disclosure in hippocratic databases. In *VLDB*, pages 08–19, 2004.

[16] S. Oulmakhzoune, N. Cuppens-Boulahia, F. Cuppens, and S. Morucci. fquery: Sparql query rewriting to enforce data confidentiality. In *DBSec*, pages 146–161, 2010.

[17] C. Yu and L. Popa. Constraint-based xml query rewriting for data integration. In *SIGMOD Conference*, pages 371–382, 2004.

Medical Data Management in the SYSEO Project

Yahia Chabane*, Laurent d'Orazio*, Le Gruenwald**, Baraa Mohamad* and Christophe Rey*
* Blaise Pascal University, LIMOS UMR 6158, CNRS, Clermont-Ferrand, France
** School of Computer Science, University of Oklahoma, Oklahoma, United States
*firstname.lastname@univ-bpclermont.fr, **ggruenwald@ou.edu

ABSTRACT

The SYSEO project aims at producing a software solution suitable for endoscopic imaging in order to enable physicians to manage, manipulate and share medical images. This paper presents our two main components for data management in this system: (1) a novel hybrid row-column database for medical data storage within the cloud and (2) a system for semantic image annotation and retrieval relying on an ontology for polyps.

1. INTRODUCTION

Medical data management has become an important challenge. The emergence of new medical imaging techniques and the necessity to access medical data at any time have led to a need to find new solutions for managing these data. Our work focuses on Digital Imaging and Communications in Medicine (DICOM) [1], one of the most important medical standards. DICOM aims to achieve interoperability between medical imaging systems. The requirement in storing DICOM-based images is that all the study related information be stored within the image file so that the image can never be separated from its information by mistake. The wide use of DICOM standard has led to the development of some management systems, such as the Picture Archiving and Communication System (PACS) [2] and the ORDICOM (object-oriented) data type in Oracle 11G [3]. Unfortunately, such systems are highly expensive, IT experts dependent and not scalable. Particularly, in these systems the crash of a server may prevent accessing the required images. Moreover they do not include facilities to efficiently and intuitively annotate and retrieve images. Particularly, no semantic retrieval technique has been developed yet in this field, in spite of efforts in medical ontology building (see section 2).

To fill this gap, we have developed SYSEO [4] that includes facilities to deal with medical data effectively and efficiently. Indeed, it aims at producing a comprehensive software solution that enables physicians to acquire, enrich, store, retrieve, manipulate, share and export medical images easily.

Data management in SYSEO addresses the following key challenge: how to best store and query huge quantities of DICOM images and videos? Three dimensions have been investigated: (i) performance (allowing huge data to be stored and quickly accessed), (ii) relevance (allowing query results to be the most precise possible) and (iii) completeness (allowing the use of many query mechanisms and their associated advantages).

Our first proposal considers performance through a cloud-enabled and hybrid medical data management system. Such a system first takes advantage of the cloud features to provide a highly available and cost-effective solution; then it provides an appropriate storage model that overcomes the intense heterogeneity, complexity and huge size of DICOM files and, at the same time, provides high expressiveness. To deal with relevance, our second proposal is a semantic system that allows images to be intuitively annotated and retrieved via semantic web techniques. This semantic system implements a complete semantic approach for endoscopic polyp images annotation and retrieval. It is based on a polyp ontology we have developed. Our goal is to provide more relevance compared to classical syntactic search. Moreover, this ontology is the first step towards a reference library of annotated polyp images that physicians may use in their everyday practice. This article aims at showing how completeness is reached in SYSEO using our proposals and the various querying modalities it supplies.

This paper is organized as follows. Section 2 presents related works. Sections 3 and 4 present the hybrid cloud-based data store and the ontology-based system, respectively. Section 5 presents the system implementation. Finally, Section 6 concludes the paper with a discussion of future research.

2. RELATED WORK

2.1 Medical Data Stores

Picture Archiving Systems (PACS)[2] systems are used in many medical centers. They are very expensive and low-expressive (pre-defined queries). Additionally they do not cope with heterogeneity since they mostly use a relational database that stores all heterogeneous attributes in a blob-like datatype without any ability to interrogate them.

Oracle 11g introduces a DICOM support feature [3], consisting of a new data type ORDicom that allow any column of this type in a table to hold DICOM content. Even though Oracle provides indexing and compression techniques, each DICOM file is stored in a separate object, leading to significant data redundancy, and as a consequence, increasing the storage space and reducing the performance, especially when using certain DICOM-specific methods.

eDiaMoND [5] is a grid-enabled medical imaging database, that relies on an object-relational approach to store DICOM files. It supports only three modalities (secondary capture images, mammography x-Ray images and structured reports) and restricts users to a set of pre-determined queries. This system is designed over a grid (a structure with a limited number of dedicated severs); therefore it is not suitable for a huge infrastructure of unreliable machines (such as the cloud).

Commercial cloud-based medical systems exist, such as DicomGrid [6], but without any documentation or research papers about them.

2.2 Reasoning and Ontologies to manage Medical Images

The first purpose of medical ontologies [7] is to gather existing taxonomies so as to link together concepts having the same meaning but a different name [8, 9]. We refer to [10] for a more complete discussion of existing ontologies in medicine.

A concrete usage of medical ontologies is image annotation, especially in the case of syntactic keyword-based image retrieval system. The Medico scenario in the Theseus project [11] aims at setting up standards for the syntax and semantics in medical image annotation from ontologies. Our approach is quite similar in that we handle the annotation and retrieval problems using description logics. However, our aim is less oriented towards diagnosis than towards giving physicians a semantic infrastructure to manage their medical images. The AIM project [12] aims at setting up an ontology-based standard for the annotation and the markup of medical images. Our approach differs in that we put the semantic capabilities at the heart of the system since we use a true ontology (not a lexicon) based on a Description Logic (DL) [13] and associated with precisely defined reasoning. The semantic features seem not to be a main objective in the AIM project. Other works handle the issue of semantic image annotation [14, 15]. Our proposal is close to these works, but is different in the used retrieval reasoning.

Concerning DL image retrieval reasoning, what differs from one approach to another is the proximity notion that is used to qualify the good answer images to a query. We can find two classical approaches [16, 17, 14] which correspond to our R1 (see Table 1), which is the classical individuals retrieval, and the composition of R2 followed by R1, which amounts to finding images associated with concepts that have the same properties as the query (and maybe others properties). Other approaches are based on non-standard DL reasoning (abduction and contraction) [18, 19], which imply, however, the use of a less expressive DL. This reasoning enables a better ranking of answers than the previous one.

3. HYBRID CLOUD-BASED DICOM DATA MANAGEMENT SYSTEM

The architecture of SYSEO's DICOM Management System is illustrated in Figure 1. The implemented system shows interesting results to store and query DICOM files. We present in the following sections the main components in our solution: the data storage and the query execution.

3.1 Data Storage

The DICOM standard defines more than 3000 attributes. Only some of them are mandatory to be inserted in a given DICOM file, whereas the others are optional. Therefore, each DICOM file could contain a different subset of these attributes. Modifications/additions can be proposed by do-main experts resulting in a new version of the standard every year, so the schema is changing over time. We propose a hybrid (row-column) two cloud-based

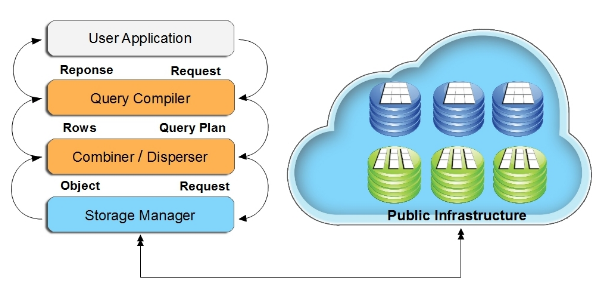


Figure 1: Data Management Architecture

layers data storage structure. Each of these layers is designed to store a special set of DICOM attributes. For that, we decompose the attributes into three categories: (1) Mandatory/frequently used attributes, (2) frequently accessed together attributes; and (3) optional/private attributes. Then, we propose the most appropriate layer to store each of them. We link these layers together by a unique identifier that allows us to reconstruct our DICOM files. Both (row-column) layers are cloud-based, which ensures the elasticity and fault tolerance for each of them (e.g. with GFS [20] storing automatically several copies of our data in geographically separated areas, a server crash is not a problem).

We propose to store mandatory/frequently used attributes (e.g. PatientID and Pixel Representation) and frequently accessed together attributes (e.g. Patient Name and Birth-date) in a traditional row-oriented database layer. We store frequenlty accessed together attributes in the same table in order to reduce join costs (tuple reconstruction time) by applying appropriate vertical partitioning. The advantage of this layer is its write-optimized feature (each tuple insertion in row-oriented databases needs one disk block I/O for insertion alone). Thus, having a lot of insertions over this layer will not be challenging. A sharded database, like Azure [21] or RDS [22], is a candidate solution for such a layer. However, in order to reduce the storage cost and have a more scalable solution, we focus on shared nothing MapReduce based approaches like Pig[23] or Hive [24].

Optional/private attributes (e.g., Smoking Status and Chemical Shift) vary enormously from one medical file/center to another. For these heterogeneous attributes, we propose storing them in column-oriented databases. Only non-null values will be inserted into their corresponding columns which improves significantly the query performance. Therefore, this model copes perfectly with our heterogeneous data. Columnar databases are OLAP-optimized, so this layer offers the ability to perform efficiently ad-hoc/statistical queries which are very selective. Additionally, this storage model provides a good solution for the evolutive schema issue since each column is stored in a separate disk block, so adding new columns is not challenging. On the other hand, the attributes stored in this layer are less frequently accessed together, so we minimize the tuple reconstruction time. A number of cloud columnar systems can be possible solutions for this layer. Examples are BigTable [25], Vertica [26] and HBase [27]. The high cost and proprietary features of Vertica and BigTable (GFS dependent) lead us to focusing on the other systems (e.g. HBase).

To improve the dynamicity of this storage, we plan to implement a column mover, which is a process that moves (when necessary) some attributes from one layer to the other when needed. A similar idea has been implemented by SAP database [28].

3.2 Query Execution

The query execution engine contains two main components: the Query Compiler and the Combiner/Disperser.

The Query Compiler is responsible for compiling user requests and translating them into a cloud query language. Actual systems (e.g. Pig, Hive) - under heavy development - have some limitations such as the absence of metadata/schema in some of them, and/or the lack of some functionalities (e.g. join). Therefore, new operations and optimizations will be added to adapt the used system to our hybrid storage.

The Combiner/Disperser is responsible for partitioning the coming queries according to the layers (row-oriented, column-oriented). After the query execution, the Combiner/Disperser is in charge of combining the results coming from both storage layers and sending the final results to the user.

In order to provide a good compromise between storage cost and query response time, we propose a query optimizer. It is responsible for choosing the best query plan/execution order (i.e. column layer first, or row layer first, or parallel execution of both layers) for executing the query over our hybrid storage model.

The query optimizer applies a Cost/Rule Based Optimization [29]. Yet the existing CBO/RBO solutions should be rethought for the cloud by taking into account the pay-per-use and elasticity features. In this context, we distinguish two query types. The first is the real time search where the physician may need certain images rapidly. In this case, the response time is crucial; so we may increase the number of resources used from the cloud according to the Service Level Agreement (SLA) [30].The second is the data analysis where the response time is not crucial; so we can reduce the used resources. Hence we maintain a good correlation between response time and resource cost.

4. USING SEMANTICS

The architecture presented in the previous section provides storage and querying capabilities on the DICOM attributes. Yet, there is not any standard set of attributes for a practitioner to store his/her observations of an image. In the best case, he/she stores them as full text in a private DICOM attribute. The consequence is that query relevance may be low on these observations since physicians do not always use a standard vocabulary. That is why we propose to address the querying problem using a new ontology following a semantic web approach, namely a description logic approach.

Description Logic (DL) [13] is a well-known knowledge representation and reasoning formalism [13]. The OWL language [31], one of the main standards in semantic technologies, is based on DL. We now present the content of the polyp ontology, the annotation and query mechanisms.

4.1 The Polyp Ontology

The ontology is divided into three main parts related to the observable properties of the image content (colors, shapes, textures, etc.), its anatomical properties and the medical diagnosis comments on it. An image annotation is then defined as a set of information coming from these three parts. The base gastroenterological concepts come from four standard classifications that have been integrated in the ontology: (Paris, PitPattern, Vienne and MST which describe polyp shapes, polyp surfaces, polyp pathological states and many gastroenterological concepts respectively). Each concept coming from a classification and denoting a special set of polyps is called a class.

The language we choose to build our ontology is $\mathcal{SHOIQ}^+$ [32]. It is a very expressive DL for which the powerful HermiT reasoner is built [33].

4.2 Annotating and Querying

The process of our semantic image retrieval approach is illustrated in Figure 2.

First, DICOM images are stored in a cloud database (1). The ontology (2) is linked to this database via a keyword database (3). In the keyword database are stored image identifiers linked with keywords which are concepts taken from the ontology. Moreover image identifiers are also stored in the ontology as individuals that are instances of their associated image annotations. Two modules (4) and (5) ensure the coherency among (1), (2) and (3). Upon this knowledge infrastructure, the semantic image retrieval process runs as follows. First the system displays the concept hierarchy computed from the ontology (6). Then the user can browse it (7) and select a set of keywords which are concepts of the ontology (8). This set is then mapped to the generic definition of an image to obtain an image annotation (9). So, such an annotation is an instance of the generic definition of an image. Af-

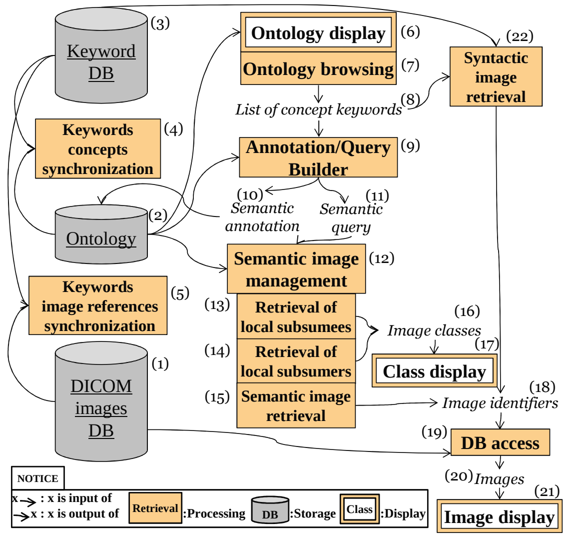


Figure 2: Semantic Image Retrieval Approach

#	**Scenario**	#	**Reasoning**	Fig. 2
S1	Semantic images retrieval	R1	Individual retrieval	(15)
S2	Exact classes retrieval	R2	Retrieval of local subsumees	(13)
S3	Approximated classes retrieval	R3	Retrieval of local subsumers	(14)

Table 1: Scenarios and corresponding reasonings.

terwards, this annotation can either be stored in the ontology (this is the annotation scenario)(10), or this annotation can be viewed as a query (this is the semantic retrieval scenario)(11).

We define three semantic retrieval cases (12): S1 for semantic images retrieval (15), S2 for exact class retrieval (13) and S3 for approximated class retrieval (14). We propose three kinds of DL reasoning (R1, R2 and R3) to implement them (see Table 1). Reasoning R1 is well-known in the DL litterature: we want to find all individuals (image identifiers) that belong to a given concept description (the annotation). Reasoning R2 and R3 are mainly based on subsumption (i.e. the subclass/superclass relation among concepts). In R2, we find all the subclasses of a given classification (e.g. Paris or PitPattern) that are also subclasses of the query. In R3, we find all the superclasses of the query that are subclasses of a given classification. That is why it is an approximation reasoning. Once image identifiers have been obtained (18), the system looks for (19) their associated DICOM images (20). Then the images can be displayed (21). Once the image classes have been inferred (16), they can be displayed to the user (17). A very interesting feature in this process is that the semantic part can be inserted within a classical syntactic retrieval. Indeed, once the list of keywords is known (8), a keyword-based search engine can be run (22) to retrieve image identifiers (18) from the keyword database (3).

To conclude this part, we point out how the cloud-based and the semantic query mechanisms complement each other. The former allows efficient image retrieval with a classical syntactical approach on DICOM attributes. The latter allows less efficient but more relevant image retrieval with a semantic approach on semantic descriptions of the images that are not already stored as DICOM attributes. Moreover, classical syntactic retrieval can be achieved over concepts from these semantic descriptions. That is why we claim that the Syséo query mechanism is complete.

5. IMPLEMENTATION

5.1 Implementation Details

For the development of our application, we have used Struts 2 framework and servlets for the implementation of the MVC pattern, and JSP for the user-interface creation.

For the Storage system we have built the row-oriented layer using Pig.We have simulated the columnar storage by the use of ZEBRA library over Pig. We have developed dedicated user-defined functions for the parsing and decomposition of DICOM files for the corresponding layers. The user-interface is dynamic to allow the user create easily his/her query. This query is then translated into the corresponding Pig query language. Our current work is about assuring efficient query execution over our storage structure by proposing new optimizers.

For the semantic part, we have used the OWL API for ontology manipulating, the HermiT reasoner [33] for reasoning on the ontology and Prefuse [34] for creating user interfaces of image annotations and query generation.

5.2 Example

We show in Figure 3 an example illustrated within a general schema of our system.

Adding Images: When the physician wants to save a new image, the DICOM parser reads and decomposes the image into three categories: 1) attributes should be sent to the private infrastructure (health care data center) (e.g.Patient Name), 2) attributes to be stored in the row-oriented layer (e.g. Patient ID) and 3) attributes will be stored in the column storage layer (e.g. Pregnancy Status).

Retrieving Images/Statistics: The user uses the user-interface to create her query. The query is then written in the PigLatin query language. The Disperser rewrites the query according to the location of each of the required tables/attributes (e.g. Patient ID, Sex and Birth date attributes belong to the patient table on the row-oriented layer whereas SmokingStatus resides on the columnar layer).

Semantic Annotation and Retrieval: Image annotation and queries are generated manually using an interactive user-interface. This interface allows navigation in the ontology. According to the physician observation/need, she selects the most appropriate concepts and individuals for the representation of images.

The annotation (query) mechanism building is illustrated in Figure 3. The user selects three concepts of ontology: stomach, orange and haemorrhaic. The subsumers (belonging to the annotation concept definition) of these concepts will be determined in order to select the most appropriate roles for each concept. Thereafter, a concept description is built from these subsumers and roles. The result is the user annotation (or the user query).

6. CONCLUSION

In this paper, we presented the data management designs in the SYSEO project. We introduced a cloud-enabled hybrid database and semantic approach for medical data management. The challenges in this context are due to the high heterogeneity and huge volumes of DICOM files. For that we propose a new architecture providing: (1) ease of use, high performance and ad-hoc queries over DICOM files, (2) the capacity to exploit the cloud elasticity, billing-by-use and scalability and (3) give a complete and flexible semantic infrastructure to manage medical images, diagnosis and education.

The next objective of our project is to validate our prototype for real medical applications. We are about to integrate different solutions and install them in hospitals in order to validate the solutions and the Ontology. We plan to achieve a high level of QoS that allows querying large amounts of data via different types of computing devices. Additionally, some optimization (e.g. materialized views, cache manager, semantic reasoning) should be rethought for our particular structure. In the near future, we will provide more details about the prototypes and results on the project Web site [4].

7. ACKNOWLEDGEMENT

This work is supported by Yansys, the Agence Nationale de la Recherche (under grant SYSEO ANR-10-TECSAN-005-01), and the Conseil Régional d'Auvergne.

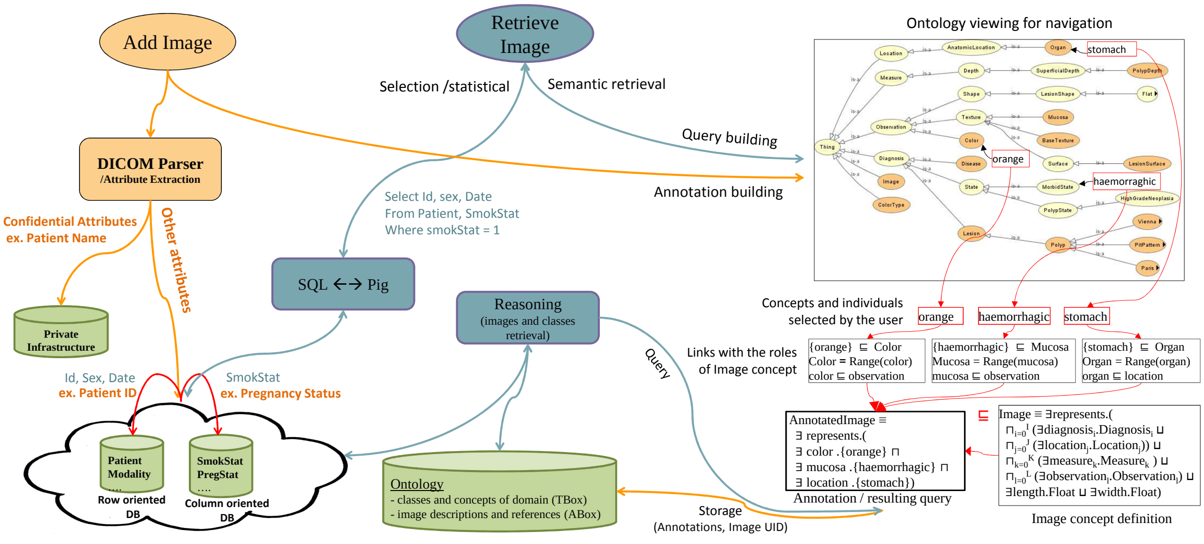


Figure 3: Image Adding and Retrieval

8. REFERENCES

[1] "Dicom," 2013. http://medical.nema.org/.

[2] M. Tieche, J. Gump, M. E. Rieck, and A. Schneider, "PACS: Picture Archiving and Communication Systems," *Springer*, 2010.

[3] Oracle, "Oracle Database 11g DICOM Medical Image Support," Tech. Rep. September, 2009.

[4] "Syseo," 2013. http://www.syseo-anr.fr/.

[5] D. J. Power, E. A. Politou, M. Slaymaker, S. Harris, and A. C. Simpson, "A relational approach to the capture of dicom files for grid-enabled medical imaging databases," in *SAC*, pp. 272–279, 2004.

[6] "Dicomgrid," 2013. http://www.dicomgrid.com.

[7] "Openclinical: knowledge management for medical care," 2011. http://www.openclinical.org/ontologies.html/.

[8] "Galen and the galen-core high-level ontology for medicine.," 2011. http://www.opengalen.org/.

[9] U. N. L. of Medicine, "Systematized nomenclature of medicine - clinical terms," 2011. http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html.

[10] "Bioportal," 2011. http://bioportal.bioontology.org/.

[11] G. F. M. of Economics and Technology, "Theseus project, medico scenario," 2010. http://theseus.pt-dlr.de/en/920.php.

[12] S. C. for Biomedical Informatics Research., "Aim project.," 2010. http://cabig.cancer.gov/solutions/applications/aim/.

[13] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, eds., *The Description Logic Handbook: Theory, Implementation, and Applications (2nd Edition)*, Cambridge University Press, 2007.

[14] D. L. Rubin, P. Mongkolwat, V. Kleper, K. Supekar, and D. S. Channin, "Medical imaging on the semantic web: Annotation and image markup.," in *AAAI Spring Symposium: SSKI*, pp. 93–98, AAAI, 2008.

[15] P. Wennerberg, K. Schulz, and P. Buitelaar, "Ontology modularization to improve semantic medical image annotation.," *JBIOBL*, vol. 44, no. 1, pp. 155–162, 2011.

[16] E. D. Sciascio, F. M. Donini, and M. Mongiello, "Semantic indexing in image retrieval using description logic," in *ITI*, 2000.

[17] B. Hu, S. Dasmahapatra, P. H. Lewis, and N. Shadbolt, "Ontology-based medical image annotation with description logics," in *ICTAI*, pp. 77–, 2003.

[18] T. Di Noia, E. Di Sciascio, F. M. Donini, F. di Cugno, and E. Tinelli, "Non-standard inferences for knowledge-based image retrieval," in *EWIMT 2005, IEE press*, pp. 191–197, IEE, 2005.

[19] S. Colucci, T. D. Noia, E. D. Sciascio, F. M. Donini, and M. Mongiello., *Description Logic-Based Resource Retrieval.*, pp. 185–197. Encyclopedia of Knowledge Management, 2011.

[20] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *SOSP*, pp. 29–43, 2003.

[21] "Windows azure," 2011. http://www.microsoft.com/windowsazure/.

[22] "Amazon rds," 2012. http://aws.amazon.com/rds/.

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

[24] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, "Hive - a warehousing solution over a map-reduce framework," *PVLDB*, vol. 2, no. 2, pp. 1626–1629, 2009.

[25] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. Gruber, "Bigtable: A distributed storage system for structured data," *ACM TOCS*, vol. 26, 2008.

[26] "Hp vertica," 2011. http://www.vertica.com/.

[27] "Apache hbase," 2011. http://hbase.apache.org/.

[28] P. Rösch, L. Dannecker, F. Färber, and G. Hackenbroich, "A storage advisor for hybrid-store databases," *Proc. VLDB*, vol. 5, no. 12, 2012.

[29] H. Pirahesh, J. M. Hellerstein, and W. Hasan, "Extensible/rule based query rewrite optimization in starburst," in *SIGMOD*, pp. 39–48, 1992.

[30] S. A. Baset, "Cloud slas: present and future," *SIGOPS*, vol. 46, July 2012.

[31] "Owl, the web ontology language," 2007. http://www.w3.org/2007/OWL.

[32] B. Motik, R. Shearer, and I. Horrocks, "Hypertableau reasoning for description logics," *JAIR*, vol. 36, pp. 165–228, 2009.

[33] "Hermit," 2012. http://www.hermit-reasoner.com/.

[34] J. Heer, S. K. Card, and J. A. Landay, "prefuse: a toolkit for interactive information visualization," in *CHI*, pp. 421–430, 2005.

Andreas Reuter Speaks Out on Transactions, Reinventing things, Creating a University, and more

by Marianne Winslett and Vanessa Braganholo

Andreas Reuter
http://www.h-its.org/english/homes/reuter

Welcome to ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today we are in Indianapolis, site of the 2010 SIGMOD and PODS conferences. I have here with me Andreas Reuter, who is a professor at the University of Heidelberg[1] and the director of two research institutes in Heidelberg. Andreas was a freelance programmer for more than ten years. He is the coauthor with Jim Gray of the famous textbook Transaction Processing Concepts and Techniques[2]. His PhD is from the Technical University of Darmstadt. So, Andreas, welcome!

---

[1] When this interview was conducted (2010), Andreas was at Kaiserslautern University.

[2] Jim Gray, Andreas Reuter. Transaction Processing: Concepts and Techniques. Morgan Kaufmann 1993, ISBN 1-55860-190-2.

How was it to write that book with Jim Gray?

Oh, it was a once in a lifetime experience. It was very strenuous. We had a lot of fun. We had many debates, arguments and so on. I think the whole atmosphere is best summarized by what Jim said right at the end of the writing experience, “I’m totally stunned that we didn’t kill each other!”

How long did it take?

It took about four times longer than we anticipated. We wanted to write a book covering transactions, databases, everything in about 600 pages. We had a set of slides prepared for a course that we taught in Berlin in the late ‘80s. And we said, well, we’ve got all these slides, we just have to turn them into prose, it shouldn’t be too hard to do, and so, let’s do it. So, Jim went back to America, and I stayed in Germany, and nothing happened. After 2 years of nothing having happened, we decided that if we ever wanted to get this book out, we should do something about it. So we arranged to have a writing assignment for two or three months in Italy.

It was a long argument where we should go in early spring, between middle February to April. Jim said, “I don’t want to go to Germany. It’s cold, it’s raining, and it’s terrible. I’m from California. I need it warmer. I need sunshine.” So we rented a place in northern Italy and we started writing. And we arrived in wonderful weather; it was sunny, bright skies. It was pretty cold, but bright skies. One week after we arrived, it started raining and never stopped again. We wrote. We had a couple of power failures; the power system is not overly reliable. Jim got more and more upset about this whole thing, the unreliable infrastructure, and no telephone, it was the late ‘80s. So we decided to pack things, and drive back to Germany, and in Germany, it was warm spring. That was the irony. After these three months, we had about half the subjects covered. We decided to have another writing assignment the next year. And Jim said, “I’ve had it in Europe in spring, this is all nonsense. We do the next thing in California! I’ll rent the place.” So he rented a place in Bolinas, north of San Francisco, about the same period of time: February to April. We arrived, and it started raining, and it never stopped.

First build the code, and then write the paper.

And it was foggy too, up there?

Yeah, it was foggy, but it was more rainy than foggy. If there is enough rain, then there is no fog. Close to when we finished up our writing in Bolinas, there was an announcement on the news that the drought was over. That sort of added to the general tension of getting this stuff organized, and getting the right angle on things, and so on. We decided by the end of the Bolinas assignment that this was it. Whatever we have, this is the book. And we hadn’t covered all of the subjects we wanted to cover, there is no SQL, there is no database design, there is no application design in the book, which originally was part of the plan. But we just said, well, that’s it. It’s 1100 pages now, and the rest is sort of editing, and polishing, and doing the index, etc.

But back to your original question, how was it? It was, as I said, technically, and from an emotional perspective, from a human perspective, one of the greatest experiences I have ever had. We necessarily grew very close to each other, just through the narrow confinement of this writing closure. There was nobody else. It was Jim and I for about 3 months. Once in a while, Jim's wife would come up, but she came from San Francisco, stayed overnight, and went back. My wife was back in Germany, and so, there was just nothing else going on 18 hours a day. That was roughly the period of time that we spent per day.

Is there life after transactions?[3]

That's a good question. Our motto through the writing in Italy, and then in Bolinas more so, was Jim saying at the end of the day, or around midnight or whenever we stopped writing, he would come to my room and say, "Reuter, stop – let's go out and see if there is life outside transactions!". And we used it in various contexts to make fun of other things. Technically speaking, I think the main reason for writing the book from Jim's perspective was of course, that he wanted to sum up everything he had learned about building distributed systems, building online systems, and without the notion of transactions it would be impossible. So, in that sense, transactions are a necessary elementary particle, if you will, in establishing such systems. If you don't have transactions, then all the architectural principles just fall apart.

We thought about adding a chapter or a subchapter somewhere on comparing transactional approaches to other approaches using non-transactional concepts, but we finally decided against it. Because, again, Jim in some points was very strict, and he said we only write about this stuff, we don't write about that stuff. From his perspective, non-transactional systems were just bad stuff, and he wouldn't mention it. What we tried to do of course was to make the point, from the technical perspective, that transactions facilitate a whole lot of architectural issues that would have to be solved in more tricky ways otherwise: transactional RPC and all these things. So, if you take the perspective of the time the book was written and from the background of the people who wrote the book, and the people whose work we were drawing on (it is not that we had invented these things): No, there wasn't life outside transactions, really.

But of course, transactions weren't all of life. It is like the basic processes enabling biological life; there is metabolism, there is replication, and some essential protocols that have to be implemented in order to make life viable. When you look at a cell, or when you look at a mammal in isolation, you will find, just by looking at them, very little in common between the two. But if you look at the internal plumbing at the cell level, you will find a lot in common. From our perspective, transactions establish this elementary set of protocols, plumbing, that make very complex distributed systems possible.

What about those people who think eventual consistency is good enough?

That's one of the approaches that you have to take when simple transactions don't help you anymore. But then of course the question is, who rings the bell: Now is the time to talk about consistency, or not. Of course, if you use proposals that have been described in the context of more generalized synchronization protocols, absolute serializability and Escrow-type of locking,

---

[3] Andreas Reuter. Is there life outside transactions?: writing the transaction processing book. SIGMOD Record 37(2): 54-58 (2008).

etc., this is all the flavor of eventual consistency. Let things get a little bit out of hand along the way, as long as correctness is maintained at the end, its good. The question is, what is the end? Who calls the shots? If things go on continuously, and there is no point when you can say, "well, stop", then it doesn't really help you. You have to make sure that the differences don't accumulate. Once you have to introduce these points of control artificially, well, then you have something like intervals of consistency checking and that's one aspect that transactions have, originally. But, it always comes back to the ACID properties, which in the original transaction model were captured by one system construct. When you wanted to do more complex things, it turns out that you couldn't get all of them. You could get, maybe, correctness, but then there goes isolation and atomicity. Or you could atomicity but had to sacrifice consistency, etc.

Where will transaction processing be in 20 years?

Well, if the people who say they don't play any role anymore are correct, then well we will see transactions in the computer history museum, in some niche, in some corner. Otherwise, I think transactions will still be there, very far down in the system, in the form of transactional memories, which have a good chance of becoming part of many future processor architectures, I assume, especially in multicore environments. Because multicore means you have to raise the level of parallelism to levels that have been unheard of and unmanageable, at least so far, without putting too much burden in terms of controlling parallelism in the program. And so you need system artifacts in between – between the application program and the actual hardware – that will take care of all the hairy parallelism issues. And again, some of the ideas that transactions have incorporated will play a role there: making things atomic; making things automatically recoverable within a certain distance of history, by invalidating some cache lines or through some other mechanism – I can't go into any detail here. But I think that's where transactions will play a role. Reliable message delivery, that's where transactions, or transactional protocols and the idea of atomicity will still be needed. If they will still be called transactions, I have no idea – most likely not. People will not be talking about that.

Reuter, stop! Let's go out and see if there is life outside transactions!

In computer science, are we always reinventing the wheel?

Yes.

Is it a bad thing?

No, it's not a bad thing. I mean, when you go from a triangular wheel to a quadratic wheel, this is progress. Then you take this idea further, and things get smoother, and smoother, and smoother. It's fine. Another way of saying we are reinventing the wheel is that we, in many fields, go around in circles: At any point in time, there is a focus on a specific set of questions, mostly dictated by current technology, for example, the balance between memory and I/O, or between CPU and memory. This is what is given, basically. There's no such thing like a completely virtual machine; in the end it runs on real hardware. So people start trying to use the hardware as best that they can, and that influences architectural decisions all the way up. Once these problems are solved, and we have a reasonably good performance platform, you build applications on top of that, which take into account what the software underneath does, and so on. After some time, when everything works, the hardware is changing, or has been changing all the time, which could be ignored for a while, but now it becomes obvious that you have to do things differently if you want to exploit the resources optimally. Or your performance requirements have changed, you want to do more with your system, you want to do it faster.

Ten years ago, data mining or data analytics or OLAP were offline activities. You created a warehouse from your online database, and then you had these wonderful OLAP tools scanning the warehouse and telling you things. Now people want to do it in real time, on the real online data. They want no warehouse reflecting the world two hours ago. They want to analyze the online data, and not just the data as they are stored, but they want to analyze the data as they are stored plus the incoming data streams, and compare this to other data streams that other people have and they have access to. So that shifts the set of requirements, and in order to make that possible, you have to look at the hardware again. Given the new hardware that we've got now this requires significant changes of the software; we have to reinvent some of the basic algorithms. The search structures that we have come to love and optimize in databases, do they make sense in really huge memories? Or what about the balance between I/O and buffers in databases, just to stay in this field.

When we wrote the book, there was this famous 5 or 10-minute rule. If somebody referenced a piece of data once in every 5 minutes, then keep it in main memory otherwise, put it on disk. Now we are talking about 3 hours for the same criterion, and it's still going up. This has huge implications on the way you maintain your data structures. In the old days, query optimization was looking at a query, but right now, especially when you look at these data streams, you have to appreciate that at the same time, you have to process hundreds, thousands, millions of queries, roughly on the same data set. Now, what does optimization mean? Well, it is clear what optimization means, but how do you do it, and how do you support it?. And that's what I mean by going around in circles. You start rebuilding the mechanics, the basic engines of the system, and then of course you go off to offer new services, with new constraints for the applications, and get to build applications in different manners – more real-time things, more this, and more that. And I think we have been through the cycle at least 3 times in the history as far as I can see, and right now we are in the start of a new cycle, because people are really building new engines for databases. Not just adding new object types and new blades or whatever: they are rebuilding the engine.

Because of the multicore aspect?

Mostly because of the multicore aspect, because of the huge increase in memory size, and because of the shifting balance between memory bandwidth and I/O bandwidth.

Is the community paying enough attention to those things?

Oh, yes, absolutely. That's what I really take away from this conference, that there are enough people in the technical community who are just happy to be plumbers. They don't need all of this fancy stuff, you know, analyzing trends in social communities, or discussing Hadoop versus parallel SQL. Instead, they really focus on the low level technical issues, and ask: How can we exploit the new hardware that we get, and make databases run optimally on that hardware? Think, for example, of big applications on the database at petabyte sizes, which in parallel to maintaining this huge set of data, look at a large number of streams with high bandwidth. We will talk about systems with millions of cores. These systems have to be self-optimizing, self-configuring, self-everything, because no human will be able to "understand" what they are doing. And there are no tools right now to help humans in setting up such systems. Of course, can't do it all themselves, this would be wishful thinking, but getting from systems that we know and can manage today, to those systems, will take a major leap in a couple of dimensions. People are excited about this, and at the same time, people are coming with interesting requirements from new applications that they want to do with their data. Whether the result is called a database or something else, again, what's in a name? But the technology is new. This seems to be going on with considerable vigor. So, it's good.

> ***These [million-core] systems have to be self-optimizing, self-configuring, self-everything, because no human will be able to understand what they are doing.***

Why don't database people use databases?

I am not sure if this is still true. It depends on when, in the course of a day, you look at what a database person does. When database people do their research, or do their development work, they probably use a database, because that is what they are working on, that's what they are paid for. The question came up repeatedly, in the so-called old days, when databases where, as a matter of fact, fairly hard to install, hard to manage, and expensive, compared to what they could do for people who didn't have just a large number of routine jobs, like the classical OLTP type. OLTP was characterized by a small number of repetitive applications, called transactions. But for the stuff that people normally do when they are alone in the office, doing ad hoc things, that's what databases typically weren't very good at, and other tools, like Excel where much better at. Since people normally, for their own purposes, maintain some addresses, some references, they don't need big stores, so that's not a big deal for them. They just use whatever PC tools are available. So it is a simple trade-off in terms of productivity. But this doesn't say that database people don't practice what they preach, it is just the fact that for a long time databases were built for large routine applications and individuals don't handle large routine applications. They do very many different ad hoc things, and so that was not the design space for classical databases.

What was it like when you started a new university?

This was a delusional thing. I say this in hindsight, of course. You have to be very naive and very optimistic, but naive I think is the better word, in order to do this. The reason was that I was vice president of the State University in Stuttgart for a couple of years. I had come to understand a large number of things, many more than I ever cared to learn about, of why state universities in Germany had so many difficulties, problems in running up to their real potential. It had to do with their fiscal structure, with their internal legal structure, that their administrations were owned by the state government rather than independent bodies, etc. And at that time (it was the mid-90's) there was a climate in Germany when people all over the community started to recognize this and said we need to do something, the university system has to be reformed. And I said well, if we focus on some efficient and interesting areas, why not set up something like a model shop, for certain aspects of academic teaching, and try it out, and see how it works. And of course, you couldn't do this in the public sector, because these decision processes just take forever. And there was a chance to set up a privately funded university.

Even the state government was mildly supportive of this, which is important, because they had to okay it – all the degree granting things are a monopoly of the states in the German system. The minister in charge of this at the time publicly said, "we support these private universities because we hope they will be a thorn in the flesh of the public universities" - which didn't help us a lot. It is a strange thing for the minister to say, but he did. And I really had the hope that this would create some kind of more open debate on which way to go, on evaluating different concepts, etc.

But, anyhow, we started the thing[4], which was the first German university to teach entirely in English, and the first German university to use the bachelor/master degree system. The public universities had a completely different one at that time. And, well, I think I never worked so much in my life as I did during that time. I taught classes, about four times the teaching load I had at the state university - simply because we were short of people. We really started from scratch, and initially there was almost nothing. We defined programs in IT and in the business area. And then the curriculum had to be implemented. For everything where we didn't have people, which happened from time to time, I had to jump in and do it myself, besides the other duties. We came off the ground fairly well. There were some evaluations, some rankings, which came out quite favorably.

But, what dominated the whole project throughout was the political debate. This university was always a token in political combat between different sides. You can ignore this for a while, but over time you will be confronted with so many problems that have nothing to do with what's happening in the institution, with the way the institution is set up, with the plan you have. It's just political ambushing at different levels and at different times that I felt I spent just an excessive amount of time besides the real work I did for this institution by just coping with the side effects of political intrigue. State universities are all for free, there was no tuition, and we charged, I don't know what the exact price was, $15,000 a year. And so you had to convince people: why should they pay such a high amount of money, when something like this is for free next door? You have to be much better, you have to be more efficient, etc. And you need sponsors, corporate sponsors. Those corporate sponsors are completely willing to support such projects, but once the institution gets into some kind of public conflict, or some kind of public debate with negative overtones, than corporations are very quick to withdraw. They say, "Oh, we don't want to be associated with this". "Sorry, no hard feelings, we know you do the right thing, but it's just the way it appears on the outside".

[4] In 1998 Andreas Reuter co-founded the International University in Germany.

Just preventing this from happening, and getting new sponsors, and some sponsors who withdrew, it was so tiring that, in 2004, I decided to just leave in the interest of my own health. I learned a lot from this experience. And I don't want to miss it, but I understand now that when I started this, I was just very naive. But on the other hand, there was very little experience to draw on. The good thing is that we and some other private universities that were started roughly at the same time - one to three years later - they had the net effect of changing the structure of the public universities, so they now have international programs, they're all moving, with a lot of opposition, to a professional master's system instead of the old system. So this model shop character I think was something that we really achieved, but the price was a bit high.

> ***Is it a bad thing [to reinvent the wheel]? No! When you go from a triangular wheel to a quadratic wheel, this is* progress.**

Sounds like you were a thorn in the flesh at that time.

Right, and I had a lot of thorns in my own flesh.

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

I think I don't have some overarching perspective that could cover everything. From my own experience, I should say it's very important that you don't lose contact, in no phase of your career, with real problems, with real applications. Databases is not a field that is dominated by theory. You need the theory, but in the end, we need to build systems that help solve application problems. And so, in regular intervals – I don't say every day, I don't say every year – in regular intervals, you should get your fingers dirty by building something, by implementing stuff. Then see how the system behaves and derive new questions and new problems from that. It should be a continuous circle, between building things, evaluating things, and thinking about better solutions, better ways of doing stuff. I think it was Jim Gray again who said this in an unmistakable, concise way: first build the code, and then write the paper.

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

Well, I think the perspective I am taking is from my current positions in the research institutes I am heading. One of our main purposes is to provide infrastructure, IT, database infrastructure, computing infrastructure, for people who are doing theoretical research in life sciences, in physics, and in other fields. In none of these fields, I have a good enough understanding of what their data management problems really are, so if I had extra time, I would pick one of those fields, let's say systems biology, go into it a little deeper than I can at the moment, and just understand what kind of data management problems, data analysis problems do they have, and how could that be supported by the technology that we have.

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

If there were an institution where I could buy or rent or whatever, more patience, I would go there and try to get some. Because, I think I am a little bit too quick in abandoning a problem.

But in computer science, patience doesn't really pay off in general, does it?

I think it does.

It changes so fast.

Well, I don't agree completely. Many of the concepts that we are talking about are fairly stable. It requires patience to really filter out the stable, the longer lasting insights and concepts from all the barrage of new words that are coming out. Of course we are changing words to denote the same thing, and again, we are not the only discipline doing this. I know others, which are about as bad. And we are changing words faster than we change concepts. Of course, one can very easily be mistaken by a new word meaning something entirely new. In many cases, it is just all the same old stuff in a new wrapping. To get to the bottom of these things, and understanding whether they are conceptual issues as opposed to whether they are just wrapping issues – that requires a certain amount of patience. I admit having been taken away by new words, saying, oh "I shouldn't miss this", and then after some time, I say "oh #$%@, this isn't so new after all", and then I jump to something else again. It is part of my personality, I'm afraid I can't change it, because you know at my age, I can't change anything at all. Had I been more patient, consistent, I might have ended elsewhere. But, that's the way it is.

Thank you very much for talking with me today.

Thank you, Marianne. It was a real pleasure.

Data Centric Research at The University of Queensland

Xiaofang Zhou Shazia Sadiq
School of Information Technology and Electrical Engineering
The University of Queensland, Australia
{zxf,shazia}@uq.edu.au

1. INTRODUCTION

The University of Queensland (UQ) is in the top 100 universities worldwide measured through a number of major rankings. In 2012, the University had more than 46,000 students including 11,000 international students from 134 nations. It has one of Australia's largest PhD enrolments, with more than 12,600 postgraduate students. The University's outstanding 200,000-plus alumni include a Nobel laureate, the CEO of a Fortune 500 company, an Academy Award winner, and leaders in government, law, science, public service and the arts.

UQ hosts a cross-disciplinary group of researchers working on data centric approaches, spanning the engineering, business and science faculties. The group's strength lies in its diversity and ability to cater for the entire data pipeline. Members of the group have, and continue to contribute to all aspects of data-centric research including data modelling, indexing, querying, cleansing, analyzing, visualizing, as well as use and adoption over a number of application areas. Currently the group has 26 full time staff and over 50 PhD students. The group was awarded the highest ranking under the Australian Research Council (ARC) Excellence of Research in Australia scheme in 2012 (www.arc.gov.au/era/).

From 2005-2010, we managed the ARC Research Network in Enterprise Information Infrastructure (eii.edu.au). The aims of EII were to provide focus for research exchange via networking and collaboration and to improve the quality, impact and visibility of Australian ICT research. This was a unique initiative of the ARC, funded by the ARC in conjunction with financial support provided by 21 Australian and international institutions. Over the course of 5 years, and led by Xiaofang Zhou as convenor of the network, EII made a profound impact on the Australian research community, University sector and IT industry in terms of research collaboration, research training (especially through the annual PhD school), and research quality.

Over the years, we have also organized, or contributed to the organization of a number of major conferences in Brisbane including VLDB 1990, WWW 1998, WISE 2004, BPM 2007, DASFAA 2009, ACIS 2010 and ICDE 2013. ACM Multimedia will also be hosted by us in 2015.

In this article we first present a short chronology outlining the major topics and achievements of the group in the last three decades. We then present details on a selected number of current topics that we would like to share our viewpoint on, in terms of our focus, contributions and way forward.

1.1 Research Chronology

Late 1980s - Early 1990s:

Conceptual Modelling: One of the earliest strengths of the group was in the area of conceptual modelling[29]. In the late eighties and early nineties, Sjir Nijssen, considered as a founder of verbalization in computer science, developed a fact-based modelling methodology for information analysis, known as NIAM, subsequently extended as ORM (Object Role Modelling) together with Terry Halpin. This work was instrumental in promoting the benefits of conceptual modelling to many businesses, and over two decades later, the ORM approach continues to be used widely in teaching and research (www.ormfoundation.org).

Early - Mid 1990s:

Theory of Representation: At the same time as the influential works on conceptual modelling were being developed, UQ researchers were developing their theory of representation[42], also known as BWW model. The models that made up this theory provided a basis for work over the ensuing 25 years to date that has led to increased understanding of conceptual modelling, and the strengths and weaknesses of languages and tools that have been commonly used in conceptual modelling.

Mid - Late1990s:

Distributed Systems: During the nineties, the group was heavily involved with the Distributed Systems Technology Centre (DSTC), a centre of excellence that gained an international reputation for one of the most influential IT research organizations in Australia. During this period, the group's focus was on advanced distributed systems including data integration, semantics and advanced database applications.

Late 1990s - Mid 2000s:

Workflow Technology: Led by Maria Orlowska and the DSTC team, a number of industry strength applications in the area of workflow systems were developed, including FlowMake, a business process modelling and verification tool [36]; Chameleon, a high performance workflow engine for non-traditional workflow applications; and FlexeL, an eLearning platform based on workflow technology. Members of the group were one of the first to introduce a declarative constraint based approach to business process modelling [34], which influenced a shift in research on flexible business processes.

Bioinformatics: From 2004 - 2010, we were part of ARC Centre of Excellence in Bioinformatics, and worked on data management issues in bioinformatics, including 3D protein structure search and fuzzy biomedical image search [18, 50].

Current:

Spatial and Multimedia Databases: Led by Xiaofang Zhou and Heng Tao Shen, the group has made wide ranging contributions to the topics of spatial and multimedia databases. These are further detailed in sections 2 and 3 below.

Data Quality Management: In section 4, we outline our work on data quality management, led by Shazia Sadiq and Marta Indulska, highlighting specifically the initiatives for improved cross-fertilization between research, industry and user groups in the area of data quality management.

Information Use and Adoption: Led by researchers from the UQ business school, Peter Green and Andrew Burton-Jones, a range of topics relating to theory of representation, effective use of information and IT governance and compliance have been studied. The contributions of this research which is predominantly empirical in nature, are further detailed in section 5.

Application Driven Research: Although all aspects of our theoretical, experimental and empirical research are positioned in the context of meaningful use cases with strong links to application scenarios and often developed together with industry partners and domain experts, the eResearch group at UQ, led by Jane Hunter is an Australian hub of activities on data centric applications in science and humanities. In section 6, we will present a summary listing of some of the current cross-disciplinary research projects.

Next we will briefly present some of our recent work on these current topics.

2. TRAJECTORY COMPUTING

Location and time are two ubiquitous attributes of useful data. With the increasing ease of capturing spatiotemporal data, it becomes an important aspect of modern data management to efficiently and effectively manage spatial and spatiotemporal data, and to discover rich information about spatial and spatiotemporal patterns and trends.

Our research in this area is rooted in the development and deployment of an early spatial database management system SIRODBMS [3], especially for a set of efficient parallel spatial join algorithms [53] and spatial data warehousing algorithms [54]. When spatial DBMS evolved from a highly specialized tool used by well-trained professionals to a necessity in our daily life used by everyday people from the Internet and via mobile devices, our focus moved to provide light weight access to cater for a wide range of users and devices with variable resolutions and to minimize IO costs and communication bandwidth demand. To that end we proposed multi-resolution map data access methods and spatial query processing algorithms in both the Euclidean space and some constrained spaces such as road-networks and terrain surfaces [45, 9, 43].

With the prevalence of GPS devices and smartphones, an increasing amount of spatiotemporal data has become available. New applications such as environmental resource management, intelligent transport systems, urban planning, fleet management, location-based services and marketing, and location-based social networks can benefit from analyzing large amounts of trajectory data. We developed data mining algorithms to identify object movement patterns for future location predication [22] and object "convoy" identification [23], various trajectory search algorithms [7], location-based search combined with text keywords and multimedia tags [51, 27], and handling of low-sampling trajectory data that comes with location uncertainty [52].

In a recent work [41], we discovered that raw trajectory data cannot always be used for similarity comparison directly for a large range of proposed trajectory similarity measures, especially for low sampling data. Based on that we introduced trajectory calibration as a fundamental operation for similarity-based trajectory analytics. This work further opens new opportunities in trajectory computing, such as incorporating the temporal information in calibration, novel indexing and query processing methods with calibrated trajectories.

One new direction for our group on trajectory computing is to develop new spatiotemporal database systems for real-time trajectory analytics, based on in-memory technologies and multi-core parallelism. That approach is expected to bring an order of magnitude processing performance improvement.

3. MULTIMEDIA SEARCH

Driven by the rapid advances in hardware and multimedia technologies, we have witnessed unprecedented interest in large-scale multimedia retrieval in many computer science areas, including database management, multimedia analysis, information retrieval, computer vision, machine learning, image processing and distributed computing. It emerges as a key technology for a broad range of critical application domains, including security, health management, environment monitoring and astronomy.

Lack of indexing support for Web-scale multimedia databases has been recognized as one of the bottlenecks for fast multimedia search. In the last few years, we have made significant contributions to effective and efficient multimedia search. To enable real-time content-based search over large-scale multimedia databases, we have proposed various novel indexing approaches, including optimal one-dimensional indexing method [38], dimensionality reduction [39], data co-reduction [16] and hashing [40]. In particular, we have developed the first real-time near-duplicate video retrieval system over millions of video clips by introducing compact video signatures and effective indexing structures [17]. The system has been demonstrated in several major conferences such as VLDB, ICDE, ACM Multimedia, and attracted interest from industry with three filed patents. Our recent survey paper summarizes the state of the art on near-duplicate video retrieval and sheds light on its future research trends [28].

Complementary to content-based search, semantic-based search provides a convenient way for users to find relevant multimedia data with textual keywords. However, one major limitation of the traditional multimedia tagging framework is that tags are only assigned or propagated at the global level of objects. We have further advanced existing methods by allocating tags at the local level, i.e., regions in images [47] and shots in videos [55], to provide finer semantic indexing for multimedia objects. Web-based image tagging tools have also been developed to harness the richness of Web data [46].

Using new cost-effective depth cameras, we have investigated effective feature extraction methods for improved human gesture recognition from motion data streams [49]. Human body gesture recognition has many valuable applications in video surveillance, patient monitoring, smart homes and entertainment.

Clearly, the emergence of big multimedia data has presented many new opportunities across different research disciplines. One future trend for multimedia search is to fuse heterogeneous data sources with multi-modal representations to greatly improve the quality of the results. Additionally, the wide availability of growing social multimedia data has inspired us to leverage crowd wisdom to reshape the landscape of multimedia search, by discovering and utilizing the synergy between multimedia data and social data. With the application shift from desktops to mobiles, another future trend is to propose mobile solutions for novel applications, where mobile context, user profiles, device configurations and user experiences all play a part in the search process. We envisage that our future work related to multimedia search will be mainly focussed on main memory indexing of hash codes generated from scalable learning methods on heterogeneous data, utilizing social knowledge, and developing mobile platforms for multimedia retrieval.

4. DATA QUALITY

Data quality can be broadly interpreted as the state of completeness, validity, consistency, timeliness and accuracy that makes data appropriate for a specific use. The issue of data quality is as old as data itself. However due to the proliferation of large scale data in every walk of life, it is now exposed at a much wider and critical level, increasing manifold the stakes involved for corporations, government agencies and individuals. Due to the changing nature of data management, traditional approaches and solutions to data quality control are challenged, and there is an evident need to incorporate data quality considerations into the whole data cycle.

The topic of data quality has been pervasive in several aspects of our work. These include data consistency, linkage and matching [8, 25], data uncertainty [52], data completeness [26], data diversity [24], and data freshness [37, 44]. Although historically issues of data integrity control have been addressed widely from the database research community, we identified an evident gap between research and practice [35], which has resulted in relative isolation between the organizational and computational aspects of data quality management and poor uptake of significant technical contributions from research. We initiated a number of initiatives for improved cross-fertilization between research and industry, which includes a Handbook of Data Quality [33], formation of a community of practice in data quality in Asia Pacific (apac.iaidq.org), as well as more locally establishing a data quality roundtable that currently has over 20 members from 7 different organizations.

We observe that current data quality management tools and practices cannot cope with the phenomenon of big data, where an overwhelmingly large amount of diverse and dynamic data needs to be processed and used for real-time decision-making. There are two intrinsic issues that make flexible, automated, and efficient support of data quality management highly challenging: Firstly, the quality of data is fundamentally perceived through the fitness for use lens, wherein the application (user and purpose) perspective makes generic data quality control very difficult. Secondly, the prevalence of large, multi-source data sets makes it very difficult for data owners and users to have a comprehensive understanding of the data characteristics and properties including data quality levels and targets. In light of these challenges, our current research aims at improving quality awareness [48] in data management systems, especially developing extensions in data management systems that link together data quality specification, measurement, improvement and tracking capabilities.

5. EMPIRICAL RESEARCH

Current members of the group, led by Peter Green, at the UQ Business School have continued to contribute to the body of knowledge on conceptual modelling as well as making significant contributions to various empirical aspects of data management. The theory of representation has been used and extended into many areas including business process modelling [12, 31], software maintenance [15], activity-based costing in management costing systems [32], and process interoperability languages [13]. Most recently, Andrew Burton-Jones has returned to his alma mater from Georgia State University and then UBC, Vancouver. He has continued the strong research into conceptual modelling by theorizing how representation is at the heart of effective use of information systems by users in organizations [6].

As organizations generate, collect and store continually increasing amounts of data, the drastic and ongoing increase has spawned the need not only for new research on data processing and retrieval techniques and data quality management approaches, but also for auditing of information systems use practices, IT governance approaches and structures to ensure accountability for the use of systems and data, and compliance management to ensure processes that use and generate the data are in line with legal and regulatory requirements.

Our research in the broad domain of compliance management has identified key challenges for the Information Systems community [1], and addressed the specific problem of lack of shared understanding in the compliance management environment within an organization by developing a validated ontology of compliance management, the first of its kind, to facilitate the required common conceptual model of the phenomenon. The Compliance Management Ontology (CoMOn) [2] has been validated with several organizations and has received significant attention from industry with respect to use in training and professional development of compliance professionals.

In the area of IT governance which pertains to the establishment of structures and practices that indicate accountability for the use of data and systems in organizations and encourage desirable behaviour in the use of IT, our research includes a study of factors that facilitate top management to increase IT governance absorptive capacity [5] and identification of IT governance mechanisms in practice that lead to effective IT governance in organizations [4]. While governance helps to motivate appropriate IT usage, it is a periodic systems audit that proves to regulatory bodies that inappropriate usage is not taking place. Thus, in our research we have also studied information systems audit issues [11] as well as the changing role of the IS audit.

6. E-RESEARCH

We are involved in a large number of data centric applications in science and humanities, and are investigating innovative approaches to the management, analysis and visualization services of large scale data collections, to accelerate scientific discovery. Members of the group are working on cross-disciplinary research projects across a broad range of topics including environmental informatics, digital humanities and biomedical sciences. The common aim across all of these projects is to expedite research outcomes through the sharing, integration and analysis of open access data, using Semantic Web approaches. Brief summaries of some of the current projects are presented below:

Online Environmental Report Cards Project provides GIS interfaces for online interactive access to integrated environmental monitoring data and environmental models, for both South East Queensland and the Great Barrier Reef, enabling decision support and adaptive resource management [19].

Twentieth Century in Paint Project is for painting conservators and links information about artists techniques, paint chemistry, characterization data, structured knowledge extracted from publications and experiments on paint degradation, to answer queries associated with paint conservation [21].

Semantic Annotation Services Project includes exploring new tools to support the annotation of 3D digital representations of cultural heritage artefacts, using the recently developed Open Annotation (OA) data model [20].

OzTrack Project develops a national repository for the storage, analysis, sharing and exchange of GPS-based animal tracking data. It combines a GIS interface (built on OGC standards) with R statistical analysis tools to calculate home ranges and understand interactions within and between species in the wild, and to assist with conservation management planning for endangered species [10].

Skeletome Project builds a community-driven knowledge base for the Skeletal Dysplasia domain. The research focus is the development of improved methods for extracting phenotype-disorder associations from medical literature, to improve the diagnosis and treatment of skeletal dysplasias [30, 14].

7. OUTLOOK

Data centric research is increasingly become a key element in addressing some of the most wicked problems of our times. We have been particularly fortunate to have the benefit of a team with diverse interests, expertise and networks. Our groups diversity has been instrumental in assisting us with the ability to tackle the whole data pipeline. As we move forward, we see that such multi-disciplinary teams have a growing importance, due to the need for database research and technical contributions to be positioned within a clear and evident business or scientific need on one hand, and a deployment blueprint on the other, in order to realize impact of our contributions on technological, economic and societal level.

We believe that such a creation of research impact requires genuine partnerships with user communities so as to bring true big (and dirty) datasets into the realm of database research community [1]. To this end, we advocate improved mechanisms for data sharing. This also requires closer relationship with hardware configurations in database research to fully capitalize on new in-memory and multi-core computing infrastructures, so as to improve experimental repeatability and generate a stronger value chain.

8. ADDITIONAL AUTHORS

Additional authors: Heng Tao Shen, Xue Li, Mohamed A. Sharaf, Zi Huang, Kai Zheng, Jane Hunter, Peter Green, and Marta Indulska.

9. REFERENCES

[1] N. S. Abdullah, S. W. Sadiq, and M. Indulska. Emerging challenges in information systems research for regulatory compliance management. In *CAiSE*, 2010.

[2] N. S. Abdullah, S. W. Sadiq, and M. Indulska. A compliance management ontology: Developing shared understanding through models. In *CAiSE*, 2012.

[3] D. J. Abel. SIRO-DBMS: a database tool-kit for geographical information systems. *International Journal of Geographical Information System*, 3(2):103–116, 1989.

[4] S. Ali and P. Green. Effective information technology (IT) governance mechanisms: An IT outsourcing perspective. *Information Systems Frontiers*, 14(2):179–193, 2012.

[5] S. Ali, P. Green, and A. Robb. Measuring top management's IT governance knowledge absorptive capacity. *Journal of Information Systems*, 27(1):137–155, 2013.

[6] A. Burton-Jones and C. Grange. From use to effective use: a representation theory perspective. *Information Systems Research*, pages 1–27, 2012.

[7] Z. Chen, H. T. Shen, X. Zhou, Y. Zheng, and X. Xie. Searching trajectories by locations: an efficiency study. In *SIGMOD Conference*, 2010.

[8] K. Deng, L. Wang, X. Zhou, S. W. Sadiq, and G. P. C. Fung. Active duplicate detection. In *DASFAA (1)*, 2010.

[9] K. Deng, X. Zhou, H. T. Shen, K. Xu, and X. Lin. Surface k-nn query processing. In *ICDE*, 2006.

[10] L. Gao, H. A. Campbell, O. R. Bidder, and J. Hunter. A web-based semantic tagging and activity recognition system for species' accelerometry data. *Ecological Informatics*, 13:47–56, 2013.

[11] P. Green, P. Best, M. Indulska, and T. Rowlands. Information systems audit and control issues for enterprise management systems: Qualitative evidence. *Australian Accounting Review*, 15(3 Supplement 37):67–77, 2005.

[12] P. F. Green and M. Rosemann. Integrated process modeling: An ontological evaluation. *Inf. Syst.*, 25(2):73–87, 2000.

[1] http://wp.sigmod.org/?p=786

[13] P. F. Green, M. Rosemann, M. Indulska, and C. Manning. Candidate interoperability standards: An ontological overlap analysis. *Data Knowl. Eng.*, 62(2):274–291, 2007.

[14] T. Groza, J. Hunter, and A. Zankl. Mining skeletal phenotype descriptions from scientific literature. *PloS one*, 8(2):e55656, 2013.

[15] J. Heales. Factors affecting information system volatility. In *ICIS*, 2000.

[16] Z. Huang, H. T. Shen, J. Liu, and X. Zhou. Effective data co-reduction for multimedia similarity search. In *SIGMOD Conference*, 2011.

[17] Z. Huang, H. T. Shen, J. Shao, X. Zhou, and B. Cui. Bounded coordinate system indexing for real-time video clip search. *ACM Trans. Inf. Syst.*, 27(3), 2009.

[18] Z. Huang, X. Zhou, D. Song, and P. Bruza. Dimensionality reduction in patch-signature based protein structure matching. In *ADC*, 2006.

[19] J. Hunter, P. Becker, A. Alabri, C. van Ingen, and E. G. Abal. Using ontologies to relate resource management actions to environmental monitoring data in south east queensland. *IJAEIS*, 2(1):1–19, 2011.

[20] J. Hunter and A. Gerber. Harvesting community annotations on 3d models of museum artefacts to enhance knowledge, discovery and re-use. *Journal of Cultural Heritage*, 11(1):81–90, 2010.

[21] J. Hunter and S. Odat. Building a semantic knowledge-base for painting conservators. In *eScience*, 2011.

[22] H. Jeung, Q. Liu, H. T. Shen, and X. Zhou. A hybrid prediction model for moving objects. In *ICDE*, 2008.

[23] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen. Discovery of convoys in trajectory databases. *PVLDB*, 1(1):1068–1080, 2008.

[24] H. A. Khan, M. Drosou, and M. A. Sharaf. Dos: An efficient scheme for the diversification of multiple search results. In *SSDBM*, 2013.

[25] H. Köhler, X. Zhou, S. W. Sadiq, Y. Shu, and K. L. Taylor. Sampling dirty data for matching attributes. In *SIGMOD Conference*, 2010.

[26] Z. Li, M. A. Sharaf, L. Sitbon, S. W. Sadiq, M. Indulska, and X. Zhou. Webput: Efficient web-based data imputation. In *WISE*, 2012.

[27] J. Liu, Z. Huang, L. Chen, H. T. Shen, and Z. Yan. Discovering areas of interest with geo-tagged images and check-ins. In *ACM Multimedia*, 2012.

[28] J. Liu, Z. Huang, H. C. H. T. Shen, C.-W. Ngo, and W. Wang. Near-duplicate video retrieval: Current research and future trends. *ACM Computing Surveys*, 45(4), 2013.

[29] G. M. Nijssen and T. A. Halpin. *Conceptual schema and relational database design - a fact oriented approach*. Prentice Hall, 1989.

[30] R. Paul, T. Groza, J. Hunter, and A. Zankl. Decision support methods for finding phenotypedisorder associations in the bone dysplasia domain. *PloS one*, 7(11):e50614, 2012.

[31] J. Recker, M. Rosemann, P. F. Green, and M. Indulska. Do ontological deficiencies in modeling grammars matter? *MIS Quarterly*, 35(1):57–79, 2011.

[32] M. Rosemann and P. F. Green. Integrating multi-perspective views into ontological analysis. In *ICIS*, 2000.

[33] S. Sadiq. *Handbook of Data Quality: Research and Practice*. Springer-Verlag New York Incorporated, 2013.

[34] S. W. Sadiq, M. E. Orlowska, and W. Sadiq. Specification and validation of process constraints for flexible workflows. *Inf. Syst.*, 30(5):349–378, 2005.

[35] S. W. Sadiq, N. K. Yeganeh, and M. Indulska. 20 years of data quality research: Themes, trends and synergies. In *ADC*, 2011.

[36] W. Sadiq and M. E. Orlowska. Analyzing process models using graph reduction techniques. *Inf. Syst.*, 25(2):117–134, 2000.

[37] M. A. Sharaf, C. Xu, and X. Zhou. Finding the silver lining for data freshness on the cloud. In *CloudDb*, 2012.

[38] H. T. Shen, B. C. Ooi, and X. Zhou. Towards effective indexing for very large video sequence database. In *SIGMOD Conference*, 2005.

[39] H. T. Shen, X. Zhou, and A. Zhou. An adaptive and dynamic dimensionality reduction method for high-dimensional indexing. *VLDB J.*, 16(2):219–234, 2007.

[40] J. Song, Y. Yang, Y. Yang, and Z. H. H. T. Shen. Inter-media hashing for large-scale retrieval from heterogenous data sources. In *SIGMOD Conference*, 2013.

[41] H. Su, K. Zheng, H. Wang, J. Huang, and X. Zhou. Calibrating trajectory data for similarity-based analysis. In *SIGMOD Conference*, 2013.

[42] Y. Wand and R. Weber. On the ontological expressiveness of information systems analysis and design grammars. *Information Systems Journal*, 3(4):217–237, 1993.

[43] K. Xie, K. Deng, S. Shang, X. Zhou, and K. Zheng. Finding alternative shortest paths in spatial networks. *ACM Trans. Database Syst.*, 37(4):29, 2012.

[44] C. Xu, M. A. Sharaf, M. Zhou, A. Zhou, and X. Zhou. Adaptive query scheduling in key-value data stores. In *DASFAA (1)*, 2013.

[45] K. Xu, X. Zhou, and X. Lin. Direct mesh: a multiresolution approach to terrain visualization. In *ICDE*, 2004.

[46] Y. Yang, F. Wu, F. Nie, H. T. Shen, Y. Zhuang, and A. G. Hauptmann. Web and personal image annotation by mining label correlation with relaxed visual graph embedding. *IEEE Transactions on Image Processing*, 21(3):1339–1351, 2012.

[47] Y. Yang, Y. Yang, Z. Huang, H. T. Shen, and F. Nie. Tag localization with spatial correlations and joint group sparsity. In *CVPR*, 2011.

[48] N. K. Yeganeh, S. W. Sadiq, K. Deng, and X. Zhou. Data quality aware queries in collaborative information systems. In *APWeb/WAIM*, 2009.

[49] X. Zhao, X. Li, C. Pang, X. Zhu, and Q. Z. Sheng. Online human gesture recognition from motion data streams. In *ACM Multimedia*, 2013.

[50] K. Zheng, P. C. Fung, and X. Zhou. K-nearest neighbor search for fuzzy objects. In *SIGMOD Conference*, 2010.

[51] K. Zheng, S. Shang, N. J. Yuan, and Y. Yang. Towards efficient search for activity trajectories. In *ICDE*, 2013.

[52] K. Zheng, Y. Zheng, X. Xie, and X. Zhou. Reducing uncertainty of low-sampling-rate trajectories. In *ICDE*, 2012.

[53] X. Zhou, D. J. Abel, and D. Truffet. Data partitioning for parallel spatial join processing. *GeoInformatica*, 2(2):175–204, 1998.

[54] X. Zhou, D. Truffet, and J. Han. Efficient polygon amalgamation methods for spatial olap and spatial data mining. In *SSD*, 1999.

[55] X. Zhu, Z. Huang, J. Cui, and H. T. Shen. Video-to-shot tag propagation by graph sparse group lasso. *IEEE Transactions on Multimedia*, 15(3):633–646, 2013.

Report on the 6th International Workshop on Business Intelligence for the Real Time Enterprise

Malu Castellanos
HP Labs
1501 Page Mill Road
Palo Alto, CA 94304, USA
malu.castellanos@hp.com

Umeshwar Dayal*
HP Labs
1501 Page Mill Road
Palo Alto, CA 94304, USA
umeshwar.dayal@hp.com

Elke Rundensteiner
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609, USA
rundenst@cs.wpi.edu

Overview

The 6th International Workshop on Business Intelligence for the Real Time Enterprise (BIRTE 2012) was held on August 27, 2012 in conjunction with the VLDB 2012 Conference in Istanbul, Turkey. With a good number of attendees, from various acedemic and industrial sectors, the workshop was an extremely attractive forum to discuss topics related to the emerging field of making business intelligence more real-time. The workshop included one keynote, three invited talks, and five presentations of peer-reviewed research papers in this context. After the official opening of the workshop, Donovan Schneider (Salesforce.com, USA) delivered an excellent and engaging keynote about real-time reporting practices at Salesforce, which was followed by three sessions with talks focusing on "Advanced BI Applications and Vivification", "ETL and Query Processing in Real-Time BI" and "Challenges and Advances in Analytics Platforms".

Keynote

The keynote talk speaker was Donovan Schneider who is a Principal Architect at Salesforce.com developing real-time analytics for their multi-tenant sotware-as-a-service platform. n this talk "Real-Time Reporting at Salesforce", Donovan explained the issues with running millions of real-time reports every day on Salesforce multi-tenant cloud platform and presented the solution including the architecture, query processing, optimization strategies and other considerations to get real-time reporting at such scale.

Advanced BI Applications and Vivification

The first morning session on advanced BI Applications featured the talk titled "The Vivification Problem in Real-Time Business Intelligence" by Patricia Arocena which delved on the vivication problem and sketched approaches towards a solution aimed at integrating the IT world, data management technologies and the Business world. Ulrike Fischer's (TU Dresden and Aalborg University et al.) talk on "Real-Time Business Intelligence in the MIRABEL Smart Grid System" presented the challenges in real time BI and the data management solutions for a distributed energy grid system in the MIRABEL project. The second paper on "Data Mining in Life Sciences Using In-Memory DBMS" by Joos-Hendrik Boese et al. from SAP described how HANA meets some inherent requirements of data mining in life sciences and presented a use case.

*Current Affiliation Hitachi America

ETL and Query Processing in Real-Time BI

The afternoon session started with the invited talk "On-Demand ETL Architecture for Right-Time BI" by Florian Waas from EMC /Greenplum. He outlined some of the key differences practiced in Greenplum for ETL queries in the view of real-time query processing. The work by Yagiz Kargin et al. (CWI) on Instant-On Scientific Data Warehouses for Data Intensive Research addressed the inefficiencies in ETL processes commonly used in scientific data analysis by proposing Lazy ETL. The third talk "Query Processing of Pre-Partitioned Data Using Sandwich Operators" by Stephan Baumann et al. (TU Ilmenau, CWI) introduced the Sandwich Operators approach to exploit pre-sorting or pre-grouping from clustered storage schemes to lower memory consumption and execution time. This session also featured the invited talk by Raymond Ng from the University of British Columbia entitled "Towards multi-modal extraction and summarization of conversations" which described the challenges in summarizing email and other conversational data and discussed open problems to make it a reality and be conducted in real-time..

Challenges and Advances in Analytics Platforms

The final session opened with the invited talk of Meichun Hsu from HP Labs on "Live Analytics Service Platform". In her presentation Hsu highlighted the challenges, key features and integration of data management and analytics services in HP Lab's Real-time Analytics Platform. Konstantinos Zoumpatianos (U. Trento) followed by presenting the paper "Strategic Management for Real-Time Business Intelligence" where the authors introduced the idea of developing a Business Intelligence stack on top of data warehouses to enable continuous evaluation of analytical queries. In the talk "Pricing Approaches for Data Markets", Alexander Loser (TU Berlin) highlighted the key challenges with regard to pricing strategies for trading raw data, associated analytical services, and analytic results on cloud-based platforms in different market situations. He posed several interesting research problems for the business intelligence community.

Discussion

Overall, there was clearly a wide spectrum of related technologies to be explored and much interest in this area. The presentations of the speakers were accompanied by lively discussions. The issues discussed centered around several main questions: (1) What are some of the newly emerging applications of real-time Business Intelligence and how they can be integrated with existing business models and IT infrastructure. (2) What are the main challenges in BI real-time query processing over data warehouses and various approaches of optimizing ETL queries in a real-time context. (3) What are the challenges in developing real-time analytics platforms for Business Intelligence and scientific applications and (4) how to extend existing database technology in order to exploit the advantages of the exisiting data mining techniques?

Summary

In summary, the BIRTE 2012 workshop was one more extremely successful event in the BIRTE series attracting a peak audience of over 50 attendees. The slides of the presentations and a preliminary version of the corresponding papers can be found on the BIRTE's 2012 webpage: http://birte2012.cs.wpi.edu/. The final version of the papers are available as post-proceedings in an LNBIP volume edited by Springer. We are looking forward to BIRTE 2013 to be held in conjunction with VLDB 2013 in Italy.

Databases, Information Retrieval and Knowledge Management: Exploring Paths and Crossing Bridges

Mouna Kacimi
Free University
of Bozen-Bolzano, Italy
Mouna.Kacimi@unibz.it

Fabian M. Suchanek
Max Planck Institute
for Informatics, Germany
suchanek@mpi-inf.mpg.de

Aparna Varde
Montclair State University
Montclair, NJ, USA
vardea@mail.montclair.edu

ABSTRACT

The International Conference on Information Retrieval and Knowledge Management (CIKM) brings together three avenues of data-oriented research, namely, Database Management, Information Retrieval and Knowledge Management. The confluence of these avenues becomes evident also in the PhD theses of doctoral students: Stream processing makes use of knowledge representation techniques, linked data is emerging as a research topic that bridges information retrieval and knowledge representation, and new forms of querying draw on techniques from both information retrieval and databases. In this paper, we survey new PhD theses at the meeting point of the three research avenues. Our survey is based on the 5th PhD workshop at the ACM CIKM conference. The topics include themes as diverse as link prediction, source code querying, and video stream processing.

1. INTRODUCTION

PIKM 2012 was the 5th workshop in a series of PhD workshops collocated with the International Conference on Information and Knowledge Management (CIKM). The goal of this workshop series is twofold. First, it gives doctoral students an opportunity to present their work on a global stage. This allows them to receive feedback from reviewers, from fellow students and from the general CIKM audience. Second, we believe that the CIKM research community, too, benefits from such a workshop: PhD theses are the grassroots of research. They point out new research avenues and provide fresh viewpoints from the researchers of tomorrow. Previous workshops have been held in 2007 [4], 2008 [3], 2010 [2], and 2011 [1].

This year's PIKM workshop attracted 19 submissions, of which 10 were accepted as full papers and 4 further submissions as poster papers. The PIKM has always had a poster session in order to promote interaction among student presenters and attendees. The topics of this year's papers included business process modeling, recommendation using linked data, collaborative Web search, link prediction, and querying scientific data. The PIKM honors the best submission to the workshop with a "Best Paper Award". The winner of the award is determined by the best reviews, after discussion in the program committee and the steering committee. This year's award went to the paper "When Big Data Leads to Lost Data" by V.M. Megler and David Maier.

Our experience with the preceding workshops allows us to carefully select our reviewers for the PIKM. We invite the reviewers who have provided reviews of outstanding quality for the last workshops. Thereby we can steadily raise the calibre of the reviews, and make them more helpful for the students. To further motivate reviewers to deliver high quality reviews, this year's PIKM introduced the "Best Reviewer Award" for the reviewer with the most helpful reviews. With this prize, we want to encourage reviewers to provide rich in-depth reviews, and reward the most engaged reviewer for his effort. This year, the honor went to Gerard de Melo (ICSC Berkeley, California, USA). The runners-up for the best reviewer award were Georgiana Ifrim (4C, Cork, Ireland), and Pierre Bourhis (Oxford University, UK). Also, we sincerely appreciate the efforts of our entire team of 21 reviewers comprising experts from academia and industry across the globe.

In the last years, the PIKM has always been able to attract a senior keynote speaker. This year's speaker was Ingmar Weber from Yahoo Research Barcelona/Spain. In a highly entertaining talk, Dr. Weber gave "Advice for Young Jedi Knights and PhD Students". The talk looked at the task of pursuing a PhD with both a humorous and a serious eye. The serious eye gave practical advice on how to choose an advisor and a thesis topic, and looked at different successful approaches to doing research and to publishing. The humorous eye looked at whether education can contribute to personal happiness, the positive influence of chocolate on studying behavior and the effect of too much time spent on Facebook. Dr. Weber deserves our sincere thanks for making this talk a true highlight of the workshop.

The submissions to this PhD workshop spanned different areas of research. Even though much of the work presented at PIKM crosses multiple CIKM tracks, we broadly group the papers by their main area corresponding to each CIKM track, pointing out links to the other areas. The main areas are Database Systems, Knowledge Management, and Information Retrieval.

2. DATABASE SYSTEMS

Research in database systems explores advances in technology that supports data management, as well as the new problems that arise from such advances. The work leading to PIKM's best paper award ("When Big Data Leads to Lost Data" by V.M. Megler et al.) falls precisely in this category. It explores the popular area of cloud computing while addressing the issue of not finding exactly relevant information amidst the huge volume of data on the cloud. The paper proposes an approach to find relevant information from big scientific datasets by exploiting text retrieval techniques (such as interactive searching by asynchronous scanning for feature extraction) in the context of numeric data. This approach also uses information from a metadata catalog for feature extraction, scoring and ranking to enhance performance. The paper provides experimentation to convince the audience that the proposed approach is a feasible solution to the lost data problem. The primary area of this work is databases, but it bridges to the IR track due to its emphasis on techniques popular in text search. It briefly touches upon mining aspects by its interactive querying and ranking processes.

Other work in the database track of PIKM includes external source code querying, unified scientific data processing and data stream event detection. "Querying External Source Code Files of Programs Connecting to a Relational Database" by Carlos Garcia-Alvarado et al. deals with the problem of querying multiple external program files that reference metadata, which requires asserting impacts of changes between the programs and databases. Since existing solutions to this problem are found to be highly complex, the paper proposes algorithms analogous to keyword searches to analyze references between external programs and database schemas such that dependencies are preserved and changes in a source are reflected in another. "SciQL: A Query Language for Unified Scientific Data Processing and Management" by Javad Chamanara et al. addresses the realm of scientific data management by proposing a query language called SciQL to perform scientific data processing and management in a unified manner. It gives scientists the means to express queries on data refinement, transformation, visualization and other tasks in a common format irrespective of physical data sources. Thus, scientists can use one language to interact with various sources such as plain text, Excel spreadsheets, RDBMS and MapReduce systems, making it easier to manage their research. "Towards an Advanced System for Real-Time Event Detection in High Volume Data Streams" by Andreas Weiler et al., presented as a poster, focuses on streaming data, and, more specifically, on detecting events in real-time in data streams of high volumes. To achieve this, the authors consider HPC (high performance computing) coupled with continuous querying. They process queries over the data streams and fast queries over the respective historical data to detect events, classify them and rank them.

3. KNOWLEDGE MANAGEMENT

Knowledge management approaches deal with the extraction of information from data. They cover two main avenues of research, knowledge discovery and knowledge representation. Knowledge discovery is used in many real world applications to understand the meaning of data and predict its implicit properties. For example, "Exploring and Analyzing Documents with Online Analytical Processing" by Grzegorz Drzadzewski et al. helps users grasp the topics of a new document collection. To this end, the authors propose a system that explores and analyzes clusters of documents using Online Analytical Processing (OLAP), an effective strategy for extracting and analysing views of data. The approach provides efficient and accurate techniques for creating and aggregating clusters, finding representative documents, finding relationships between clusters, and determining their strength. Another example for making implicit information explicit is given in "Feature Selection for Link Prediction" by Ye Xu et al.. The authors propose a feature selection strategy for link prediction in networks. The proposed algorithm is based on the discriminative abilities and the correlations between pairs of features. The best features would maximize the total discriminability score while minimizing the total correlation scores. Link prediction is central to many real world networks such as social media platforms.

Knowledge representation approaches, in contrast, deal with capturing the different aspects of data. One of the most important aspects is dynamicity, because it challenges the accuracy of models learned from old data. Examples include stream data and dynamic business processes. "Is That Scene Dangerous? Transferring Knowledge Over a Video Stream" by Omar U Florez et al. addresses the problem of dynamicity in the context of video labeling. The authors propose an unsupervised framework based on topic modeling to represent the different activities of a video scene, and an algorithm to label such activities in previously unseen movies. This method can, e.g., detect and predict dangerous behavior of a car at a given place. "Multilevel Business Process Modeling: Motivation, Approach, Design Issues and Applications" by Christoph Schütz et al. addresses the dynamic aspects of data in process modeling. The paper extends multilevel modeling approaches that capture the dependencies of processes belonging to different hierarchical levels within a company. The approach models life cycles using UML state machines and associates a life cycle model to each class for each level in the hierarchy.

4. INFORMATION RETRIEVAL

Information Retrieval is concerned with finding relevant Web documents for a user query. Several papers at the workshop investigate new dimensions of the field. " Intent-Aware Temporal Query Modeling for Keyword Suggestion" by Fredrik Johansson et al. proposes to analyze queries not just per se, but in their temporal dimension and interaction. In winter, e.g, a search for "coughing" is more likely to indicate that the user is interested in remedies against a cold. In summer, the query is more likely to indicate interest in an allergy. Queries can also be studied in their temporal relation to each other. Queries about a vacation by car, e.g., are likely to be followed by queries about parking lots. "Assessing the Relationship between Context, User Preferences, and Content in Search Behavior" by Hanna Knäusl et al. puts forward the idea of studying what parts of a Web page a user is interested in. The authors present a user study that analyzes what parts of a Web page users read after having issued a query. The hypothesis is that we can aid the user by proactively selecting the interesting part of the Web page.

With the rise of the Web 2.0, Web search, and interaction on the Web in general, becomes more social. In this spirit, "iTop: Interaction Based Topic Centric Community Discovery on Twitter" by Denzil Correa et al. analyzes how user communities develop on Twitter. The authors analyzed thematic and social communities, based on the hashtags and @-tags that users employ in their tweets. This yields a social graph of users and their interests. "Towards a More Efficient and Personalized Advertisement Content in On-line Social Networks" by Patxi Galán-García et al. presents a more commercially oriented work. The authors aim to find relevant advertisements for social network users, based on their chats. This approach has the advantage of customizing the ads to the real-time interests of users. The task is challenging because instant messages are usually in colloquial, often even faulty natural language. "Recommendation Using Linked Data" by Rouzbeh Meymandpour et al. presents a research work that spans the Web 2.0 with the newly emerging Semantic Web. The goal is to extend the realm of social recommendation to the Semantic Web. Given that some users like some entities, how can we find other entities they like? The authors develop similarity metrics that use both content features and graph features for this purpose. "Search Tactics as Means of Examining Search Processes in Collaborative Exploratory Web Search" by Zhen Yue et al. explores the setting of collaborative search in general, where multiple users work together to solve an information retrieval task. The authors conduct user experiments on the search platform CollabSearch, and use Hidden Markov Models to model how users find information in this environment.

5. CONCLUSION

The PIKM 2012 workshop showed us a wide variety of doctoral dissertation topics in the areas of databases, knowledge management and information retrieval. In PIKM 2011, the upcoming hot topic was research on linked data and social networks. While these topics were still prevalent at the PIKM 2012, the workshop focused more on Web data management and mining, especially considering the use of the cloud. Among cloud-based areas, the favorite domain was still social networks. In particular, various areas pertinent to search seemed to attract attention, both within social networks and otherwise, e.g., temporal query modeling, personalized advertisements, user preferences and video streams. In fact, enhancing search results by exploring technological advances database management, mining, and IR appeared to attract PhD students from all CIKM tracks. “Search” thus seemed to be the bridge that connected almost all the PIKM 2012 papers along each path.

6. REFERENCES

[1] Anisoara Nica, Fabian M. Suchanek, Aparna S. Varde: New Research Directions in Knowledge Discovery and Allied Spheres. Submitted to ACM SIGKDD Explorations 2012

[2] Anisoara Nica, Fabian M. Suchanek, Aparna S. Varde: Emerging multidisciplinary research across database management systems. SIGMOD Record 39(3): 33-36, 2010

[3] Aparna S. Varde: Challenging research issues in data mining, databases and information retrieval. In SIGKDD Explorations 11(1): 49-52, 2009

[4] Aparna Varde and Jian Pei: Advances in Information and Knowledge Management. In SIGIR Forum Journal, 42(1): 29-35, 2008