SIGMOD Officers, Committees, and Awardees

Chair
Juliana Freire
Computer Science & Engineering
New York University
Brooklyn, New York
USA
+1 646 997 4128
juliana.freire <at> nyu.edu

Vice-Chair
Ihab Francis Ilyas
Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario
CANADA
+1 519 888 4567 ext. 33145
ilyas <at> uwaterloo.ca

Secretary/Treasurer
Fatma Ozcan
IBM Research
Almaden Research Center
San Jose, California
USA
+1 408 927 2737
fozcan <at> us.ibm.com

SIGMOD Executive Committee:
Juliana Freire (Chair), Ihab Francis Ilyas (Vice-Chair), Fatma Ozcan (Treasurer), K. Selçuk Candan, Yanlei Diao, Curtis Dyreson, Christian S. Jensen, Donald Kossmann, and Dan Suciu.

Advisory Board:
Yannis Ioannidis (Chair), Phil Bernstein, Surajit Chaudhuri, Rakesh Agrawal, Joe Hellerstein, Mike Franklin, Laura Haas, Renee Miller, John Wilkes, Chris Olsten, AnHai Doan, Tamer Özsu, Gerhard Weikum, Stefano Ceri, Beng Chin Ooi, Timos Sellis, Sunita Sarawagi, Stratos Idreos, Tim Kraska

SIGMOD Information Director:
Curtis Dyreson, Utah State University

Associate Information Directors:
Huiping Cao, Manfred Jeusfeld, Asterios Katsifodimos, Georgia Koutrika, Wim Martens

SIGMOD Record Editor-in-Chief:
Yanlei Diao, University of Massachusetts Amherst

SIGMOD Record Associate Editors:
Vanessa Braganholo, Marco Brambilla, Chee Yong Chan, Rada Chirkova, Zachary Ives, Anastasios Kementsietsidis, Jeffrey Naughton, Frank Neven, Olga Papaemmanouil, Aditya Parameswaran, Alkis Simitsis, Wang-Chiew Tan, Pinar Tözün, Marianne Winslett, and Jun Yang

SIGMOD Conference Coordinator:
K. Selçuk Candan, Arizona State University

PODS Executive Committee:
Dan Suciu (Chair), Tova Milo, Diego Calvanse, Wang-Chiew Tan, Rick Hull, Floris Geerts

Sister Society Liaisons:
Raghu Ramakhrishnan (SIGKDD), Yannis Ioannidis (EDBT Endowment), Christian Jensen (IEEE TKDE).

Awards Committee:
Martin Kersten (Chair), Surajit Chadhuri, David DeWitt, Sunita Sarawagi, Mike Carey

Jim Gray Doctoral Dissertation Award Committee:
Ioana Manolescu (co-Chair), Lucian Popa (co-Chair), Peter Bailis, Michael Cafarella, Feifei Li, Qiong Luo, Felix Naumann, Pinar Tozun

SIGMOD Systems Award Committee:
Mike Stonebraker (Chair), Make Cafarella, Mike Carey, Yanlei Diao, Paul Larson

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. Recipients of the award are the following:

Michael Stonebraker (1992)
David DeWitt (1995)
Serge Abiteboul (1998)
Rudolf Bayer (2001)
Ronald Fagin (2004)
Jennifer Widom (2007)
Umeshwar Dayal (2010)
Stefano Ceri (2013)
Gerhard Weikum (2016)
Jim Gray (1993)
C. Mohan (1996)
Hector Garcia-Molina (1999)
Patricia Selinger (2002)
Michael Carey (2005)
Moshe Y. Vardi (2008)
Surajit Chaudhuri (2011)
Martin Kersten (2014)
Goetz Graefe (2017)
Philip Bernstein (1994)
David Maier (1997)
Rakesh Agrawal (2000)
Don Chamberlin (2003)
Jeffrey D. Ullman (2006)
Masaru Kitsuregawa (2009)
Bruce Lindsay (2012)
Laura Haas (2015)
Raghu Ramakrishnan (2018)

SIGMOD Systems Award

For technical contributions that have had significant impact on the theory or practice of large-scale data management systems.

Michael Stonebraker and Lawrence Rowe (2015) Martin Kersten (2016) Richard Hipp (2017)
Jeff Hammerbacher, Ashish Thusoo, Joydeep Sen Sarma; Christopher Olston, Benjamin Reed, Utkarsh Srivastava (2018)

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)
Jeffrey Ullman (1996)
Raghu Ramakrishnan (1999)
Daniel Rosenkrantz (2001)
Surajit Chaudhuri (2004)
Hans-Jörg Schek (2007)
David Lomet (2010)
H.V. Jagadish (2013)
Samuel Madden (2016)
Gio Wiederhold (1995)
Avi Silberschatz (1997)
Michael Carey (2000)
Richard Snodgrass (2002)
Hongjun Lu (2005)
Klaus R. Dittrich (2008)
Gerhard Weikum (2011)
Kyu-Young Whang (2014)
Yannis E. Ioannidis (2017)
Yahiko Kambayashi (1995)
Won Kim (1998)
Laura Haas (2000)
Michael Ley (2003)
Tamer Özsu (2006)
Beng Chin Ooi (2009)
Marianne Winslett (2012)
Curtis Dyreson (2015)
Z. Meral Özsoyoğlu (2018)

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. *Honorable Mentions*: Marcelo Arenas and Yanlei Diao.
- **2007** *Winner*: Boon Thau Loo. *Honorable Mentions*: Xifeng Yan and Martin Theobald.
- **2008** *Winner*: Ariel Fuxman. *Honorable Mentions*: Cong Yu and Nilesh Dalvi.
- **2009** *Winner*: Daniel Abadi. *Honorable Mentions*: Bee-Chung Chen and Ashwin Machanavajjhala.
- **2010** *Winner:* Christopher Ré. *Honorable Mentions*: Soumyadeb Mitra and Fabian Suchanek.
- **2011** *Winner*: Stratos Idreos. *Honorable Mentions*: Todd Green and Karl Schnaitterz.
- **2012** *Winner*: Ryan Johnson. *Honorable Mention*: Bogdan Alexe.
- **2013** *Winner*: Sudipto Das, *Honorable Mention*: Herodotos Herodotou and Wenchao Zhou.
- **2014** *Winners*: Aditya Parameswaran and Andy Pavlo.
- **2015** *Winner*: Alexander Thomson. *Honorable Mentions*: Marina Drosou and Karthik Ramachandra
- **2016** *Winner*: Paris Koutris. *Honorable Mentions*: Pinar Tozun and Alvin Cheung
- **2017** *Winne*r: Peter Bailis. *Honorable Mention*: Immanuel Trummer
- **2018** *Winne*r: Viktor Leis. *Honorable Mention*: Luis Galárraga and Yongjoo Park

A complete list of all SIGMOD Awards is available at: **https://sigmod.org/sigmod-awards/**

[Last updated : June 30, 2018]

Editor's Notes

Welcome to the September 2018 issue of the ACM SIGMOD Record!

This issue starts with the Database Principles column featuring an article on data provenance by Buneman and Tan. The article starts with a brief survey of existing work on data provenance that was largely motivated by curated databases. It then looks at potential applications of provenance in a set of new applications, including data citation, machine learning, social media, blockchain technology, and privacy. The article is of particular interest to the reader as it describes the areas in which data provenance is finding applications and is opening up new lines of research.

The Vision column features an article by Shay et al. on database access control. It presents a vision and description for query control, a paradigm for database access control where individual queries are examined before being executed and are either allowed or denied by a pre-defined policy. This paradigm stands in contrast to traditional view-based database access control, which requires the enforcer to view the query, the records, or both, and hence is difficult to apply when the enforcer is not allowed to view database contents or the query itself, e.g., in privacy-preserving encrypted databases. This article further presents a reference implementation and discusses promising future applications of query control.

The Distinguished Profiles column includes two articles. The first article features Timos Sellis, Professor at the Royal Melbourne Institute of Technology, previously, the National Technical University of Athens, and the University of Maryland. Timos is an ACM Fellow and an IEEE Fellow. In this interview, Timos talks about his work on R+ trees, which won a VLDB 10-year Paper Award, and more generally, multi-dimensional indexing in recent applications. He also discusses the major accomplishments in his 20-year career in Greece, gives advice for fledging and mid-career database researchers, and expresses his desire to build more systems. The second article features Peter Bailis, who won the 2017 ACM SIGMOD Jim Gray Dissertation Award for his thesis entitled "Coordination Avoidance in Distributed Databases," under the supervision of Joseph Hellerstein, Ion Stoica, and Ali Ghodsi at the University of California, Berkeley. Peter is now a professor at Stanford University.

The Reports Column features two articles. The first article presents a report on the NSF BIGDATA PI meeting: In March 2017, PIs and co-PIs funded through the NSF BIGDATA program were brought together along with selected industry and government invitees to discuss current research, identify current challenges, discuss promising future directions, foster new collaborations, and share accomplishments. The breakout sessions were directed to discuss problems and available data sets in five application domains: policy, health, education, economy & finance, and environment & energy. The article summarizes the thoughts on promising big data research in these five applications domains, as well as the needs to promote interdisciplinary training and produce high quality data sets to broaden the impact of big data across different applications areas. The second article reports on the 2017 Dagstuhl Seminar on Big Stream Processing. Stream processing can generate insights from big data in real time as it is being produced. The article reports findings from the seminar, focusing on applications, systems, and languages of big stream processing.

Finally, the issue closes with two announcements, call for nomination for the ACM PODS 2019 Alberto O. Mendelzon Test-of-Test Award and call for papers for ICDT 2020.

On behalf of the SIGMOD Record Editorial board, I hope that you enjoy reading the September 2018 issue of the SIGMOD Record!

Your submissions to the SIGMOD Record are welcome via the submission site:
http://sigmod.hosting.acm.org/record

Prior to submission, please read the Editorial Policy on the SIGMOD Record's website:
https://sigmodrecord.org

Yanlei Diao

September 2018

Past SIGMOD Record Editors:

Ioana Manolescu (2009-2013)
Ling Liu (2000–2004)
Arie Segev (1989–1995)
Thomas J. Cook (1981–1983)
Daniel O'Connell (1971–1973)
Alexandros Labrinidis (2007–2009)
Michael Franklin (1996–2000)
Margaret H. Dunham (1986–1988)
Douglas S. Kerr (1976-1978)
Harrison R. Morse (1969)
Mario Nascimento (2005–2007)
Jennifer Widom (1995–1996)
Jon D. Clark (1984–1985)
Randall Rustin (1974-1975)

Data Provenance: What next?

Peter Buneman
University of Edinburgh
opb@inf.ed.ad.uk

Wang-Chiew Tan
Megagon Labs
wangchiew@megagon.ai

ABSTRACT

Research into data provenance has been active for almost twenty years. What has it delivered and where will it go next? What practical impact has it had and what might it have? We provide speculative answers to these questions which may be somewhat biased by our initial motivation for studying the topic: the need for provenance information in curated databases. Such databases involve extensive human interaction with data; and we argue that the need continues in other forms of human interaction such as those that take place in social media.

1. INTRODUCTION

The purpose of this paper is neither to define provenance nor to provide a survey of the relevant research; there are numerous contributions to the literature that do this [19, 18, 25, 45, 49, 71, 28]. What we hope to do here is to draw out new strands of research and to indicate what we can do practically on the basis of what we now know about provenance. A good starting point is to state two generally held but conflicting observations: first that the more provenance information one can collect the better; second that it is impossible in practice to record all relevant provenance information.

Before narrowing our discussion to data provenance, let us look at these two observations. Imagining the impossible, suppose we could record all the provenance associated with some process or artefact (digital or otherwise). In what would be a massive amount of provenance data, would we be able to answer simple questions such as where some data was copied from or whether a process invoked a particular piece of software? Such questions may involve the querying of huge data sets and complex code. So simply recording total provenance, even if it were possible, still requires complex analysis. It requires us to extract simple explanations from a massive and complex structure of data and code. What are those explanations?

Being more realistic, in practice, we only have resources to record a limited amount of provenance information. So what do we record? We may – as is the case with physical artefacts – have some standard attributes (ownership, location etc.) but for computational processes and data can we predict what will be asked of provenance? Again we have to understand what kinds of explanation we are likely to want. Is there any minimal requirement on what we should record or how we should record it?

For most purposes, what we should record is application dependent. For example, if an application is targeting to answer the provenance of a sales figure reported in a company earnings report, then the data provenance that consists of the source data and the program or query that was used to generate the report are likely to be sufficient. However, sometimes, intermediate sales results from specific regions are combined with other data sources or results from other regions to generate the final report. In this case, to provide a comprehensive understanding of the sales figure in the company earnings report, it may also be necessary to track the programs that were used to generate the intermediate results.

In yet another type of application, it is important that the results are repeatable and reproducible. This is true of experiments in chemistry and physics where it is not only crucial that one can obtain the same results by rerunning the experiments but also by running it at other locations. Software repeatability and reproducibility have also become an important topic. To enable software reproducibility, it is typically necessary to document the hardware, the version of operating system and software libraries used, in addition to the program and data used to execute the experiment.

Insofar as data provenance is separable from other forms of provenance [8, 26] we focus on provenance that has to do with data: databases, data sets, file systems etc. In the Background section that follows, we summarise some of the important research contributions to data provenance, the motivation behind the research and the practical applications of it. In Section 3, we then look at possible applications of provenance in other areas of computer science.

2. BACKGROUND

As often happens, the first paper that addressed provenance [78] in databases had to be "rediscovered" several years after it was written. This paper introduced a form of tagging or annotation to describe the source of elements of a relational database, a form of *where*-provenance. Then in the later 1990s under various names the study started in earnest. In [79] a method based on inverse functions was used to visualize the *lineage* of data in scientific programming; and in [21, 22], in the context of data warehouses, an operational definition was given of what tuples in some source data "contributed to" a tuple in the output of a relational query – perhaps a form of *why*-provenance.

The authors' interest in the topic was sparked by their collaboration with biologists [44] involved in the Human Genome Project who were building *curated* databases of molecular sequence data. While a curated database resembles a data warehouse in the integration of existing databases, it also involves the manual correction and augmentation of the source data, and it cannot simply be characterized as a data warehouse or view. The biologists complained that they were losing track of where their data had come from. Now biologists are, by training, quite meticulous in keeping a record of what they have done – in this case what queries they have made or what manual additions or corrections were made, so in some sense the provenance of some small element of data – a number or a tuple – was available. However extracting the information they needed from a complex workflow of updates and queries on other databases was proving difficult. What they appeared to need was a simple explanation e.g.: "this number was entered by ... on ..." (where-provenance); or "this tuple was formed by joining tuple t_1 from R_1 to tuple t_2 from R_2" (how-provenance); or "this tuple is in the result because some other tuple was in the input" (why-provenance).

The example in Figure 1 illustrates the types of provenance described above. Consider a `Friend` relation, a `Profile` relation, and a query that joins the two relations to find pairs of friends with identical occupations (shown below).

```
select f.name1, f.name2
from   Friend f, Profile p1, Profile p2
where  f.name1 = p1.name and
       f.name2 = p2.name and
       p1.occupation = p2.occupation
```

The value "Carl" in the result is derived from the value "Carl" in the `Friend` relation. Hence, if there were an annotation on who entered that information and when, this information can propagate to the result according to where-provenance. The figure also illustrates that the how-provenance of the output tuple is the result of joining three tuples (Carl, Bob), (Bob, 30, analyst), and (Carl, 50, analyst) from the input. The why-provenance of the output consists of the same three source tuples. We will discuss the finer differences between the latter two types of provenance in Section 2.1. However, it is important to note that to fully explain why the output tuple exists, one must also account for the query. That is, these three tuples satisfy all the equality condition in the where clause of the query.

What we should again emphasize is that the purpose of data provenance is to extract relatively simple explanations for the existence of some piece of data from some complex workflow of data manipulation. In this sense it has a similar purpose to *program slicing* which seeks to provide an explanation for a part of the output of some complex program to a small part of the input – an explanation that is much simpler than the program itself.

Given that provenance is about explanation of some part of a complex process, it is natural to ask whether there is a unified language or model for describing provenance. PROV is a W3C recommendation for a model or ontology in which one can describe provenance [60, 58]. The intention is to produce a general model for any kind of provenance such as that associated with artefacts or some general computational process. At its core, PROV can be used to describe causal relationships between entities and activities, and in doing this can naturally describe the evaluation of a workflow. Because of this the term "workflow provenance" has sometimes been used to distinguish the ambit of PROV from that of data provenance. Worse, the terms "fine-grained" and "coarse-grained" have been used for this distinction. We do not believe these distinctions to be helpful. While it is straightforward to use PROV to describe some basic aspects of data provenance, we do not do so in this paper because it does not add much to the formalisms that have been found useful in the context of databases. Conversely, there is no reason why the formalisms developed for "fine-grained" data manipulation cannot be used in a larger context as we shall see in Section 3.1.

2.1 Annotation and provenance

From the beginning it was recognized that provenance should be expressed as a form of annotation. This was precisely the purpose of the Polygen model [78]: to annotate data elements with their provenance. However, there is a much more fundamental connection between the two topics, which again shows up in curated databases. Much of curated data is about *annotation* of existing data structures. Sometimes this annotation is expressed in the primary tables in a relational database, but sometimes important information about the currency or validity of some data is held in an auxiliary table or –

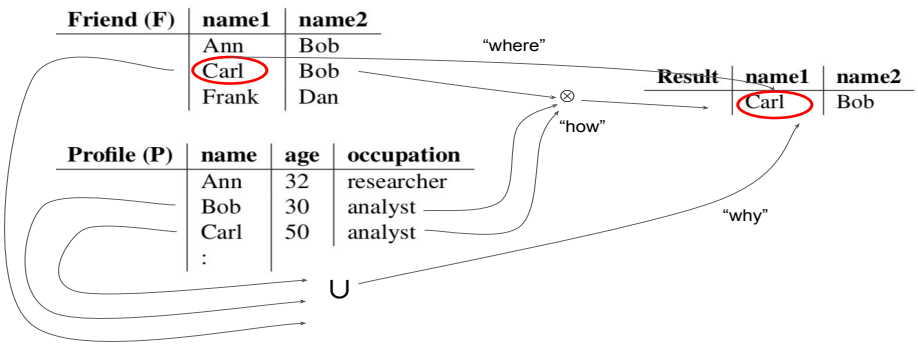


Figure 1: An illustration of where, how, and why-provenance.

in the case of semistructured data – some additional subtrees in a hierarchy or some additional edges in a graph representation. In fact, annotation data is semistructured by nature and often lives in some kind of auxiliary database. Queries over the "core" data often do not recognize this annotation, and this is one of the main sources of misleading or dirty data in both data warehouses and curated databases.

The basic question is then how do annotations propagate through database queries? This is a question that is closely related to data provenance and one that has driven much of the most interesting research on data provenance since its inception.

Annotation. The Polygen model [78] inspired the subsequent system DBNotes [6, 20] and other following work (e.g., [35, 10]). For each relational algebra operator, DBNotes provided a rule to propagate annotations based on where data is copied from. These rules are sensitive to the way the query is formulated: even though two queries are equivalent in the normal sense of always producing the same result the way the rules propagate annotations through the two queries may differ. Another propagation scheme that is agnostic to the way equivalent queries are formulated was also proposed to propagate the same annotations to the result.

That provenance may be sensitive to query formulation is seen in [10] which discusses update languages and uses a propagation scheme that is an extension of that in DBNotes. From a theoretical perspective, relational update languages, such as the update fragment of SQL, are often regarded as uninteresting because they are no more expressive than query languages. Consider the action of an SQL update: it replaces a version of the database with a new version. If we think of the old version as the input and the new version of the output, then that transformation from input to output can be expressed as a query in relational algebra. For example, Figure 2.1 shows a simple update query and an equivalent – in the sense that it produces the same output – query that doesn't involve updates. The backwards arrows show where all components of the table, values tuples and the table itself, come from. While the two queries produce the same answer, the provenance is different. The first update query only affects the where-provenance of the cell that the number "5" belongs to in the output. All the other components of the result table "come from" the corresponding component of the input table. On the other hand the more complicated query not only creates a new value 5, but a new tuple containing that value and a new table. In the figure the components that are created by the query are outlined in dotted red; the components that are copied are outlined on black.

The interesting observation is that if we take provenance into account, that is the query or update is a function that not only produces a result but also produces *where*-provenance associated with the values and tuples in a table, update languages become *more* expressive than query languages. Moreover [10] provides a completeness result: if the where-provenance can be expressed in (nested) relational algebra, then there is an update query in which the same where-provenance is implicit.

Semiring provenance The seminal work of [40] describes a formalism of data provenance that captures and extends previous formalisms such as why-provenance of [14] and lineage described in the Trio system [5].

A *commutative semiring* is a quintuple $(K, \mathbb{0}, \mathbb{1}, \oplus, \otimes)$. Here, K is a set of elements containing the distinguished elements $\mathbb{0}$ and $\mathbb{1}$, $\oplus$ and $\otimes$ are two binary operators that are both commutative and associative and $\mathbb{0}$ and $\mathbb{1}$ are the identities of $\oplus$ and $\otimes$ respectively. In addition, $\otimes$ is distributive over $\oplus$ and $\mathbb{0} \otimes t = t \otimes \mathbb{0} = \mathbb{0}$.

We assume that every tuple in the source database has a tuple identifier, and $I \subseteq K$ is the set of all such source tuple identifiers. The provenance of an element in an output table is expressed as a polynomial, an expression built up from $I, \mathbb{0}, \mathbb{1}, \oplus$ and $\otimes$. The provenance of an output tuple for each relational operator (select, project, cross product, union, rename) is obtained from the provenance polynomial of each input tuple. The simplest case is selection in which the provenance of an out-

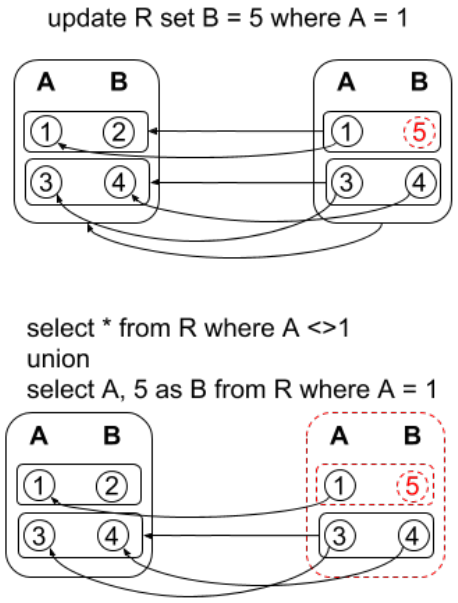


Figure 2: How updates affect provenance

put tuple is the same as the provenance of the (unique) corresponding input tuple. For join, suppose that $t_1 \in R_1$ and $t_2 \in R_2$ combine to produce $t \in R_1 \times R_2$. If e_1, e_2 are the provenance polynomials of t_1, t_2 then the polynomial for t is the polynomial $e_1 \otimes e_2$. For union, if $t \in R_1$ has provenance e_1 and the same tuple $t \in R_2$ has provenance e_2 then the provenance of t in $R_1 \cup R_2$ is the polynomial $e_1 \oplus e_2$. For a tuple t in the output of a projection, the provenance is the polynomial $e_1 \oplus \ldots \oplus e_n$ where $e_1, \ldots, e_n$ are the polynomials of the tuples in the input that "project onto" t. The polynomials attached to the tuples in the output of a query are built up inductively by these rules and others described in [40]. We can think of the polynomial as a description of *how* each tuple was constructed – by "joining" ($\otimes$) and "merging" ($\oplus$) other tuples.

The example below shows a query in SQL over the `Friend` relation of Figure 1. The query finds all people who share a friend with someone. In some sense the query is trivial because everyone shares a friend with themselves, however the provenance is interesting.

Query:

```
select f1.name1
from   Friend f1, Friend f2
where  f1.name2 = f2.name2
```

Assume that the tuples (Ann, Bob), (Carl, Bob), and (Frank, Dan) are annotated with i_1, i_2, and respectively, i_3. The result of the query is shown below alongside with annotations of the corresponding provenance polynomials and why-provenance.

name1	*provenance*	*why-provenance*
Ann	$i_1 \otimes i_1 \oplus i_1 \otimes i_2$	$\{\{i_1\}, \{i_1, i_2\}\}$
Carl	$i_2 \otimes i_2 \oplus i_1 \otimes i_2$	$\{\{i_2\}, \{i_1, i_2\}\}$
Frank	$i_3 \otimes i_3$	$\{\{i_3\}\}$

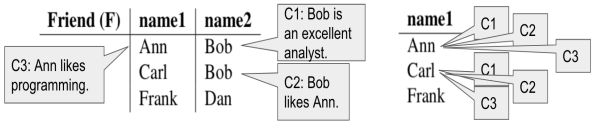
For example, the provenance polynomial for Ann is $i_1 \otimes i_1 \oplus i_1 \otimes i_2$ showing that i_1 and i_1 itself is one way of deriving the output tuple and another uses i_1 and i_2.

The remarkable property of these polynomials is that they unify many other generalizations of relational algebra such as bag semantics, C-tables and probabilistic databases. For bag semantics simply assign the "identifier" 1 to each tuple in the input and use the semiring $(\mathbb{N}, 0, 1, +, \times)$. The evaluation of the polynomial attached to a tuple gives the multiplicity of that tuple.

These polynomials also capture why-provenance with the semiring $(\text{Why}(K), \emptyset, \{\emptyset\}, \cup, \Cup)$, where $x \Cup y$ denotes the pairwise union of all sets in the two collections x and y. The evaluation of the provenance polynomial will give rise to the set of sets shown on the rightmost column above. Indeed, if we interpret each tuple identifier as a set of a singleton set, then the provenance polynomial of Ann $i_1 \otimes i_1 \oplus i_1 \otimes i_2$ is $\{\{i_1\}\} \otimes \{\{i_1\}\} \oplus \{\{i_1\}\} \otimes \{\{i_2\}\}$ which is $\{\{i_1\}\} \oplus \{\{i_1, i_2\}\}$ and gives rise to the why-provenance $\{\{i_1\}, \{i_1, i_2\}\}$.

Observe that the why-provenance describes what tuples in the source are *sufficient* for deriving the output according to the query. Indeed, either i_1 alone or both i_1 and i_2 are sufficient for generating the output tuple Ann according to the query. It is easy to see that the why-provenance can be derived from the provenance polynomial but not the other way round; the provenance polynomial is more informative.

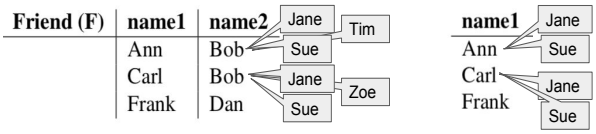
Semirings for propagating comments or beliefs can also be derived from the semiring framework. For example, the semiring $(\text{Lin}(K), \bot, \emptyset, \cup, \sqcup)$ which captures the lineage described in [22] can also be used to model how comments should propagate. Intuitively, the element $\bot$ denotes no lineage while $\emptyset$ denotes empty lineage, and $\sqcup$ is the usual union operator $\cup$ except that $\bot \sqcup X = X \sqcup \bot = \bot$.



The figure above exemplifies the "comments" semiring. The first source tuple (Ann, Bob) has two comments $C1$ and $C3$ and the second source tuple (Carl, Bob) has a single comment $C2$. Each of the first two tuples has all three comments in the result.

On the other hand, the belief of an output tuple can be captured with the following semiring $(\text{Belief}(K), \bot, \emptyset, \cup, \cap)$

which takes the intersection of the beliefs of the source tuples on a relational join.



Hence, {Jane, Sue} are the only remaining believers after the relational join operation.

Today, several database systems have been developed to support the propagation and querying of provenance such as Perm [37], LogicBlox, and Orchestra [39]. More recent implementations such as [2] provides a provenance-aware middleware implementation which can be used with different database back-ends and also supports *provenance for transactions*. Provenance support has also been implemented outside database systems. For example, in network provenance [82, 81], provenance is maintained and queryable at Internet-scale for diagnosing network errors in a distributed setting.

2.2 Provenance, repeatability, versioning

The ability to reproduce an experiment is essential to the credibility of the results of that experiment. The same is true for any kind of computational analysis or workflow that has been used to derive some data: the analysis must be repeatable. Whatever is needed to ensure repeatability is often regarded as provenance. The ability to record, reproduce, and query some computational process underlies "system-level" provenance [62], the provenance "challenge" [61] and at least one view of data citation [66]. Now almost all such analyses use some kind of external data source – this is obvious in the case of data citation, where the data source is the source being cited. The problem in all these cases is that the data source, and even its structure, is likely to evolve over time.

In curated databases we see a similar problem. When external data is incorporated, it is common to provide a link to source data as part of the provenance. While this requirement seems rather straightforward, there are at least two caveats to ensure a proper "implementation" that meets this requirement. First, the link should be a stable reference to the correct version of the database even if the database evolves. Most curated databases have a link which serves as a citation to its entire database. Web pages follow a similar organization where its URL refers to the latest version of the web page. When the database changes, the new database replaces the old database and hence, the link, which now refers to the new database, is no longer a valid reference for the previous database. The second issue is that the link is typically a coarse-grain approximation to a specific part of the database where the reference is typically intended for. While the HTML structure of web pages can be exploited to pinpoint to specific portions of the website, it is less obvious how specific portions of a database can be precisely referenced.

Data Versioning To ensure proper citation, some curated databases simply keep all past versions of the data. The onus is on the user to cite the correct (portions of the) version and to answer queries over multiple versions of data. For example, longitudinal queries such as "*what are all the changes in the last five versions?*", or "*when was this entry made?*" would be difficult to answer without going through each of the relevant database versions at least once.

Another approach, which is more economical on storage, stores only the changes (or deltas) between consecutive versions. However, the need to go through every relevant version for certain types of longitudinal queries such as "*return all versions where a particular entry exists*" is still unavoidable.

The archiving method of [13] strikes a balance between the two approaches described above; it keeps all database versions intact and economically by "merging", to the extent possible, different database versions together. Conceptually, every version is assumed to be in a hierarchical format such as in a JSON file format or XML. Every node has an associated set of intervals which captures the versions by which the node exists (the fat node method of persistent data structures [32]). Furthermore, if, as frequently happens, a node's interval set is identical to that of its parent one can save storage by taking the lack of an interval set to indicate that the interval set should be inherited. For biological databases such as those described in [13], it was observed that the dominant change is the addition of a node in the hierarchy, and that node modifications are relatively infrequent. This allows significant space savings, and a year's history of a database typically requires only a small percentage overhead in storage.

The main challenge with the archiving strategy is that it is not obvious how to match and merge nodes of a version into nodes of an existing database archive. In [13], a critical assumption is that there are keys for nodes in a hierarchical structure [12]. The keys are paths of labels or values and identify nodes in a version. Hence, they also help identify which nodes in the database archive to match and merge into. If a node in the version does not exist in the database archive, then it is a node that is new to the version and will be created as a new node in the database archive with a new interval. Conversely, if a node in the database archive has no corresponding node in the database version, then that node no longer exists and its interval of versions is terminated accordingly. Otherwise, the node is merged into the node in the database archive and its interval of versions is extended,

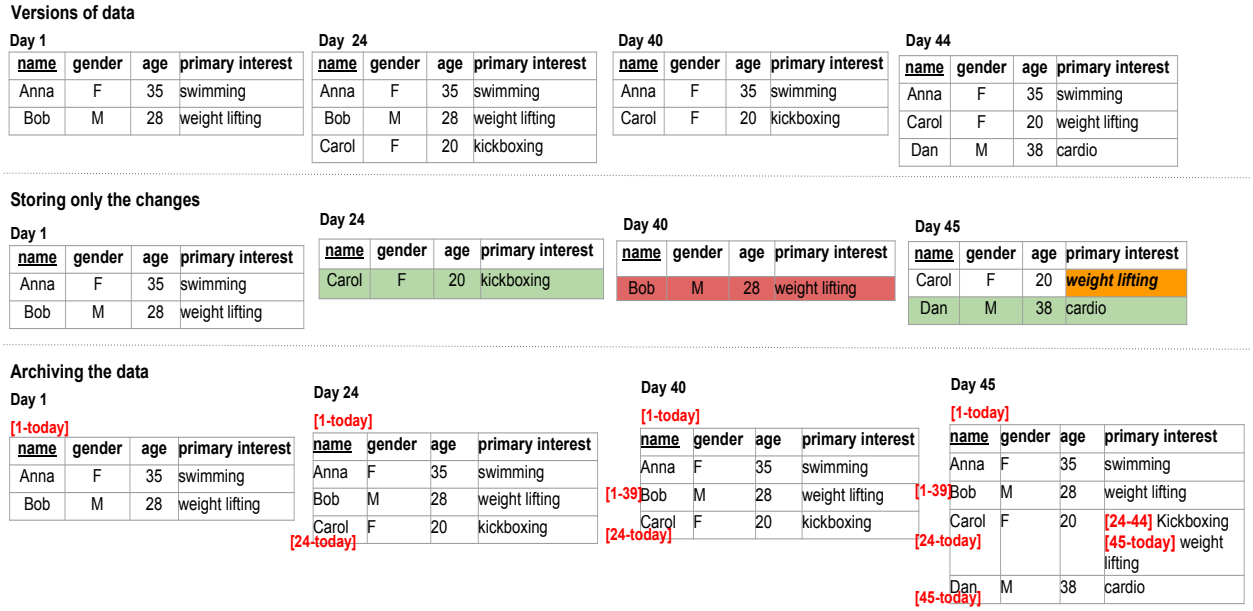


Figure 3: Three approaches to keeping all versions of data. Added tuples are shown in green, deleted tuples in red and modified values in orange.

denoting that the node continues to exist in the archive.

The assumption of a hierarchical key structure is reasonable for many curated databases and for scientific data formats [13]. Moreover the same technique can be applied to relational databases either by casting relations in a hierarchical format or performing archiving in the database engine by adding an interval to each tuple of each column of the relation schema to model the interval of versions. Figure 3 shows the three approaches in the relational context.

More recent work has directly tackled the problem of versioning relational databases [46, 57]. For example, [57] is a version-oriented storage engine designed from scratch to support versioning while [46] adds a versioning module on top of a relational database system. The latter architecture allows one to continue to exploit the advanced querying capabilities provided by a relational database system while adding efficient versioning capability to the system.

There is also a large body of work on temporal databases, which also support versioning as a special case. See [50] for a summary. In most versioning work, the notion of when a tuple has changed coincides with when the change is recorded in the database. Bi-temporal databases distinguishes these two types of time; transaction time and valid time. *Transaction time* denotes the time at which updates are applied to the database (hence, they can only "increase") which may be different from when the tuple is actually valid in the real world (valid time). In temporal databases, much of the effort is dedicated to managing and querying [52] these two notions of time efficiently.

Most versioning work and temporal databases has focused on recording data changes and there are relatively little that directly tackles the problem of managing both data and schema changes [68, 59, 23]. When the schema changes, can we easily query which data has changed (or not) across different versions? Can we effectively answer longitudinal queries across the versions? Can we seamlessly answer and even visualize the provenance of data that may consist of tuples from different versions, which may in turn be the result of another query on a database and so on?

3. WHAT IS NEXT?

So far, we have described, and asked questions about, existing work on data provenance that was largely motivated by curated databases. Next, we look at potential applications of provenance in data citation and in other areas of computer science, such as machine learning, social media, blockchain technology and privacy.

3.1 Provenance and data citation

Because so much knowledge is now disseminated through some form of database, there has been an increasing demand [34, 67] for these databases to be properly cited for the same reasons that we use citations for conventional publications. There is a problem in that data of interest is usually extracted from the database by some form of query. What citation should one associate with

the query or with the results of the query? There have been two general approaches to this. One is to treat citation and provenance as synonymous. To this end [66] have developed a system that carefully records what one might call the complete provenance associated with the evaluation of a database query. In particular they want to guarantee that the evaluation of the query is reproducible at a later stage. Critical to their approach is some form of database archiving of the kind we described in the previous section.

In contrast [11] have taken citation to mean the extraction of "snippets" of information, such as authorship, title, date etc. that one sees in a conventional citation. In fact [24] has a specification of the snippets that are appropriate for data. The problem is particularly interesting for curated databases which closely resemble, and often replace, conventional publications. In curated databases, there may be hundreds of "authors" who have contributed data. How does one extract the authors appropriate to the result of a specific query? [11] propose that by associating views with (groups of) authors, one can solve this problem using the well studied techniques of rewriting through views [30, 43, 54]. Conventional citations are, of course, a rather weak form of provenance, but techniques from the study of data provenance are nevertheless useful. [27] gives an interesting application of semiring provenance to generate and combine appropriate citations from views.

3.2 Provenance and machine learning

Machine learning and artificial intelligence have become an indispensable part in our daily lives. Machine learning methods are commonly used to automate everyday decision making in all aspects of our lives; from predicting email spams [42] to predicting crop yields [76], loan application, autonomous driving [17], disease identification and recommendation of medical treatments [51]. Even if machine learning models perform very well in practice, it is natural to question why a certain decision or prediction has been made, especially when decisions are critical. Explanations of a model's output can help build further trust in the system's performance and understand the foundations by which a decision has been made.

In machine learning research, the problem of deriving explanations of machine learning models is called *interpretability*. Somewhat ironically, there is less consensus on what the exact interpretation of *interpretability* [31, 64] should be. However, the reason for the lack of consensus should not be surprising. Like the situation in provenance, different users have different requirements of interpretability. For example, the requirements for interpretability so that a programmer can debug the model is quite different from interpretability of the predication of a crop yield. In the latter, one may only need to explain that it is because the estimated rainfall is high/low but in the former, one may need to understand how many rounds of simulation have been applied, the parameters and software modules used.

While some models lend themselves well to some form of interpretability (e.g., generalized additive models [15]), other models, especially neural networks, are opaque. An approach to overcome the opaque nature of neural networks is to learn another less opaque model based on the predictions of the original model.

The goals of data provenance and interpretability are clearly similar. Both seek to find explanations, at different levels of granularity, for the output of a program or a process. A major difference is that in database provenance, the program and process that have been considered by researchers are typically not opaque as in machine learning models.

A promising area of cross-fertilization between provenance and interpretability is the following: Instead of learning models that are interpretable based on the predictions of the original model, one can learn rules or program (in some language) that can approximate a machine learning model or special cases of it. The problem of deriving rules from the model predictions is closely related to the problem of reverse engineering queries, which is to derive the specification from known behaviors such as known input and output mappings (e.g., [7, 75, 48] to name some recent work). These rules can be further abstracted to provide human friendly explanations for the model [74]. Interestingly, the process of reverse engineering often involves developing a machine learning model to learn a query for the given input and output data, which itself may require explanations.

3.3 Provenance and social media

Social media, such as Facebook, Instagram, Twitter etc., are an effective vehicle for disseminating news at scale. They provide an easy platform for users to continuously communicate and network with one another. The continuity and scale are critical characteristics that set it apart from traditional forms of communications such as phones, television, or newspapers. Unfortunately, its effectiveness for disseminating information has also been exploited for disseminating fake news and fake claims.

There has been substantial interest lately in how to detect fake news articles or fake claims (e.g., see [1, 4, 16, 47, 80, 77]), and having adequate provenance is seen as an essential part of this process. We discuss some potential directions for further work and argue that building a mechanism for understanding the provenance of news obtained through the social network is an important part of determining fake news or fake claims.

As with data provenance, the provenance of a piece

of information found in an article or statement in social media should explain why that information is there and how it was created. One method of achieving this is to ensure that provenance is disseminated along with news propagation. We should also discredit news without mechanisms for authenticating its provenance. When an article is first created, it should include information such as the authorship and attribution to sources. The social network software responsible for disseminating the news should add the identity of the receiver into the chain of provenance information. Furthermore, there should be tamper-proof mechanisms built into the software to prevent the identity from being modified.

If provenance information may not be immediately available from an article, can we infer the provenance with social media network? For example, [72] identifies the source of rumor when all recipients are known (*rumor-centrality*). In [53], the *effectors* are determined under the independent cascade propagation model and in [65], the NetSleuth approach [65] estimates the sources under the assumption of the Susceptible Infected information propagation model. As shown in [33, 41], some provenance attributes can also be recovered from various social media websites and can lead to better knowledge of the sources.

A promising area for further research is to incorporate provenance into the *fact checking* problem. Fact checking originated from the data journalism community and refers to the problem of determining whether or not factual claims in media content are true. Today, there are websites[1] dedicated to analyzing and reasoning about facts. Google also supports an API for reviewing claims[2]. Note that whether a fact is true or not is actually independent of its provenance. However, since a trusted source tends to produce articles that are free of wrong facts, a property for judging whether a claimed fact is true or not can be based on the trustworthiness of the sources. In turn, this requires knowledge of the provenance attributes of these sources. Can we use provenance to as a reliable signal for determining whether a fact is true or not? Some recent work has begun to incorporate such information in determining the truth of news/facts [69]. Another promising direction is to incorporate trust and reputation management into social media. Can we maintain a reputation rating for different sources based on their history of the authenticity of news articles and correct facts that are wrongly reported and shared. In turn, these reputation ratings can be used as another signal for fact checking and checking for fake news [29]. Regardless of the method used to determine sources of fake news or fake claims, it is crucial that provenance about the sources can be obtained or inferred. It is also critical to create standards to institute a minimum set of attributes that should be provided before an author can publish or responsibly propagate an article on any social media platform.

[1]https://www.factcheck.org, https://www.truthvalue.org
[2]https://developers.google.com/search/docs/data-types/factcheck

3.4 Provenance and blockchain technology

Blockchain technology, or more generally, Distributed Ledger Technology (DLT) has been developed to keep a distributed immutable ledger of financial transactions. The ledger can be seen as a provenance record of, say, bitcoins; and it is therefore entirely unsurprising that DLT could be used to record provenance in other settings. There is some commercial interest in using DLT to record supply side provenance – for example the farm from which a lamb chop originated [56, 73], and there have been suggestions that it could be used for valued artefacts [70]. Superficially this kind of provenance looks rather like where-provenance for digital artefacts. Indeed there is at least one system [55] that has been developed to record data provenance at the level of file systems. The system-level provenance [62] operations on files such as read, write, share and modify are recorded using DLT.

Whether the cost of current DLT justifies its use for these applications or whether there are sufficient financial incentives to maintain a distributed ledger for the provenance of artefacts are questions well beyond the scope of this paper. However there is one interesting observation regarding data provenance. DLT was developed [63] in part to prevent "double spending": the same coin cannot be given to two parties, and a similar constraint holds for the provenance of artefacts. In nearly all forms of *data* provenance, it is understood that data gets copied, thus we do not need this constraint. Whether this will allow us to to develop simpler or less costly distributed ledgers for data provenance is an open question.

3.5 Provenance and privacy

On the face of it, provenance negates privacy. Gaining knowledge of where some piece of clinical data has come from is exactly what techniques such as differential privacy are designed to prevent. This contradiction itself poses some interesting questions because there are many situations in which we want both provenance and privacy. Imagine, for example that we have some clinical patient records provided by a hospital H and a research group R that wants to analyze some of the data in those records. H writes programs to export anonymized data to R and R writes some analysis programs. H and R interact, and both H and R keep provenance associated with their activities perhaps for repeatability as described in Section 2.2. In what sense have they kept

enough provenance to describe the combined interaction?

This raises some interesting issues with provenance models. In what sense can we *compose* the provenance descriptions of two interacting activities. In the simple world of database queries, composition is a natural requirement and is usually satisfied. The provenance of the composition of two queries can be easily derived from the provenance of each of those queries. However, it is not clear how in, for example, PROV [60] one might glue together two provenance graphs of interacting activities, and whether this would be a satisfactory model of the combined activity. In our example of medical records, supposed R discovered some anomaly that indicated that H had a patient at risk. Would one have enough information to identify that patient? Also, suppose that neither R nor H wanted to reveal their individual provenance data, could some secure multi-party computation algorithm be used to identify the patient?

4. CONCLUSIONS

We have attempted to describe some areas in which data provenance is finding applications and is opening up new lines of research. There is no doubt that the theory of provenance, annotation in relational databases, and versioning will continue to develop and will be developed for other data models. Some examples of recent work in these areas include [36], where semirings are extended to capture the semantics of SPARQL queries (with OPTIONAL) on annotated RDF data and [38] where semirings are extended to deal with negation.

However the developments that will have the most impact will, we believe, stem from the public understanding of provenance. For example, we have seen how provenance can be understood and exploited in the social media, but there are even simpler situations in which one could develop useful applications of provenance. Consider the apparently innocuous copy and paste operations and how much provenance has been lost in their use. It would surely be a relatively simple matter to instrument these operations to carry some kind of provenance token that is generated for the source data (document, spreadsheet etc.) and for this to be carried across, along with the data being copied into a provenance repository associated with the target. In experimental environments for curated databases, such a mechanism has already been shown to be workable [9] and not at all costly in resources.

Today, the prevalence of open data [3] makes it even more compelling for data providers and consumers alike to instrument such provenance-aware generation and copy-paste mechanisms. Just as we prefer to read documents with proper authorship and from trusted sources, shouldn't we place higher value on documents that contain provenance or are generated by editors that are provenance-aware? Isn't it time to instrument good "provenance manners" to practice for the mass market by enabling documents to generate provenance tokens and editors to be provenance-aware?

5. REFERENCES

[1] H. Allcott and M. Gentzkow. Social media and fake news in the 2016 election. *J. of Economic Perspectives*, 31(2):211–236, 2017.

[2] B. Arab, S. Feng, B. Glavic, S. Lee, X. Niu, and Q. Zeng. GProM - a swiss army knife for your provenance needs. *IEEE Data Eng. Bull.*, 41(1):51–62, 2018.

[3] J. Attard, F. Orlandi, S. Scerri, and S. Auer. A systematic review of open government data initiatives. *Government Information Quarterly*, 32(4):399–418, 2015.

[4] G. Barbier, Z. Feng, P. Gundecha, and H. Liu. *Provenance Data in Social Media*. Morgan & Claypool Publishers, 2013.

[5] O. Benjelloun, A. D. Sarma, A. Y. Halevy, and J. Widom. Uldbs: Databases with uncertainty and lineage. In *VLDB*, pages 953–964, 2006.

[6] D. Bhagwat, L. Chiticariu, W. C. Tan, and G. Vijayvargiya. An annotation management system for relational databases. *VLDB J.*, 14(4):373–396, 2005.

[7] A. Bonifati, R. Ciucanu, and S. Staworko. Learning join queries from user examples. *TODS*, 40(4):24:1–24:38, 2016.

[8] S. Bowers. Scientific workflow, provenance, and data modeling challenges and approaches. *J. Data Semantics*, 1(1):19–30, 2012.

[9] P. Buneman, A. Chapman, and J. Cheney. Provenance management in curated databases. In *SIGMOD*, pages 539–550. ACM, 2006.

[10] P. Buneman, J. Cheney, and S. Vansummeren. On the expressiveness of implicit provenance in query and update languages. *TODS*, 33(4):28:1–28:47, 2008.

[11] P. Buneman, S. Davidson, and J. Frew. Why data citation is a computational problem. *Communications of the ACM*, 59(9):50–57, 2016.

[12] P. Buneman, S. B. Davidson, W. Fan, C. S. Hara, and W. C. Tan. Keys for XML. *Computer Networks*, 39(5):473–487, 2002.

[13] P. Buneman, S. Khanna, K. Tajima, and W. C. Tan. Archiving scientific data. *ACM TODS*, 29:2–42, 2004.

[14] P. Buneman, S. Khanna, and W.-C. Tan. Why and where: A characterization of data provenance. In *ICDT*, pages 316–330, 2001.

[15] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad. Intelligible models for

healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *SIGKDD*, pages 1721–1730, 2015.

[16] S. Cazalens, J. Leblay, I. Manolescu, P. Lamarre, and X. Tannier. Computational fact-checking: a content management perspective. *PVLDB*, 11(12):2110–2113, 2018.

[17] C. Chen, A. Seff, A. Kornhauser, and J. Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *ICCV*, ICCV, pages 2722–2730, 2015.

[18] J. Cheney, L. Chiticariu, and W. C. Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474, 2009.

[19] J. Cheney, S. Chong, N. Foster, M. I. Seltzer, and S. Vansummeren. Provenance: a future history. In *SIGPLAN*, pages 957–964, 2009.

[20] L. Chiticariu, W. C. Tan, and G. Vijayvargiya. Dbnotes: a post-it system for relational databases based on provenance. In *SIGMOD*, pages 942–944, 2005.

[21] Y. Cui and J. Widom. Practical lineage tracing in data warehouses. In *ICDE*, pages 367–378, 2000.

[22] Y. Cui, J. Widom, and J. L. Wiener. Tracing the lineage of view data in a warehousing environment. *ACM TODS*, 25(2):179–227, 2000.

[23] C. Curino, H. J. Moon, A. Deutsch, and C. Zaniolo. Update rewriting and integrity constraint maintenance in a schema evolution support system: PRISM++. *PVLDB*, 4(2):117–128, 2010.

[24] DataCite Metadata Schema for the Publication and Citation of Research Data. http://schema.datacite.org/meta/kernel-3/doc/DataCite-MetadataKernel_v3.1.pdf, October 2014.

[25] S. B. Davidson, S. C. Boulakia, A. Eyal, B. Ludäscher, T. M. McPhillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4):44–50, 2007.

[26] S. B. Davidson, S. C. Boulakia, A. Eyal, B. Ludäscher, T. M. McPhillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4):44–50, 2007.

[27] S. B. Davidson, D. Deutch, T. Milo, and G. Silvello. A model for fine-grained data citation. In *CIDR*, 2017.

[28] S. B. Davidson and J. Freire. Provenance and scientific workflows: challenges and opportunities. In *SIGMOD*, pages 1345–1350. ACM, 2008.

[29] L. de Alfaro, M. D. Pierro, E. Tacchini, G. Ballarin, M. L. D. Vedova, and S. Moret. Reputation systems for news on twitter: A large-scale study. *CoRR*, abs/1802.08066, 2018.

[30] A. Deutsch, L. Popa, and V. Tannen. Query reformulation with constraints. *SIGMOD Record*, 35(1):65–73, 2006.

[31] F. Doshi-Velez and B. Kim. A roadmap for a rigorous science of interpretability. *CoRR*, 2017.

[32] J. R. Driscoll, N. Sarnak, D. D. Sleator, and R. E. Tarjan. Making data structures persistent. *JCSS*, 38(1):86–124, 1989.

[33] Z. Feng, P. Gundecha, and H. Liu. Recovering information recipients in social media via provenance. In *ASONAM*, pages 706–711, 2013.

[34] FORCE11. Data citation synthesis group: Joint declaration of data citation principles. https://www.force11.org/datacitation, 2014.

[35] F. Geerts, A. Kementsietsidis, and D. Milano. MONDRIAN: annotating and querying databases through colors and blocks. In *ICDE*, page 82, 2006.

[36] F. Geerts, T. Unger, G. Karvounarakis, I. Fundulaki, and V. Christophides. Algebraic structures for capturing the provenance of SPARQL queries. *JACM*, 63(1):7:1–7:63, 2016.

[37] B. Glavic, R. J. Miller, and G. Alonso. Using sql for efficient generation and querying of provenance information. *In search of elegance in the theory and practice of computation: a Festschrift in honour of Peter Buneman*, pages 291–320, 2013.

[38] E. Grädel and V. Tannen. Semiring provenance for first-order model checking. *CoRR*, abs/1712.01980, 2017.

[39] T. J. Green, G. Karvounarakis, Z. G. Ives, and V. Tannen. Provenance in ORCHESTRA. *IEEE Data Eng. Bull.*, 33(3):9–16, 2010.

[40] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40, 2007.

[41] P. Gundecha, S. Ranganath, Z. Feng, and H. Liu. A tool for collecting provenance data in social media. In *SIGKDD*, pages 1462–1465, 2013.

[42] T. S. Guzella and W. M. Caminhas. Review: A review of machine learning approaches to spam filtering. *Expert Syst. Appl.*, 36(7):10206–10222, 2009.

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

[44] K. W. Hart, D. B. Searls, and G. C. Overton. Sortez: A relational translator for ncbi's asn. 1 database. *Bioinformatics*, 10(4):369–378, 1994.

[45] M. Herschel, R. Diestelkämper, and H. Ben Lahmar. A survey on provenance: What for? what

form? what from? *VLDB J.*, 26(6):881–906, 2017.
[46] S. Huang, L. Xu, J. Liu, A. J. Elmore, and A. G. Parameswaran. Orpheusdb: Bolt-on versioning for relational databases. *PVLDB*, 10(10):1130–1141, 2017.
[47] D. Ignatius. How to protect against fake facts. The Washington Post, November 2017.
[48] D. V. Kalashnikov, L. V. S. Lakshmanan, and D. Srivastava. Fastqre: Fast query reverse engineering. In *SIGMOD*, pages 337–350, 2018.
[49] G. Karvounarakis and T. J. Green. Semiring-annotated data: queries and provenance? *SIGMOD Record*, 41(3):5–14, 2012.
[50] M. Kaufmann, P. M. Fischer, N. May, and D. Kossmann. Benchmarking bitemporal database systems: Ready for the future or stuck in the past? In *EDBT*, pages 738–749, 2014.
[51] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, A. McKeown, G. Yang, X. Wu, F. Yan, J. Dong, M. Y. T. Made K. Prasadha and, Jacqueline Pei and, J. Zhu, C. Li, S. Hewett, J. Dong, I. Ziyar, R. Z. Alexander Shi and, L. Zheng, R. Hou, W. Shi, X. Fu, Y. Duan, V. A. Huu, C. Wen, E. D. Zhang, C. L. Zhang, O. Li, M. A. S. Xiaobo Wang and, X. Sun, J. Xu, A. Tafreshi, M. A. Lewis, H. Xia, and K. Zhang. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172:1122–1131.e9, 2018.
[52] K. Kulkarni and J.-E. Michels. Temporal features in sql:2011. *SIGMOD Rec.*, 41(3):34–43, 2012.
[53] T. Lappas, E. Terzi, D. Gunopulos, and H. Mannila. Finding effectors in social networks. In *SIGKDD*, pages 1059–1068, 2010.
[54] M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.
[55] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla. Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 468–477. IEEE Press, 2017.
[56] P. P. Ltd. Retrieved September 12 2018.
[57] M. Maddox, D. Goehring, A. J. Elmore, S. Madden, A. G. Parameswaran, and A. Deshpande. Decibel: The relational dataset branching system. *PVLDB*, 9(9):624–635, 2016.
[58] P. Missier and L. Moreau. PROV-dm: The PROV data model. W3C recommendation, W3C, Apr. 2013. http://www.w3.org/TR/2013/REC-prov-dm-20130430/.
[59] H. J. Moon, C. Curino, M. Ham, and C. Zaniolo. PRIMA: archiving and querying historical data with evolving schemas. In *SIGMOD*, pages 1019–1022, 2009.
[60] L. Moreau and P. Groth. "provenance: an introduction to prov". *Synthesis Lectures on the Semantic Web: Theory and Technology*, 3(4):1–129, 2013.
[61] L. Moreau, B. Ludäscher, I. Altintas, R. S. Barga, S. Bowers, S. Callahan, G. Chin Jr, B. Clifford, S. Cohen, S. Cohen-Boulakia, et al. Special issue: The first provenance challenge. *Concurrency and computation: practice and experience*, 20(5):409–418, 2008.
[62] K.-K. Muniswamy-Reddy, D. A. Holland, U. Braun, and M. I. Seltzer. Provenance-aware storage systems. In *USENIX Annual Technical Conference, General Track*, pages 43–56, 2006.
[63] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, 2008.
[64] F. Poursabzi-Sangdeh, D. G. Goldstein, J. M. Hofman, J. W. Vaughan, and H. M. Wallach. Manipulating and measuring model interpretability. *CoRR*, 2018.
[65] B. A. Prakash, J. Vreeken, and C. Faloutsos. Spotting culprits in epidemics: How many and which ones? In *IEEE ICDM*, pages 11–20, 2012.
[66] S. Pröll and A. Rauber. Scalable data citation in dynamic, large databases: Model and reference implementation. In *IEEE Intl. Conf. on Big Data*, pages 307–312, 2013.
[67] Research Data Alliance (RDA), Data Citation WG.
[68] J. F. Roddick. A survey of schema versioning issues for database systems. *Information & Software Technology*, 37(7):383–393, 1995.
[69] N. Ruchansky, S. Seo, and Y. Liu. CSI: A hybrid deep model for fake news detection. In *CIKM*, pages 797–806, 2017.
[70] M. Schuetz. Startup codex brings blockchain to art with backing from pantera, February 2018. Retrieved September 2018.
[71] P. Senellart. Provenance and probabilities in relational databases. *SIGMOD Record*, 46(4):5–15, 2017.
[72] D. Shah and T. Zaman. Rumors in a network: Who's the culprit? *IEEE Trans. Inf. Theor.*, 57(8):5163–5181, 2011.
[73] E. Stahl. "blockchain provenance: Wheres my new sweater really from?", November 2017. Retrieved September 2018.
[74] B. ten Cate, C. Civili, E. Sherkhonov, and W.-C. Tan. High-level why-not explanations using ontologies. In *PODS*, pages 31–43, 2015.
[75] B. ten Cate, P. G. Kolaitis, K. Qian, and W.-C. Tan. Active learning of gav schema mappings. In

PODS, pages 355–368, 2018.

[76] S. Veenadhari, B. L. Misra, and C. Singh. Machine learning approach for forecasting crop yield based on climatic parameters. *2014 International Conference on Computer Communication and Informatics*, pages 1–5, 2014.

[77] X. Wang, C. Yu, S. Baumgartner, and F. Korn. Relevant document discovery for fact-checking articles. In *WWW*, pages 525–533, 2018.

[78] Y. R. Wang and S. E. Madnick. A polygen model for heterogeneous database systems: The source tagging perspective. In *VLDB*, pages 519–538, 1990.

[79] A. Woodruff and M. Stonebraker. Supporting fine-grained data lineage in a database visualization environment. In *ICDE*, pages 91–102, 1997.

[80] Y. Wu, P. K. Agarwal, C. Li, J. Yang, and C. Yu. Computational fact checking through query perturbations. *TODS*, 42(1):4:1–4:41, 2017.

[81] W. Zhou, Q. Fei, A. Narayan, A. Haeberlen, B. T. Loo, and M. Sherr. Secure network provenance. In *SOSP*, pages 295–310, 2011.

[82] W. Zhou, M. Sherr, T. Tao, X. Li, B. T. Loo, and Y. Mao. Efficient querying and maintenance of network provenance at internet-scale. In *SIGMOD*, pages 615–626, 2010.

Don't Even Ask: Database Access Control through Query Control

Richard Shay Uri Blumenthal Vijay Gadepally
Ariel Hamlin John Darby Mitchell Robert K. Cunningham
MIT Lincoln Laboratory, Lexington, MA USA
{richard.shay, uri, vijayg, ariel.hamlin, mitchelljd, rkc} @ ll.mit.edu

ABSTRACT

This paper presents a vision and description for query control, which is a paradigm for database access control. In this model, individual queries are examined before being executed and are either allowed or denied by a pre-defined policy. Traditional view-based database access control requires the enforcer to view the query, the records, or both. That may present difficulty when the enforcer is not allowed to view database contents or the query itself. This discussion of query control arises from our experience with privacy-preserving encrypted databases, in which no single entity learns both the query and the database contents. Query control is also a good fit for enforcing rules and regulations that are not well-addressed by view-based access control. With the rise of federated database management systems, we believe that new approaches to access control will be increasingly important.

1. INTRODUCTION

There are great opportunities associated with large-scale data collection, but also associated risks. Increasing privacy concerns about data collection are demonstrated by the recent European Union General Data Protection Regulation (GDPR) [6]. There is a need for greater privacy protections in securing, storing, and transmitting big data [22]. In this paper, we advance the concept of *query control*, an expressive database access control strategy.

Commonly used view-based access control restricts a user's *view* of the database. Query control is an alternative, complementary database access control strategy based on examining what queries a user is submitting. Each querier is assigned a *query control policy*; that querier's queries can only execute if they conform to the policy. Query control limits the questions being asked, rather than directly limiting the data being returned. Query control may be especially useful in enforcing policies that limit the questions that are allowed to be asked of a database, or limiting how data sets are utilized [32].

The use case that first motivated the development of query control is access control for privacy-preserving databases, in which encrypted queries are executed against encrypted data and encrypted results are returned to the querier [9, 14]. In addition to protecting data privacy, such databases protect the privacy of queries submitted and results returned from both the data owner and the entity processing the queries. As we will elaborate upon in Section 3, query control allows a third party to enforce access control on encrypted queries for an encrypted database, while preserving data privacy for both. Another promising use case of query control is in management systems supporting multiple heterogenous, federated databases. Such systems may need the ability to execute a single query across multiple individual database engines, each with its own access control and data storage approach [31]. Query control will enable database-agnostic access control policies to effect centralized, unified access control across diverse, composite database systems.

We present a brief overview of database access control, highlighting where query control fits, in Section 2. We present the background and history of query control, with a focus on privacy-preserving databases, in Section 3. Section 4 discusses salient use cases for query control. In Section 5, we present a high-level overview of our reference implementation for query control; while the focus of this paper is highlighting the case for using it, this reference implementation demonstrates its feasibility. Finally, we present promising future applications of query control in Section 6 and conclude in Section 7.

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited. This material is based upon work supported under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Air Force.

2. DATABASE ACCESS CONTROL

In order to understand the function of query control in database access control, it is useful to begin with a larger picture of database access control and then zoom in. Access control in databases has played an integral role in their development and popularity [7, 12]. Access control can be used at granularities such as the level of a table, individual rows, or even individual cells. Depending on the access control implementation, controlling which users can access what data entries can be a challenging task [3], sometimes amplified by application-specific requirements [4].

In general, *database access control* limits the access of a principal, a user or users, to the contents of a database. We separate the concepts of access control *strategies* and access control *mechanisms*. An access control *strategy* refers to how access control policies are assigned to principals, whereas an access control *mechanism* determines how access to the database is restricted. View-based access control, the most common access control mechanism found in production database systems, is data-dependent and is often implemented through metadata. *Query control* places a restriction on the queries that a principal can issue, and is therefore not data-dependent.

Access control strategies are orthogonal to access control mechanisms. For example, role-based strategies can be applied to view-based or query-based mechanisms. For the most part, relational database management systems have concentrated on view-based mechanisms with varying access control strategies. Therefore, these terms are often conflated and role-based access control in many contexts implies a view-based access control mechanism with a role-based strategy. It should be noted that multiple access control mechanisms need not be mutually exclusive. On a single database, query control can be used to limit the queries that are actually executed, and view-based access control can be used to limit the results returned.

2.1 Traditional Database Access Control

In a view-based database access control model, a principal requests access to database contents. The system evaluates whether the principal is authorized to access the database contents by examining the access control policy. Often, an access control policy depends on the contents being accessed. The system issues a decision that either allows or denies access. View-based access control uses a database view as an abstraction mechanism for the data available to a particular principal [12].

There are a number of historical models for access control strategies applied to the view-based access control mechanism [1, 2]. Some early strategies, such as Discretionary Access Control and Mandatory Access Control [25], were often implemented via individual or group level access control. Role-based access control is a popular way to implement access control policies [24].

Query re-writing was initially explored for optimization [13]. Rizvi et al. discuss using it for access control, such as ensuring a given column has a specific value [23]. This changes a query to match a policy, rather than rejecting a non-conforming query like query control. Re-writing requires the enforcement mechanism to have direct query access, which may not work in the privacy preserving use case for which query control was created.

2.2 Query Control Policy Definition

We formally define *Query Control* as a protocol between four entities: a *data owner* who provides the database and policies; a *data host* who hosts the database; a *policy enforcer* who determines if a query is valid according to the given policy; and, finally, a *querier* who is the principal issuing queries. Often the data host is the same entity as the data owner or policy enforcer.

The querier has been assigned a policy to restrict the queries that it can issue. There may be multiple queriers, each with separate query control policies. Each query issued by the querier is evaluated by the policy and met with either *accept* or *reject*. The decision is *accept* if and only if the query is properly formed and is acceptable based on the applicable policy. Otherwise, the decision is *reject*.

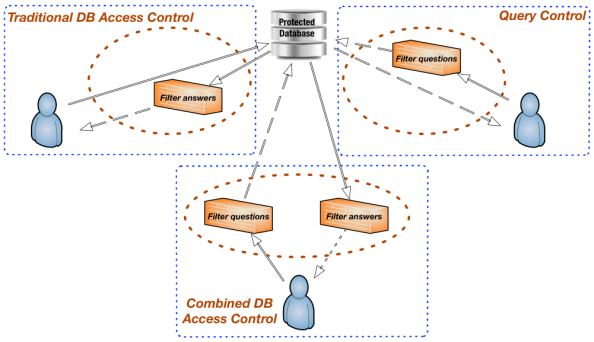


Figure 1: Placement of access control mechanisms on the database interaction path.

Figure 1 shows where the access control mechanisms can be applied. The traditional approach applies access control to the query result before being returned to the querier, filtering out what the

querier is not allowed to receive. This contrasts with query control, where the received query is evaluated against policy *before* it is executed by the database.

3. PRIVACY-PRESERVING DATABASES

The concept of query control came from IARPA research into privacy-preserving databases, which allow organizations to share data in a precisely controlled way [14]. Here, a privacy-preserving database means the data owner is the source of database records and wants assurance that any query adheres to a given policy. A querier submitting a query learns the details of any records that satisfy that query, but the data owner does not learn the contents or results of the query. Selecting query control as a model to preserve privacy enables a third party to enforce access control without learning about database contents and without the data owner learning about the contents of queries. Fuller et al. contains a more detailed explanation of the different design approaches and leakage tradeoffs of privacy-preserving databases [9].

This approach to a privacy-preserving database, with encrypted database and encrypted queries, constrains how access control can be implemented. Enabling a privacy budget requires that a single entity calculate on the distribution of data and also view queries. In this model, there is no such entity, and therefore a traditional privacy budget is not feasible. Further, this does not lend itself to changing permissions based on system load, as access control decisions may be made by an entity without any insight into the state of the database system.

The predecessor to SPAR, the Automatic Privacy Protection (APP) program, initially developed technology that included coarse query control [14]. Kagal used the AIR language for creating and enforcing permissions for semantic web technology [18], with language features such as restricting database columns [15]. Further work demonstrated that policy compliance can be enforced without being able to view database contents [27].

Following the success of APP, the SPAR program substantially increased the scope of research into privacy preserving databases. Performers designed their own mechanisms to express and enforce policies and integrated query control securely into query processing [14]. These query control mechanisms demonstrated a diverse range of capabilities but were found to be insufficient to express and enforce the variety of policy rules the government wanted. As a result, the query control policy language in Section 5 was developed under the subsequent Security and Privacy Assurance Research Software Evaluation (SPARSE) program. This program illustrates a real-world situation that called for query control, and provided an opportunity to create a reference implementation for a query control policy language.

4. QUERY CONTROL USE CASES

Query control can complement traditional view-based access control by filling in gaps in that access control strategy. Query control can replicate some types of control found in the view-based access control strategy – specifically column-based portions of view-based access control. Further, there are a number of restrictions that can be placed on queries issued by the querier that would not be easily imposed by view-based access control. These mechanisms are not mutually exclusive and can be used in tandem.

Query control policies can utilize any number of conjunctions (AND, OR, NOT) and can therefore express and enforce rules that contain conditionals. This means query control is well-suited for some types of natural-language database access control policies. Consider a policy that requires a query to ask only about a particular doctor's patients, unless that query also restricts itself to patients with a particular medical condition. This is easy to express in English, but not easily expressed using a database view. Because there is a conditional in this policy, no single static view of the database suffices to represent it. However, this policy can easily be expressed as a set of atomic rules combined with conjunctions (using the language to be presented in Section 5): (doctor_last_name == "Tyre") or (med_condition is included).

Query control can be enforced without needing access to the underlying database contents. Both approaches require access to the database schema, but view-based access control also requires access to the database contents. With query control, the results of the evaluation do not change if the database contents change. Further, defects in the data do not impact the query control decision.

Query control also lends itself more naturally to some types of time-based policies that might mirror natural language policy text. Kagal points out that a policy permitting access only to records on individuals who are at least 18 years of age can be difficult to implement using view-based access control [16]. Using query control, this becomes trivial: birthday at least 18 years ago. More complex policy examples will be presented in Section 5.1.

5. REFERENCE IMPLEMENTATION

In this section, we briefly describe a language we developed for query control, called Query Control Policy Language (QCPL). This is a preliminary sketch to demonstrate the feasibility of a query control language, and not a complete language specification. QCPL facilitates the specification of *query control policies*, which are comprised of one or more query control rules. A query control rule is made up of any number of *atomic* query control rules, combined with conjunctions. These conjunctions (AND, OR, NOT) combining atomic query control rules enable expressive and complex query control policies.

A query control rule specifies a requirement for a query. For example, the rule Count = 3 specifies that queries may only be accepted if they require that database column *Count* have the value *3*. Query `SELECT * FROM table WHERE ((Count = 3))` satisfies this rule, but `SELECT * FROM table WHERE ((Count > 1))` does not. This rule is satisfied by the following query statement because both clauses satisfy the rule: `(Count = 3 AND A = 1) OR (Count = 3 AND A = 2)` However, this rule is not satisfied by the following because its second clause does not require that `Count` be 3, and therefore it does not ensure that *Count* have a value of *3*: `(Count = 3 AND A = 1) OR (Count = 2)`.

In order to demonstrate its feasibility, we implemented the language in 1,486 lines of Ruby.[1] We evaluated a set of 1203 queries, with a mean of 7.3 operations per query, across ten policies. In the interest of space, we describe only a few of these policies. P1 limits searching on low-cardinality fields; P3 ensures queries are within a particular timeframe; and P10 ensures that searches are limited to one event within a particular timeframe. It took only 18 seconds to evaluate all 1203 queries against all ten policies, of which 16.1 seconds were used to parse the queries. Figure 2 depicts the evaluation time of all queries by each policy.

5.1 Query Control Policy Example

Consider query control on a hypothetical database of hospital patients. A policy that only allows queries on patients who are at least 18 years old unless they were admitted in the past week would be non-trivial to implement via view-based access control. It is simple to express using QCPL, combining atomic rules via conjunctions: (birthday at least 18 years ago) OR (admit_date at most 7 days ago). Consider a a policy that requires that any query with a patient name must also include a patient birthday. This can be created by combining not and or operators: (not (patient_name is included)) OR (doctor_name is included) Further conjunctions are likewise simple to add to this policy. This illustrates how some access control policies that would be complex to implement in view-based access control can be easily expressed in QCPL.

[1] `ruby 2.0.0p648`

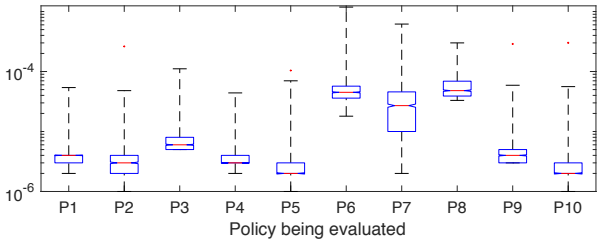


Figure 2: Evaluation time of 10 different query policies on 1203 SQL queries. Mean time is indicated by the red line. Box edges denote 25th and 75th percentiles. The Y-axis depicts time in seconds in log scale.

6. PROMISING FUTURE APPLICATIONS

6.1 Securing Consumer Data

Query control has promising applications in developing secure data-sharing solutions. One such use-case is securing consumer data. The big data phenomenon has resulted in numerous organizations collecting, storing, and processing large quantities of sensitive data. Once collected, data can be shared between organizations and within an organization – such as WhatsApp sharing user data with Facebook for advertisements [20]. In a context such as mobile devices, privacy concerns arise from users being unable to know who has their data and how their data are being used [19, 28]. If an organization is planning to share its user data with another organization, query control might be used to restrict how the second organization can access customer data.

6.2 Federated Database Systems

Developing heterogenous database management systems [31] may also benefit from an access control mechanism based on query control. These systems are built with the notion that future database systems may need to support multiple database "sizes" that are tuned to the underlying data they are storing or processing [29]. These systems are often characterized by support for heterogenous database management systems and multiple query and/or processing engines [26].

An example of such a system is the BigDAWG polystore system [10, 21]. Its current version [11] supports data querying of data stored in Apache Ac-

cumulo [17], a distributed key-value store database; PostGRES [30], a relational database; and SciDB [5], an array database. Each of these composite systems has its own access control mechanisms and strategies. Currently, developing access control for such systems may require using the "greatest common factor" of access-control-mechanism granularity across the disparate systems. As new systems are added, this challenge is compounded. Further, view-based access control heavily depends on the data being stored. Modern systems, such as BigDAWG, routinely copy data from one system to another in order to execute a query. Such challenges call for a new way of thinking about access control in database systems.

We believe that a *query-based* access control strategy can be readily applied to such systems, creating a centralized and abstract access control mechanism. Its evaluation need not depend on the database engine of any one system. Query control can be applied to any system with data stored in a predefined schema, which almost all database systems support. Thus, in the example system presented above, access control could be evaluated on the query directly and would not rely on the access control of underlying systems.

6.3 Usability Research

Query control has been initially studied through pilot testing under SPAR [8]. Further work is needed for validation of both query control as a concept and the QCPL language in particular. User studies can continue examination of how a user's experience is impacted if his or her queries are rejected. Further studies can examine how well data owners are able to express their ideal access control policies via QCPL, leading to improvements in the language.

Future work may also focus on how much a querier learns about a query control policy in a protected database. Attempting to obscure the query control policy from the querier leads to a number of interesting questions. If a querier is allowed an unlimited number of queries, he or she may discover that executing similar queries with minor changes allows discernment of part or all of the query control policy. Likewise, the querier may be able to compare the timing of queries that do and do not return results to learn about the query control policy.

7. CONCLUSION

In this paper, we have discussed query control and how it fits into the larger context of database access control. We have highlighted the past, present, and future of query control – how it was created to meet the needs of privacy-preserving databases, how organizations might benefit from it today, and how future use-cases might benefit from it even more. We advocate for further research into the space of query control, and for further usability studies. We believe that query control will become increasingly important as more data is accumulated, databases are increasingly federated, and searchable encryption becomes increasingly popular.

8. ACKNOWLEDGMENTS

The authors thank the many participants and program managers on IARPA-funded projects for interesting discussions and helpful feedback that improved this work.

9. REFERENCES

[1] E. Bertino and J. Crampton. Security for distributed systems: Foundations of access control. *Information Assurance: Survivability and Security in Networked Systems*, 2008.

[2] E. Bertino, G. Ghinita, and A. Kamra. Access control for databases: Concepts and systems. In *Foundations and Trends in Databases*, 2010.

[3] E. Bertino and R. Sandhu. Database security-concepts, approaches, and challenges. *IEEE Transactions on Dependable and secure computing*, 2005.

[4] M. A. Brookhart, T. Stürmer, R. J. Glynn, J. Rassen, and S. Schneeweiss. Confounding control in healthcare database research: challenges and potential approaches. *Medical care*, 2010.

[5] P. G. Brown. Overview of SciDB: large scale array storage, processing and analysis. In *SIGMOD International Conference on Management of data*, 2010.

[6] European Commission. 2018 reform of EU data protection rules. `https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en`.

[7] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed NIST standard for role-based access control. *TISSEC*, 2001.

[8] B. Fuller, D. Mitchell, R. Cunningham, U. Blumenthal, P. Cable, A. Hamlin, L. Milechin, M. Rabe, N. Schear, R. Shay, M. Varia, S. Yakoubov, and A. Yerukhimovich. SPAR pilot evaluation.

Technical report, MIT Lincoln Laboratory, 2015.

[9] B. Fuller, M. Varia, A. Yerukhimovich, E. Shen, A. Hamlin, V. Gadepally, R. Shay, J. D. Mitchell, and R. K. Cunningham. SoK: Cryptographically protected database search. *Oakland*, 2017.

[10] V. Gadepally, P. Chen, J. Duggan, A. Elmore, B. Haynes, J. Kepner, S. Madden, T. Mattson, and M. Stonebraker. The bigdawg polystore system and architecture. In *HPEC*, 2016.

[11] V. Gadepally, K. O'Brien, A. Dziedzic, A. Elmore, J. Kepner, S. Madden, T. Mattson, J. Rogers, Z. She, and M. Stonebraker. Bigdawg version 0.1. In *High Performance Extreme Computing Conference (HPEC), 2017 IEEE*, pages 1–7. IEEE, 2017.

[12] P. P. Griffiths and B. W. Wade. An authorization mechanism for a relational database system. *Transactions on Database Systems (TODS)*, 1976.

[13] A. Y. Halevy. Answering queries using views: A survey. *VLDB Journal*, 2001.

[14] Intelligence Advanced Research Projects Activity. Security and privacy assurance research (SPAR) program broad agency announcement, 2011.

[15] L. Kagal. Policy compliance of queries for private information retrieval. IARPA BAA Appendix E, August 2010.

[16] L. Kagal. Policy compliance of queries for private information retrieval. `http://dig.csail.mit.edu/2009/IARPA-PIR/`, August 2011.

[17] J. Kepner, W. Arcand, D. Bestor, B. Bergeron, C. Byun, V. Gadepally, M. Hubbell, P. Michaleas, J. Mullen, A. Prout, et al. Achieving 100,000,000 database inserts per second using accumulo and d4m. In *HPEC*, 2014.

[18] A. Khandelwal, J. Bao, L. Kagal, I. Jacobi, L. Ding, and J. Hendler. Analyzing the air language: A semantic web (production) rule language. In *International Conference on Web Reasoning and Rule Systems*, 2010.

[19] P. G. Leon, B. Ur, Y. Wang, M. Sleeper, R. Balebako, R. Shay, L. Bauer, M. Christodorescu, and L. F. Cranor. What matters to users? Factors that affect users' willingness to share information with online advertisers. In *SOUPS*, 2013.

[20] N. Lomas. WhatsApp to share user data with facebook for ad targeting – here's how to opt out. `https://techcrunch.com/2016/08/25/whatsapp-to-share`, August 2016.

[21] T. Mattson, V. Gadepally, Z. She, A. Dziedzic, and J. Parkhurst. Demonstrating the bigdawg polystore system for ocean metagenomics analysis. In *CIDR*, 2017.

[22] President's Council of Advisors on Science and Technology. Big data and privacy: A technical perspective. Technical report, Executive Office of the President, `https://bigdatawg.nist.gov/pdf/pcast_big_data_and_privacy_-_may_2014.pdf`, May 2014.

[23] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *SIGMOD International Conference on Management of data*, 2004.

[24] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[25] R. S. Sandhu and P. Samarati. Access control: principle and practice. *IEEE Communications*, 1994.

[26] A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys (CSUR)*, 1990.

[27] J. H. Soltren. Query-based database policy assurance using semantic web technologies. Master's thesis, MIT, September 2009.

[28] C. Spensky, J. Stewart, A. Yerukhimovich, R. Shay, A. Trachtenberg, R. Housley, and R. K. Cunningham. SoK: Privacy on mobile devices – it's complicated. In *Proceedings on Privacy Enhancing Technologies*, 2016.

[29] M. Stonebraker and U. Cetintemel. One size fits all: An idea whose time has come and gone. In *ICDE*, 2005.

[30] M. Stonebraker and L. A. Rowe. *The design of Postgres*, volume 15. ACM, 1986.

[31] R. Tan, R. Chirkova, V. Gadepally, and T. G. Mattson. Enabling query processing across heterogeneous data models: A survey. In *2017 IEEE International Conference on Big Data*, 2017.

[32] P. Upadhyaya, N. R. Anderson, M. Balazinska, B. Howe, R. Kaushik, R. Ramamurthy, and D. Suciu. Stop that query! The need for managing data use. In *CIDR*, 2013.

Timos Sellis Speaks Out on Research in Australia and Greece

Marianne Winslett and Vanessa Braganholo

Timos Sellis
https://www.swinburne.edu.au/science-engineering-technology/staff/profile/index.php?id=tsellis

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 Snowbird, Utah, USA, site of the 2014 SIGMOD and PODS conference. I have here with me Timos Sellis, who is a professor at the Royal Melbourne Institute of Technology[1]. Before that, he was at the National Technical University of Athens and the University of Maryland. He is an ACM Fellow and an IEEE Fellow, and he has a VLDB 10-Year Paper Award. His PhD is from Berkeley.

[1] Timos Sellis is currently the Director of Swinburne Data Science Research Institute at the Swinburne University of Technology in Australia.

So, Timos, welcome!

Thank you, Marianne. It is very nice to meet you.

Your VLDB 10-Year Paper Award was for introducing R+ trees in 1987. Is multidimensional indexing a solved problem now?

Well, to tell you the truth, multidimensional indexing has been used mostly for indexing spatial objects where the number of dimensions was, say, three at most. So what became interesting was that after a while, people started using multidimensional indexing to index higher dimensionality objects. There we could see that R-trees and all the variations of R-trees were not enough. So, it's not surprising that people are still working on multidimensional indexing.

I actually see people working on different platforms instead of following the standard hierarchical indexing methods we've been using in the past. They've tried to accommodate over MapReduce and distributed file systems. So, the interesting issue is scalability. The multidimensional indexing structures we have built in the past do not scale easily to a large number of dimensions. So, to some extent, it has been solved for the standard types of applications which have been seen in the past. But the fact is that we are getting more applications where you have what we call feature vectors with thousands of dimensions, right? Then apparently you cannot use these kinds of indexing methods anymore. So (a) you have to reduce the number of dimensions, so we have all these problems of dimensionality reduction and (b) you have to think of techniques that scale very well in thousands of dimensions. So, in my opinion, it's not a solved problem.

I think the more we see new interesting applications coming up (like from social networking applications), we'll see more work on multidimensional indexing. What is also different are the types of queries that people ask on these multidimensional indexes: they are no longer the standard type of range queries we have seen in the past. So, they want to do more complex kinds of analysis on these indexes, which makes it quite interesting in my opinion.

I think computer vision apps will be the next big thing for our field and they have very challenging index requirements, so I think there will be a little renaissance of work in this area.

To tell you the truth the most interesting applications I have seen up to now is where you have thousands of queries coming at the same time with thousands of updates. So, anything around location-based kinds of services or applications have this characteristic, where you have lots of updates coming in from the moving objects, yet at the same time, you have all these continuous queries that have to access the same indexes. So, coming up with indexes that can scale to thousands of updates and thousands of queries at the same time is an interesting problem.

Is there a gold standard for indexing for those kinds of apps?

You know we always thought that the hierarchical indexing methods (because of the *log n* kind of performance) were the solution. Our experience in the last few years that we've been working on these location-based applications is that you need the combination between the hierarchal indexing and grid or cell partitioning techniques. So, the old grid file that we used to have in spatial databases, combined with hierarchical indexes is something that we see more often coming out now. I think it's no longer a game of having, as you've said, one kind of hierarchical B-tree extension that will do it all. You need to combine approximate kinds of representations for the data because you cannot afford to index everything with all the details. So, it's an interesting landscape, I would say. That's why I'm excited for my Ph.D. students in Australia now we're coming back to look at similar issues around multidimensional indexing in different contexts, for example, in tweet analytics. Again, we can say for sure that the techniques we had in the past are not appropriate.

Someone was commenting to me today that researchers from Europe and Australia don't have as many papers in top conferences in our field as their counterparts in the US do. Why do you think that is?

So, I've been in Australia only for 1½ years, so I can tell you what my impression is on this issue. Australia is far away from everywhere. So, to send a student to a conference, or even a professor to attend a conference, that is very expensive. Funding is not that bad in Australia, but it is not enough to cover expenses to go to many conferences. So, I've noticed that people are more directed towards publishing in journals rather than in conferences. I think it's (a) financial, and (b) there is a culture in Australia that conferences do not count that much for promotion in tenure. Even in computer science and we know this discussion has been going on for years now. In Australia, coming from the Anglo-Saxon kind of system, they give more emphasis to journals. So, I'm surprised on how many journal publications I've seen from several groups in Australia, but I must confess that some of these people I didn't know because I wouldn't

see them in conferences. So, it's only if this particular work they did was of interest to me, I would find it in a journal, but it's not the people that we meet very often in conferences. I think that's the major reason. Now that I've seen that, I've told my students that I don't want to follow this principle. I would like them to send papers to conferences. You know, I see they have no problem sending papers and they do very good work. So, there are some very good groups in Australia. It's not that they are hesitating to submit papers to good conferences. I feel that because of the financial issue, they don't do it as much as in the US.

I feel that my biggest accomplishment is the fact that we have generated a very good new generation of Greeks in databases...

What about Europe versus the US?

Well, Europe you see, we have a lot of presence from European researchers in conferences, right? So, I don't think with Europe there is a problem of the same kind, especially since there is a lot of money going into research because of the European Union funded projects. So, I don't feel people are restricted to submit papers to conferences. I think in our community we see representatives from all over Europe, so it's not the same as with Australia. Now the difference that I see, for example, are the people coming from China. That's a huge difference compared to some years ago. I know very well that China is investing a lot in research. So many students have funding to come to conferences. This is not the case in Australia.

From the years that you spent in Greece what accomplishment are you most proud of?

So, I spent 20 years in Greece. That is a long time. I must confess that what I feel is my largest accomplishment is the fact that I helped create a culture in our database field. So, I'm very happy to see so many faces that have gone through my laboratory to do what you call in the US an "Honors Thesis" (the thesis that students do finishing their Undergraduate degree). I'm very happy that many of these students are now professors elsewhere or they are either doing their Ph.D.'s now. So, I feel that my biggest accomplishment is the fact that we have generated a very good new generation of Greeks in databases and of course after a while (I moved in '91 a few years after Yannis Ioannidis and later on to recent years like Minos Garofalakis, Antonis Deligiannakis many well-known Greek researchers came back). So I'm very happy to see this going on and I'm very happy to see new faces. I'm always proud of myself that we managed to create the environment that can generate international level work in Greece.

So it sounds like the pipeline is in place and operating.

It is.

Well, a related question. What has been the impact of the financial crisis on the research environment in Greece?

This is an interesting question because unlike other areas in Greece, research has not been influenced too much by the financial crisis. I'm sorry to say that the reason was that the government never funded research at a significant level. All the money was basically coming from projects that we were getting from the European Union. Even if it was from projects that were coming from Greek ministries, the money would come to these ministries from Europe. So once this hose is still open, the money comes in. I would say that the community has not been influenced too much.

Of course, what I have seen as a difference is that many students would not stay to study in Greece anymore. For a Ph.D., for example, they would prefer to go abroad just because they have more opportunities once they get through their Ph.D. It is easier to get a job abroad than getting jobs in Greece. For example, very few academic positions are now available in Greece and very limited other positions for people with Ph.D. or Masters degrees are available in Greece. So, to some extent, research has not been influenced in reality, so you will see that the work that gets published at international conferences are pretty much similar. We get the same number of students that go for a Ph.D., but the only difference I would say is that I would expect that in a few years we might see the numbers dropping just because not too many students would like to continue for a Ph.D. in Greece.

While I was working in Singapore in the last few years, I visited Australia a few times to recruit summer interns for our Research Institute. But I didn't get any takers. From that, I conclude that Australian's level of interest in research is lower than in the US. Although maybe there are other factors I don't know about. What is the research environment and culture like in Australia?

It's interesting that you didn't find any takers. I would imagine that you would not find any takers from native

Australians. So, what happens is that the majority of the graduate students that I see in Australia are international students. Australians, somehow, don't feel the need to go for a Ph.D. Some of them go for a Masters. Many of them just go directly to the job market. The unemployment rate is something like less than 6%, maybe 5%, so there are jobs. People don't feel the need to study any further to get jobs.

Given that most of the graduate students are international students, it would be very rare for these international students to leave Australia once they came, to go, for example, to the US. One thing that I've seen in the last couple of years that I've been there is that because the Australian dollar was kind of high, many students would not come any more to Australia, they would prefer countries where life is cheaper. That is why I can see that the number of students coming from China, for example, is declining. We have lots of students from India, Bangladesh, Iran, various places in the area, but I would suspect that the people from China, just because most of them come with scholarships from the government, they would prefer to go to the US because it's cheaper. So, I wouldn't say that Australians not interested in research. They are interested. Most of them prefer to go to the job market, get some experience first to see what the job market looks like, and then they may come back to do a Masters and perhaps a Ph.D., but it's not the typical thing I would see, for example, in Greece or in other countries where you would go immediately after your bachelors for a Masters and a Ph.D.

Is there a startup culture there?

Yes, there is very much a startup culture. I've seen more and more money from the government going to help startups. I've seen startups established in Australia come to the Silicon Valley and there are some very good examples. I think the culture is there, but I don't see that much in RMIT with our students in our undergraduate program. I cannot sense it that much, and that's the reason we decided to add entrepreneurship types of courses in our curriculum to help the students towards this direction. But it seems that the Australians do have this culture of taking the risk and getting something started up. I've seen many young people doing it.

Do you have any words of advice for fledging or mid-career database researchers?

I think what I will say will sound very traditional. My advice to both my students and as you say, researchers because I was lucky for the last six years to fund and run a new research institute in Athens in information systems and data management. So, I had the opportunity to help people come to our institute from fresh PhDs to mid-career people. I'm always trying to push the same thing that all of us are saying. Do something that you like. Don't follow necessarily the trend in terms of what is funded or don't hesitate. For example, I was telling them, "Come to me, I will support your research", to get some prototype out, for example. Then we can go for funding. Don't try to change your research to be closer to something that is being funded. Instead, do what you can do best. I can help you find the leads to an application area. We have this experience after so many years in the field. My advice would be to not change their career based on what seems to be fashionable.

This brings me a bit to this Big Data area. Somehow Australians are trying to advertise me as a Big Data person. I don't like that. Wherever I sit in a panel with various CEOs or CTOs or whatever, I usually make this introduction that Big Data is not a term that I would like for us in data management to cover ourselves under. I'm aware it's something that everyone understands when you talk about Big Data, but I'm just telling them that we were always dealing with Big Data.

Don't try to change your research to be closer to something that is being funded. Instead, do what you can do best.

True, we always were.

I don't want them to think that Big Data is something necessarily new. It's a new setting. It's an opening to many other areas. It's very interesting to work with data from various other areas, but I wouldn't like them to think of Big Data as the revolution that is happening nowadays.

Why is that bad? If they think it's a revolution, then they can get all excited about funding it for example or supporting it.

So, the reason I'm saying that is because I see that with many people, when they think about Big Data, they have the impression that it is something totally new and that we expect to hear something new. Of course, it has become almost a synonym for analytics, which is not false and I wouldn't say that is not the case. Most people think of Big Data in terms of data analytics. What I've seen is that people, at least in Australia, have seen this wave becoming large over there. The government,

funding, everybody talks about Big Data over there. It's good for us and I enjoy it, I must say. On the other hand, I don't want them to think that it is something totally new. They expect to hear new buzzwords. Personally, I find that difficult. I would like people to understand what we are doing and how this is related to Big Data or what they think is Big Data, rather than having to reinvent new kinds of terms or even methods to convince them that what we are doing is something totally new.

Anyway, Australia, I think at this time, is a good place for research in our area. That is the reason why I accepted to run SIGMOD there in 2015. So, I think it's good for our community to get the exposure in this country. It's not a coincidence that CIKM will also be in Melbourne next November. So, we will have both SIGMOD in June and CIKM in November. It's going to be a very interesting setting for people to come to Australia.

Big Data is not a term that I would like for us in data management to cover ourselves under […] we were always dealing with Big Data.

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

I think the work that we did in the last 6 years with one of my Ph.D. students (Kostas Patroumpas) which is around what we call positional streaming data which means coordinate data streaming in, no matter what the application is… (like XY coordinate data streaming). I think I have enjoyed the work in this area because it's the first time in my life where we've started from data models that you need, to query languages, to indexing, all the way down to the applications. I enjoyed the fact that, at least for me, it was the first time I addressed a data management area and I saw the whole thing from its theory, like what kind of algebra and operators we need to run in these kinds of applications, all the way down to building systems. I think it's probably the longest engagement I have had in my career. I've gone through various areas like query optimization, spatial data, data warehousing, personalization, but with this one, I liked how the idea evolved from geometry problems to query processing, to indexing, etc. So, I think I'm very happy that we managed to do work in this area for like 5 or 6 years, which has resulted in quite some interesting outcomes.

There are two papers that I think are quite interesting. Both of them appeared in SSDBM. The first one[2] has to do with techniques for positional steaming data where you get trajectories, and you would like to compress them online. So, imagine that you have all the cars and you want to record all trajectories. This is too much information. So, we looked at various techniques like, for example, if I can predict the location where you will be after a few seconds, then I don't need to store it. So, what are the samples that I take that I store, or I drop them because I can predict them?

The other one was in the same paper of what we call amnesic compression where you would like to have an online method where, as the data comes in, you approximate the past with enough information so that you have an idea what the trajectory was like, but you keep all the details for the present. So, the present is very accurate. As you go to the past, you get less detail, that's why we call it amnesic. This is work that I find interesting because you don't find equivalents for other types of data. Trajectory data brings opportunities for very interesting problems. So, compression was one.

The second one[3] is the one we got the Best Paper Award in SSDBM 2012 that was about privacy issues. So, with position streams, if you wish to instead of sending your actual location, to send a blurred kind of area where you're in, can you still answer interesting continuous queries? If I want to continuously know how many of my friends are around me or who is around me within a distance of say, 1 kilometer, where they don't send their actual locations, but they blur them a little bit and send me an area where they are, can I still answer this query with some probability? For example, I would like to say that 5 of your friends are in this area with a probability of 75%. We wanted to show that if you want to compromise, to some extent, accuracy but to gain privacy, you could do it with these methods. Again, these problems are interesting because you have the issue where the queries move with the users, and the objects move also move. You need to come up with these kinds results fast but at the same time, we wanted to show that even under uncertain situations, you can still answer these queries. So, I think that out of my

[2] Michalis Potamias, Kostas Patroumpas, Timos K. Sellis: Sampling Trajectory Streams with Spatiotemporal Criteria. SSDBM 2006: 275-284.

[3] Kostas Patroumpas, Marios Papamichalis, Timos K. Sellis: Probabilistic Range Monitoring of Streaming Uncertain Positions in GeoSocial Networks. SSDBM 2012: 20-37.

work, although I said we have done several things, I would pick these two as interesting.

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

I can tell you now that I've reached my mid-50s. What have I missed? I have missed working with systems people. When I was in Berkeley, I enjoyed working with Michael Stonebreaker. He was my advisor. I could see a person who understood and actually wanted to build systems out of every idea that came out. I think I missed that. I missed it in the course of my work in Greece. I see this opportunity now and I try to take it with the people now in Australia, to work with systems people (so, people who understand systems issues very well).

The second thing is the multidisciplinary environment. So, in Greece for the first time, after 20 years in my career, I started working with biologists and we worked for about 6 years together. I enjoyed that very much. So, picking an area where we can contribute with our knowledge from data management… I like that very much. With systems people, I want to work with them because I want them to influence my thinking about my problems in data management, but with another science kind of area, I would like to work with them, so we can influence this area as much as we can.

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

As a computer science researcher, I think I would pick the same answer as with the previous one. I would love to have emphasized more on systems issues. You know, I'm an electrical engineer by training, so my degree in the National Technical University of Athens in '82 was electrical engineering just because there was no computer science back then in Greece. If you're an engineer, you carry this continuous kind of interest on how things work. If I could have found a way to combine more engineering work or more systems work with the things I have been doing in the past, I think I would have enjoyed that very much.

Thank you very much for talking with me today.

Thank you, Marianne.

Peter Bailis Speaks Out on building tools users want to use

Marianne Winslett and Vanessa Braganholo

Peter Bailis

http://www.bailis.org/

Welcome to this installment of ACM Sigmod Records series of interviews with distinguished members of the database community. I'm Marianne Winslett and today we're at the 2017 SIGMOD and PODS Conference in Chicago. I have here with me Peter Bailis who's a professor at Stanford University. Peter won the 2017 ACM SIGMOD Jim Gray Dissertation award for his thesis entitled "Coordination Avoidance in Distributed Databases." Peter's advisors were Joseph Hellerstein, Ion Stoica, and Ali Ghodsi at Berkeley.

So, Peter, welcome!

Thanks.

What is your thesis about?

My thesis looks at distributed databases – if and when it's possible to build databases that execute concurrent operations without incurring communication across replicas. As we saw the rise of geo-distributed cloud computing, it became possible to run databases in multiple data centers, a setting where the speed of communication is fundamentally limited by the speed of light. And so the question we asked was: can I run transactions and other kinds of operations in my database without actually having these different replicas communicate?

Now, we knew from a bunch of research dating back to the 70s and 80s that you have to pay the price of this coordination via synchronous communication when you use conventional serializable transactions. But with the rise of many new applications, like those we saw in the online services (e.g., the Facebook social graph, maintaining distributed secondary indices), we wanted to know: could we satisfy these new types of application demands without coordination, and make them faster?

We are the database community, but I think more broadly we're the data-intensive systems and tools community. So finding users that will actually give you feedback on what you're working on, that can potentially adopt the algorithms or even the software that you're producing is incredibly valuable.

Was it a point paying for them?

There was a bit of a culture war between the database old guard, the David DeWitt's and Mike Stonebraker's of the world and this new class of developers, who threw a lot of the conventional wisdom from database management systems out the window and built their own class of data stores. These "NoSQL" developers started rebuilding databases from scratch and saying: "We don't need transactions. We don't want to run with the overhead of transactions for a number of reasons, one of which is scalability." And so, in our research, we found that we can actually provide many of the guarantees these developers wanted for their applications, but without the overhead of the conventional protocols that they had given up on.

There was a really interesting interplay between these evolving application demands and the core ideas behind conventional protocols, which in many cases were very close to what we'd like in the coordination-free setting, but not exactly. That is, we'd still use protocols like two phase commit in the design of these coordination-free algorithms. But we didn't use them with conventional synchronization mechanisms like locks. We also used a lot of multi-versioning but modified conventional versions of these protocols to scale while still providing guarantees that application developers wanted.

So, would you say you're working on a NoSQL killer? Or relational database killer?

The goal of my work and in particular this thesis is to provide useful tools that help people work more productively with their data. I think that as a community, we tell our users a lot of things that they should do. What I'm personally interested is in helping build tools that our users want to use in the first place. In the case of my thesis, developers had an application specification. They didn't have protocols to implement the specification. However, it turned out there was a lot of interesting theory and practical algorithms that came out of listening to what they wanted to use. We weren't throwing away the old theory, but adapting it to these new use cases and actually bringing it to practice. And so, the "relational versus NoSQL" debate is a bit of a red herring.

What I'd like to see more of our community doing and one of the things I'm proud of in this work is starting to bridge this divide between classical protocols, like consensus and two phase commit, and the demand of modern applications today. And I think that if you look at how programmers actually interact with transactional databases today, they need dramatically different interfaces, abstractions, and semantics than what we've built in the systems we provide them from the last 40 years.

So, can we find those features going into commercial systems now?

The work has seen various degrees of uptake. Some of the work we did early on in consistency prediction with the Apache Cassandra database, we just learned recently it's just now on Azure's CosmosDB. And some of the protocols we did for the secondary index maintenance, we have an ongoing dialogue with NoSQL developers about putting these into their systems as well.

That's great. Do you have any words of advice for graduate students or recent graduates?

The No. 1 piece of advice I'd give for grad students or recent graduates is find people who have real problems working with data. We are the database community, but I think more broadly we're the data-intensive systems and tools community. So finding users that will actually give you feedback on what you're working on, that can potentially adopt the algorithms or even the software that you're producing is incredibly valuable.

And I agree with you completely but in your specific case working on a classic database topic, most of our readers don't have that kind of shoulders rubbing with Facebook and other big companies that are facing this problem because most of them aren't located in Silicon Valley and other hotbeds. So, what does that advice mean for them?

That's a great question. There are many types of users and Internet services are only one type of user. I imagine most listeners are at or near a university and there are a large number of people at universities that are dealing with these sorts of data-intensive problems of crippling scale. Maybe not multi-data center databases, but, for instance, some of our work at Stanford right now is working with folks in Earth Sciences. They have all this seismic data coming in, with literal decades of archives that they'd like to process with more sophisticated methods. But they don't have the computational resources or the algorithms to scale them up.

So, I think that, at almost any university, if you go out and you spend some time doing some needs finding with domain scientists, with large amounts of data or even small amounts of data that could be dirty or not correctly labeled, there are interesting problems there. In a sense, your prerogative as a researcher is to actually step away from classic database systems. Don't work on faster serializable transaction processing. Don't work on query optimization. Don't work on relational analytics. Figure out what people in the wild who aren't necessarily Facebook and Google need to build.

It could be your roommate who is doing her Ph.D. in Biochemistry or in Earth Science. Go talk to them and ask them, "Hey, what do you do with data?" If you think about it, this is really the golden era of data. Everyone has recognized the value of data and yet the tools we have for dealing with data are geared towards a very particular, conventional, buttoned-up world of relational data management are really not in many cases adequate or serving the needs of the people who need it the most.

Working with this class of users that's beyond just the Facebooks and the Googles of the world, the folks who can't afford to hire the teams of data scientists to build these models to maintain their data and so on – that's where a lot of the new action is.

Great advice. Thank you very much for talking with us today.

Thanks.

NSF BIGDATA PI Meeting – Domain-Specific Research Directions and Data Sets

Lisa Singh[1], Amol Deshpande[2], Wenchao Zhou[1],
Arindam Banerjee[3], Alex Bowers[4], Sorelle Friedler[5], H.V. Jagadish[6],
George Karypis[3], Zoran Obradovic[7], Anil Vullikanti[8], Wangda Zuo[9]

[1]Georgetown University, [2]Univ. of Maryland at College Park, [3]Univ. of Minnesota, Twin Cities, [4]Columbia University,
[5]Haverford College, [6]University of Michigan, [7]Temple University, [8]Virginia Tech., [9]University of Colorado

1. SUMMARY

In March 2017, PIs and co-PIs funded through the NSF BIGDATA program were brought together along with selected industry and government invitees to discuss current research, identify current challenges, discuss promising future directions, foster new collaborations, and share accomplishments, at BDPI-2017. Given that two recent NITRD [2] and NSF [1] meeting reports contained a set of recommendations, grand challenges, and high impact priorities for Big Data, the organizers of this meeting shifted the focus of the breakout sessions to discuss problems and available data sets that exist in five application domains – policy, health, education, economy & finance, and environment & energy. These domains were selected based on a survey of the PIs/co-PIs and should not be interpreted as being more important than others. Slides that were presented by the different breakout group leaders are available at `https://www.bi.vt.edu/nsf-big-data/`. We hope this report will serve as a blueprint for promising big data research in five application domains.

2. COMMON BIG DATA RESEARCH CONCERNS AND CHALLENGES

While a number of unique domain-specific challenges were identified, some broader concerns were repeatedly mentioned across the breakout groups. Here we focus on two of them that have a large impact on advancing big data research more broadly.

Concern 1:

How can successful multidisciplinary collaborations be facilitated given the potential misalignment in training, perspectives, and research goals?

The ramp-up time for interdisciplinary research can be substantial. There are discipline-specific language barriers that need to be overcome. Researchers from different disciplines need to teach each other about their domain, methods, etc. They all also need to have an interest in advancing research in other disciplines – too often domain experts are viewed as clients and computer scientists and statisticians are viewed as programmers and data janitors. In reality, they are all researchers who need to view each other's disciplines as equal – otherwise the research partnership is doomed from the start.

Recommendations:

1. Allow awards to include time for interdisciplinary training of team members during year one. In other words, require new collaborations to develop training lectures that can be shared on project websites as an outcome.
2. Require students across disciplines to be supported on interdisciplinary grants to promote interdisciplinary thought and training of students. This will help alleviate some training mismatches for the next generation of researchers.
3. Develop workshops that can be disseminated through Hubs and Spokes for successful interdisciplinary collaborations.

Concern 2:

How can we increase the availability of high quality data sets? For many of the challenging, societal-scale issues, clean, well-processed data do not exist. Collecting, transforming, labeling, and validating these data for further analyses is costly and time-consuming. However, these pre-processing steps are necessary to ensure high data quality and eventually, meaningful big data results.

Recommendations:

1. Support grants focusing on the development of pre-processing tools that can be shared and easily adapted for different domains.
2. Generate more universally accepted quality standards for big data so that researchers under-

Acknowledgments: This meeting was supported by NSF Award #1561908. The opinions and findings described in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

stand the limitations of the data without wasting time going through them.
3. Develop privacy-driven pre-processing tools that identify unique records or data features that need altering or should not be shared.
4. Have calls focused on data sharing and support infrastructure costs associated with sharing. Create and maintain benchmark databases that are publicly available for research.

3. DOMAIN-SPECIFIC RESEARCH DIRECTIONS AND DATA SETS

Here we present research directions (with focus on short-term, three-year window) and available data sets identified in the breakout sessions.

3.1 Education

There have been a number of recent successes in this domain, including degree planning software for identifying early warning of poor student performance, intelligent tutoring systems, and educational analytic reporting tools. Over the next three years, promising directions include:

- Develop a sharing environment that can be used to combine and run models on restricted data that are siloed across the industry. This infrastructure should allow researchers to provide analytics, test different methods, and understand outcomes across the combined data sets. The infrastructure should be developed with student privacy as a central design tenet.
- Develop partnerships among workforce development specialists, data scientists, and education specialists to model career paths and provide recommendation software for students and adults at different stages in their career.
- Create data sets, analytics, visualizations, and learning algorithms that analyze the learning life cycle – from how teachers teach different topics to student engagement to student learning outcomes to career outcomes.
- Develop clear usage guidelines related to the confidentiality, ethical uses, and data privacy.

While the number of available data sets is limited, a few projects have anonymized and released data for use by researchers. A few datasets exist that are related to student performance and assessment on different Massive Open Online Courses (MOOC) platforms – the 2010 KDD Cup Challenge data set that contained interaction records between students and a computer-aided intelligent tutoring system for learning algebra,[1] the 2015 KDD Cup Challenge data set that contained data about student interactions with the virtual learning environment (China's XuetangX MOOC platform),[2] and the Open University Learning Analytics dataset (OULAD) that contains anonymized student demographic and interaction data for over 30,000 students across 22 courses.[3] guidance, general learning styles, knowledge requirements for progressive learning, etc.

There are also data sets that can be used to explore research questions related to adaptive learning. Rossi and Gnawali released anonymized discussion threads from 60 Coursera MOOCs.[4] Papousek and colleagues have released a data set related to student learning of geography facts.[5] Finally, there is also a substantial literature that is being developed about learning analytics methodologies. SoLAR curates a data set containing these different research sources to support computational analyses,[6] and the AFEL Data Catalog contains a collection of nonuser-centric data sets for understanding different online and social learning contexts.[7]

3.2 Environment and Energy

Big data analytics has been successfully applied in the broad field of environment and energy, to study the air quality with a large amount of air quality sensors, for water resource management (e.g., water supply, water quality and quantity), and for smart cities (e.g., smart transportation, smart parking, smart buildings). Researchers are also studying the potential impact of climate change, environmental impact on food, building energy efficiency standards and policy, smart grid enabled by the smart meters, ecology and ecosystem management, as well as geophysics (e.g., oil and gas exploration and production, geothermal, contaminant transport, carbon sequestration, and ground water).

Over the next three years, promising directions for advancing the state of the art include:

- Develop better, large-scale visual analytics methods and tools for this domain.
- Develop hybrid analytics approaches that use cloud analytics for large, public data sets and local analytics for sensitive data sets.
- Develop approaches and tools for interfacing machine learning with scientific models.

[1] KDD Cup Challenge (2010): https://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp

[2] KDD Cup Challenge (2015): http://data-mining.philippe-fournier-viger.com/the-kddcup-2015-dataset-download-link/

[3] OULAD: https://www.nature.com/articles/sdata2017171

[4] Coursera MOOC discussion threads: https://github.com/elleros/courseraforums

[5] Adaptive learning: https://github.com/adaptive-learning/data-public

[6] SoLAR: https://solaresearch.org/initiatives/dataset/

[7] AFEL online and social learning: http://data.afel-project.eu/catalogue/learning-analytics-dataset-v1/

- Use available long-term data sets to demonstrate validity of different algorithms and models for understanding factors associated with consumption and needs, as well as environmental impacts.

Because much data in this domain is not linked to consumers or individuals, a number of sources exist for public data sets. Through the government's open data initiative,[8] researchers can get access to hundreds of agriculture-related databases and data sets including soil survey databases and maps, various food and crop databases, and local pollution data. DataRefuge has hundreds of climate, clean water, and pollution data sets.[9] The building performance database.[10] is the largest curated database of information containing energy-related characteristics of commercial and residential buildings. The TRY database is a global archive of curated plant traits.[11] Hundreds of long-term ecological research data sets (habitat, animal population, environmental event data, etc.) are curated as part of the LTER research network.[12] Finally, IRIS is a research project that manages access to global earth science data, including earthquake, atmospheric, infrasonic, and hydrological data.[13]

3.3 Health

Health is a very broad domain with application areas in precision medicine, epidemics, health care cost management, and medication therapy, to name a few. There have been a number of recent successes including over 90% of hospitals and clinics using electronic healthcare records (EHR) and the use of these data for medication surveillance, the use of mobile health for real-time interventions, and the use of analytics generated from wearable devices for improving chronic disease patient care.

Over the next three years, promising directions for advancing the state of the art include:

- Improve methodologies and scale of causal inference techniques. While we have seen a large number of advances in machine learning, for precision health, mechanistic understanding (of the relationship, interactions and contributions of different variables) is important.
- Develop methods that can be easily explained and interpreted by clinicians.
- Use EHR data for more extensive public health understanding.
- Use available data to improve patient diagnosis and identify precursors of illnesses earlier.
- As the number of mobile and wearable devices increases in this space, develop standardized ontologies for more rapid development of analytics-based healthcare outcomes.

Over the years, the federal government has helped fund a large number of studies that generated data. Many of these databases and data sets are accessible from the U.S. National Library of Medicine.[14] This site also links to other non-federal and state data repositories. Types of data sets include: Medicare provider utilization and payment data, health care outcome databases, inpatient hospital stays of children, healthcare claims data, CDC epidemiology data sets, emergency response data, veterans data, data on aging, substance abuse, etc. The National Library of Medicine also links to the HealthData.gov initiative that provides access to healthcare data sets from different Federal agencies. The Big Cities Health Coalition maintains aggregate health data from 28 large cities including data related to opioids, obesity, and tobacco.[15] Healthcare Cost and Utilization Project (HCUP) developed through a Federal-State-Industry partnership has the largest collection of longitudinal hospital care data in the United States.[16] Finally, a number of bioinformatics databases have large data sets, including GenBank, a data repository of known genetic sequences, from the National Center for Biotechnology Information (NCBI)[17] and UniProt[18].

3.4 Policy

Big data has a number of different roles to play with regards to policy. First, big data can be used as evidence to inform policy and decision making. These data can be used to improve accountability through generated policy. Big data algorithms and data collection methods are also candidates for regulation and new technology-related policy, e.g., privacy and ethics related to using big data for different types of inference. Some recent successes in this area include the use of satellite data by NASA to improve food security through spatio-temporal data analysis, improved response to the Nepalese earthquake using opaque building images collected by drones, and updated water management policies in the Chesapeake Bay based on climate change and pollution models.

[8] Open Data Initiative: https://www.data.gov
[9] DataRefuge: https://www.datarefuge.org/
[10] Building Performance: https://bpd.lbl.gov/
[11] TRY (curated plant traits): https://www.try-db.org
[12] LTER: https://portal.lternet.edu/nis/home.jsp
[13] IRIS: http://www.iris.edu/hq/
[14] National Library of Medicine: https://www.nlm.nih.gov/
[15] Big Cities Health Coalition: http://www.bigcitieshealth.org/city-data/
[16] HCUP: https://hcup-us.ahrq.gov/databases.jsp
[17] NCBI: https://www.ncbi.nlm.nih.gov/genbank/
[18] UniProt: http://www.uniprot.org/

Over the next three years, promising directions for advancing the state of the art include:

- Develop case studies and automated detectors of potential misuse of big data to support specific policy agendas – identifying these types of scenarios and educating policy makers and the public may help reduce this form of misuse.
- Incorporate explanations of error into analyses that use big data – develop standards and techniques for sharing levels of noise, bias, missing values, etc. to enable clearer communication of big data results accompanying policy recommendations.
- Make models interpretable by policy makers so that big data can be used more readily in evidence-based policy recommendations.

Data in this domain are more scattered and very issue-specific. For example, environment, energy, economic, finance and health are all examples of domains with data that may impact policy. General population statistics, demographics, and voting data sets can be found at the Census Bureau.[19] Some other policy issues that have available data sets include: urban policy from the Urban Center for Computation and Data (UrbanCCD),[20] women and public policy including political participation, health, work and family, and safety from the Institute for Women's Policy Research (IWPR),[21] immigration,[22] global health,[23][24] and income inequity by the Organization for Economic Co-operation and Development (OECD).[25] There are also a number of simulation data sets that can be used to influence future policy, including the SUMO simulation of urban mobility/traffic on roads.[26]

3.5 The Economy and Finance

Big data and big data analytics have been applied in both finance and economics. Hedge funds have been using them successfully as alternative data inputs, scraping 100s of millions of websites daily. Twitter data has been used to predict a number of economic indicators including unemployment with mixed success. Other areas of success include: market manipulation detection by searching for anomalies in daily and tick trading stocks, financial entity profile construction from public regulatory filings (SEC, FDIC), and identification of emerging risks.

Over the next three years, promising directions for advancing the state of the art include:

- Systematic generation of synthetic data sets that are designed as a "challenge" similar to the NIST challenge.[27]
- Manipulation detection in financial markets.
- Prediction of wider set of economic indicators.

There is no central repository for data in this domain. The NIST data challenge was mentioned above. Real time stock data is relatively easy to obtain online. Company SEC filings (over 11 million) can be obtained from the SEC search engine.[28] Data.gov hosts a diverse collection of datasets.[29] Finally, anonymized credit reports can be purchased through different credit companies.

4. FINAL THOUGHTS

This report has highlighted a number of immediate research directions and available data sets for five different application areas. One evident finding is that the impact of big data varies considerably depending on the domain. While important research directions exist across all the domains, this variability is partially due to the variability of curated data sets in different domains. To increase the pace of research innovation, more effort and funds need to be devoted to data curation and data plumbing. Even after data curation, much work is still needed to make big data and data science concepts more accessible to a broader community. Initiatives like the DataCore and workplace data science training are vital for broadening the community and integrating big data analysis techniques into research across domains. Finally, we as a community need to be honest about big data as a field – while we push the boundaries, we also need to explain the limitations, assumptions, and biases that may be present in different analyses and that may differ from what people in different disciplines are accustomed to. We need to pause and think about the ethical implications of using certain data sets and pause to make sure we are preserving privacy and promoting fairness when developing new methods.

5. REFERENCES

[1] NSF BDPI-2016 Workshop Report, 2016.

[2] L. Singh, D. Logston, S. Nusser, and H. Wactlar. NITRD BDSI-2015 Workshop Report: Spearheading Innovation in the Face of Massive Data, 2015.

[19] Census data: https://www.census.gov/data/datasets.All.html

[20] UrbanCCD: http://www.urbanccd.org/research-and-tools/

[21] IWRP: https://iwpr.org/

[22] EU Data Portal: https://data.europa.eu/euodp/data/dataset/L0q3araJ0g9Dk3TXZWkJg

[23] Global Health Data Exchange: http://ghdx.healthdata.org/

[24] World Health Org.: http://www.who.int/gho/database/en/

[25] OECD: http://www.oecd.org/social/income-distribution-database.htm

[26] SUMO: http://sumo.dlr.de/index.html

[27] NIST Financial Entity Identification and Information Integration: https://ir.nist.gov/dsfin/

[28] SEC Company Filings Search: https://www.sec.gov/edgar/searchedgar/companysearch.html

[29] https://www.data.gov/finance/

Dagstuhl Seminar on Big Stream Processing

Sherif Sakr
Uni. of Tartu, Estonia
sherif.sakr@ut.ee

Tilmann Rabl
TU Berlin, Germany
rabl@tu-berlin.de

Martin Hirzel
IBM Research
hirzel@us.ibm.com

Paris Carbone
KTH EECS, Sweden
parisc@kth.se

Martin Strohbach
AGT International, Germany
mstrohbach@agtinternational.com

ABSTRACT

Stream processing can generate insights from big data in real time as it is being produced. This paper reports findings from a 2017 seminar on big stream processing, focusing on applications, systems, and languages.

1. OVERVIEW

As the world gets more instrumented and connected, we are witnessing a flood of raw data generated, at high velocity, from different hardware (e.g., sensors) or software in the form of streams of data. Examples abound in several domains including financial markets, surveillance systems, manufacturing, smart cities, and scalable monitoring infrastructure. In these domains, there is a strong requirement to collect, process, and analyze big streams of data to extract valuable information, discover new insights in real-time, and detect emerging patterns and outliers. Since 2011 alone, several systems (e.g., SPL [13], Storm [19], Apex[1], Spark Streaming [20], Flink [7], Heron [16], and Beam [3]) have been introduced to tackle the real-time processing demands of big streaming data. However, there are several challenges and open problems that need to be addressed to improve the state-of-the-art and achieve further adoption of big stream processing technology [18].

This report is based on a seminar on "*Big Stream Processing Systems*" at Schloss Dagstuhl in Germany from 29 October to 3 November 2017[2], attended by 29 researchers from 13 countries. Participants came from different communities including systems, query languages, benchmarking, stream mining, and semantic stream processing. A benefit of this seminar was the opportunity for scholars from different communities to get exposure to each other and get freely engaged in direct and interactive discussions. The program consisted of tutorials on the main topics of the seminar, lightning talks by participants on their research, and two working groups dedicated to a deeper investigation. The first working group focused on applications and system of big stream processing while the second group focused on streaming languages. This report presents highlights and outcomes.

2. TUTORIALS

The tutorials of the seminar aimed at sharing knowledge between attendees from different communities, offering perspectives for group discussions.

2.1 IoT Stream Processing Applications

This tutorial analyzed IoT applications from two domains: sports and entertainment as well as Industry 4.0. The application examples are based on commercial deployments using AGT International's[3] Internet of Things Analytics (IoTA) platform.

Sports and Entertainment. The example applications of this domain provide real-time narratives about highlights during a live event. This way, it is not necessary to watch the whole event, but one can be notified in real-time about such highlights based on insights derived from sensor data. For instance, in *basketball*, sensors that have been successfully used in commercial deployments[4] include smart shirts worn by players, microphones deployed to monitor the audience, cameras, and wristbands. Data from these sensors in combination with play-by-play data can be used to recognize behavior, emotions, activities, actions, pressure, and other physical aspects of the game. These insights are related to players, teams, fans, and family preferably in the form of semantic data streams. Semantic data access decouples applications from data providers and enables domain experts to better work with the data, e.g., for generating content and distributing it via social media.

Another example is *mixed martial arts*[5], where cameras and sensors embedded in floors and fight-

[1] https://apex.apache.org/

[2] www.dagstuhl.de/en/program/calendar/semhp/?semnr=17441

[3] http://www.agtinternational.com

[4] https://t.co/ZkQjQwXw13

[5] https://youtu.be/vataVq9gY_o

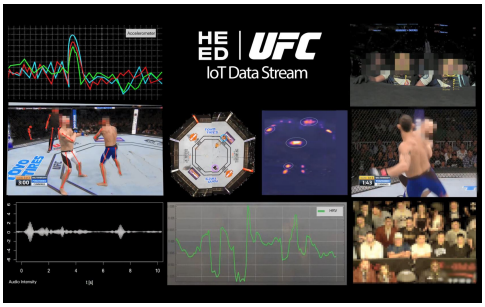


Figure 1: Sample IoT data streams in mixed martial arts.

ers' gloves[6] offer insights including punch strength and stress levels of each fighter (Figure 1). In this example, it is important that insights can be delivered in real-time without noticeable delay compared to a broadcast of the fight.

In *professional bull riding*, sensors are attached to riders and bulls and used to quantify the bull's and rider's performance[7]. As this information is, among other things, used for automatic scoring, it is of particular importance that analytic results are available as soon as the ride is finished. Similarly, a range of wearable sensors are used for creating event highlights for participants at *mass sport events* such as the Color Run[8]. The CPaaS.io project[9] uses action cameras and fitness bands to automatically detect event highlights based on the the runner's activity, emotions, dance energy levels, and many more metrics. In this application, real-time aspects include scenarios in which event highlights are being directly sent to friends of the participants.

Industry 4.0. For this domain, the tutorial presented applications around *predicting energy peaks* and *predictive maintenance*. In principle, predicting energy peaks can help in reducing energy costs as electricity bills of industrial consumers contain a pricing component that incurs higher charges for higher peaks of electrical load. For small-to-medium enterprises, avoiding such peak load events can lead to significant savings[10]. This can be achieved by predicting expected peaks, e.g., up to 30 minutes ahead of time and taking precautionary measures such as temporarily switching off high energy consumers such as air conditioning.

For *predictive maintenance*, the tutorial presented an application for detecting anomalous machine states to reduce maintenance costs. For instance, in injection molding machines, a sudden high energy consumption may indicate that an injection nozzle is jammed and checking the machine may avoid further damage. The tutorial reported about the DEBS Grand Challenge 2017 [11] that has been designed to objectively measure some of these requirements using pre-defined machine learning algorithms and RDF streaming data. The main KPI for the challenge was latency. The original data set has been provided by Weidmüller[11]. For reasons of confidentiality, the organizers provided a mimicked data set[12]. The systems under test were evaluated using the HOBBIT benchmarking platform[13] that ensured the objectivity of quantifying the performance of distributed stream processing pipelines. Overall, 7 out of 14 participating teams in the challenge passed the correctness test. The fastest system [4] achieved an average latency of about 39ms. The DEBS Grand Challenge 2017 benchmark is openly available as part of the HOBBIT platform.

2.2 Big Stream Processing Systems

This tutorial started by identifying the most differentiating characteristic of scalable data stream processing systems, which is the notion of data as a continuous, possibly infinite resource instead of "facts and statistics organized and collected together for future reference or analysis"[14]. In fact, data stream processing systems broaden the context from retrospective data analysis to continuous, unbounded processing coupled with scalable and persistent application state. Various forms of stream processing have been employed in the past within their respective domains, such as network-centric processing on byte streams, functional (e.g., monads) and actor programming, complex event processing, and database materialized views. Besides, *stream management* has been an active research field for many years [2, 5, 8]. Nonetheless, several of these ideas have only just recently been put together in a consistent manner to compose a stack centered around the notion of data as an unbounded partitioned stream of records (Figure 2). Most importantly, stream processing did not restrict but complemented existing scalable processing models (e.g., MapReduce [10]) with persistent partitioned state, time domains, and flexible scoping via windows. The general programming stack addresses storage, compute, and domain-specific library support.

Stream Storage. Data dissemination from consumers to producers is a problem that has been revisited multiple times with different assumptions and needs in mind. In the context of data streaming, direct communication (e.g., TCP channels) was not an option despite low-latency requirements, since it required application ingestion to be actively in

[6] http://bit.ly/2D4lCqD
[7] http://bit.ly/2CXpc2g
[8] https://thecolorrun.com/
[9] http://www.cpaas.io
[10] http://bit.ly/2DjhvUh
[11] http://www.weidmueller.de
[12] https://hobbit.iminds.be/dataset/weidmuller
[13] http://bit.ly/2muMNkY
[14] Google Dictionary

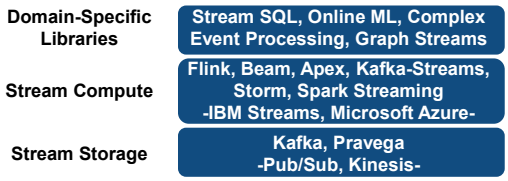


Figure 2: The Stack of Scalable Stream Processing

sync with data creation while also lacking the transparency and durability of today's cloud computing ecosystem. Furthermore, message brokers (e.g., RabbitMQ, JMS) were insufficient for the needs of supporting multiple applications and configurations (i.e., task parallelism). Thus, a class of open-source stream storage systems based on *partitioned replicated logs* was introduced, led by Apache Kafka [15] and more recently Pravega[15] as well as proprietary cloud services such as Amazon Kinesis[16]. Partitioned replicated logs provide high sequential read and write throughput by exploiting copy-on-write and strict data-parallel access by distinct consumers. Furthermore, they perform offset-based bookkeeping of data access for the purposes of data reprocessing, reconfiguration, and roll-backs, among others. Finally, more effort has been devoted to supporting transactional logging and repartitioning, allowing for seamless integration with modern stream compute systems.

Stream Compute. We further divide compute into *programming models* and *runtime engines*. In terms of programming model support, there has been a shift from purely event-based, compositional models (e.g., Apache Storm [19]) to more declarative representations [3, 7, 20]. Currently, most standard APIs are fluid, functional, and allow declaring relational transformations (e.g., joins, filters) while providing first-class support for persistent partitioned state, stream windows, and event-time progress using watermarks. The latter allowed application logic to incorporate timers that operate consistently on different time domains (e.g., origin-time), thus allowing out-of-order processing [17], a concept popularized e.g. by Beam [3].

With respect to runtime engines, we observe converging commonalities such as a dataflow execution model, explicit locally embedded state (using log-compaction trees [1]), and asynchronous snapshots for fault tolerance and reconfiguration [6, 14]. Spark Streaming [20], as a special case, emulates streaming by slicing computation into recurring batch jobs, yet, it currently makes use of locally embedded state and there are plans to adopt a continuous processing runtime for low-latency data streaming.

[15] http://pravega.io/ [16] https://aws.amazon.com/kinesis/

2.3 Stream Processing Languages

This tutorial provided an overview of several styles of stream processing languages: streaming SQL, synchronous dataflow, big-data streaming, complex event processing, and end-user programming. After the Dagstuhl seminar, some of the participants wrote a survey paper inspired by this tutorial [12]. For space reasons, rather than describing the tutorial here, we refer interested readers to that paper.

3. WORKING GROUPS

During the seminar, two separate working groups formed to discuss current challenges in streaming applications and systems and in streaming languages.

3.1 Applications and Systems

In this working group, participants discussed characteristics and open challenges of stream processing systems, focusing on state management, transactions, and pushing computation to the edge.

State Management. Modern streaming systems are stateful, which means they can remember the state of the stream to some extent. A simple example is a counting operator that counts the number of elements seen so far. While even a simple state like this poses several challenges in streaming setups (such as fault tolerance and consistency), many use cases require more advanced state management. An example is the combination of streaming and batch data, e.g., when combining the history of a user with their current activity or when finding matching advertisement campaigns with current activity; a popular example of such a setup is modeled in the Yahoo! Streaming Benchmark [9]. Today, most setups deal with such challenges by combining different systems (e.g., a key value store for state and a streaming system for processing). However, it is desirable to have both in a single system for consistency and manageability reasons.

State can be considered the equivalent of a table in a database system [5]. As a result, several high-level operations can be identified: conversion of streams to tables (e.g., storing a stream), conversion of tables to streams (e.g., scanning a table), as well operations only on tables or streams (joins, filters, etc.). The management of state opens the design space between existing stream processing systems and database systems, which has only been partially explored by current systems. In contrast to database systems, stream systems typically operate in a reactive manner, i.e., they have no control over the incoming data stream, specifically, they do not control and define the consistency and order

semantics in the stream. This requires advanced notions of time and order as for example specified for streams in the dataflow model [3].

Transactions. A further discussion topic was transactions in stream processing systems. The main difference between traditional database transactions and stream processing transactions is that in databases the computation moves and data stays (in the system), whereas in stream processing systems the computation stays and the data moves to the computation (and out again). Considering state management, the form of transactions as applied in databases can also be used in a stream processing system, if the state is managed in a transactional way. However, the operations on streams themselves can be transactional and then we can differentiate between single-tuple transactions and multi-tuple transactions (possibly accessing multiple keys in a partitioned operator state space). Multi-tuple transactions can only commit when all tuples are consumed. The tuples then have to traverse the whole operator graph or at least the transactional subgraph. The semantics of transactions on streams is currently still an open field of research.

Pushing computation to the edge of a network enables stream processing to be highly distributed and decentralized. This is very useful when preprocessing or filtering can be done without a centralized view of the data, especially in setups with high communication cost or slow connections (e.g., mobile connections): it makes sense to not send all data to a central server, but distribute the computation. A logical first step is filtering, but aggregations and even more complex operations can be pushed to the edge, if possible. Many modern scenarios prohibit centralized data storage, which further encourages distributed setups with early aggregations.

3.2 Languages and Abstractions

Based on the corresponding tutorial (Section 2.3), this working group identified three challenges faced by streaming languages: input variety, output veracity, and adoption of streaming languages. After the seminar, some of the participants continued the discussion and incorporated it in the same survey paper that was inspired by the tutorial [12].

4. CONCLUSION

The tutorials, presentations, dialogs, and working groups at the "*Big Stream Processing Systems*" seminar provided an overview of current developments and emerging issues. This report highlighted the main outcomes of the seminar. The discussions of the seminar have also revealed several open challenges and interesting future research directions including (1) semantic data access and reasoning, (2) defining a standardized query language for streaming applications, (3) providing better support for machine learning including a wide range of data science programming languages (Python, R, Julia), and (4) improving optimizations for low latencies and short-lived stream processing pipelines.

Acknowledgements

We thank the seminar participants and the Dagstuhl staff. The work presented in this paper has partly been funded by the European Regional Development Fund via the Mobilitas Pluss program (grant MOBTT75), by the H2020 projects CPaas.io, HOBBIT, Streamline, and Proteus (grant agreement numbers 688227, 723076, 688191, and 687691), and by the German Ministry for Education and Research as Berlin Big Data Center (funding mark 01IS14013A).

5. REFERENCES

[1] Rocksdb. http://rocksdb.org/, 2018.
[2] D. J. Abadi et al. Aurora: A new model and architecture for data stream management. *VLDB J.*, 12(2), 2003.
[3] T. Akidau et al. The dataflow model: A practical approach to balancing correctness, latency, and cost in massive-scale, unbounded, out-of-order data processing. In *VLDB*, pages 1792–1803, 2015.
[4] C. Amariei, P. Diac, and E. Onica. Optimized stage processing for anomaly detection on numerical data streams: Grand challenge. In *DEBS*, 2017.
[5] A. Arasu, S. Babu, and J. Widom. The CQL continuous query language: Semantic foundations and query execution. *VLDB J.*, 15(2):121–142, 2006.
[6] P. Carbone, S. Ewen, G. Fora, S. Haridi, S. Richter, and K. Tzoumas. State management in Apache Flink: Consistent stateful distributed stream processing. In *VLDB*, 2017.
[7] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas. Apache Flink: Stream and batch processing in a single engine. *IEEE Database Engineering Bulletin*, 36(4):28–38, 2015.
[8] S. Chandrasekaran et al. TelegraphCQ: Continuous dataflow processing for an uncertain world. In *CIDR*, pages 668–668, 2003.
[9] S. Chintapalli et al. Benchmarking streaming computation engines: Storm, Flink and Spark Streaming. In *IPDPSW*, 2016.
[10] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *CACM*, 51(1), 2008.
[11] V. Gulisano, Z. Jerzak, R. Katerinenko, M. Strohbach, and H. Ziekow. The DEBS 2017 grand challenge. In *DEBS*, 2017.
[12] M. Hirzel, G. Baudart, A. Bonifati, E. Della Valle, S. Sakr, and A. Vlachou. Stream processing languages in the big data era. *SIGMOD Record*, 2018.
[13] M. Hirzel, S. Schneider, and B. Gedik. SPL: An extensible language for distributed stream processing. *Transactions on Programming Languages and Systems (TOPLAS)*, 39(1):5:1–5:39, 2017.
[14] G. Jacques-Silva et al. Consistent regions: Guaranteed tuple processing in ibm streams. In *VLDB*, 2016.
[15] J. Kreps, N. Narkhede, J. Rao, et al. Kafka: A distributed messaging system for log processing. NetDB, 2011.
[16] S. Kulkarni et al. Twitter Heron: Stream processing at scale. In *SIGMOD*, pages 239–250, 2015.
[17] J. Li et al. Out-of-order processing: A new architecture for high-performance stream systems. In *VLDB*, 2008.
[18] S. Sakr. *Big Data 2.0 Processing Systems: A Survey.* Springer, 2016.
[19] A. Toshniwal et al. Storm @Twitter. In *SIGMOD*, 2014.
[20] M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica. Discretized streams: Fault-tolerant streaming computation at scale. In *SOSP*, 2013.

CALL FOR NOMINATIONS

ACM PODS 2019 ALBERTO O. MENDELZON TEST-OF-TIME AWARD

Nominations are solicited for the PODS 2019 Test-of-Time Award. The award will recognize a paper or a small number of papers published in the PODS 2009 proceedings that had the most impact in terms of research, methodology, or transfer to practice over the intervening decade. All papers are nominated by default, but the committee welcomes input from our community. Please feel free to nominate a paper if you think it has had great impact, even if you have not thoroughly compared it to the other eligible papers. The usual conflict of interest rules apply.

The PODS 2019 Test-of-Time Award Committee consists of Jianwen Su, Dirk Van Gucht and Victor Vianu (chair). Please email your nominations to Victor (vianu@cs.ucsd.edu) with subject line "PODS 2019 ToT Award nomination" together with a brief justification. Please send your nominations no later than December 15, 2018. Nominations are confidential and will only be shared among the committee members.

The PODS Test-of-Time Award for 2019 will be presented during the SIGMOD/PODS Joint Conference held June 30 - July 5, 2019 in Amsterdam, The Netherlands.

The PODS 2009 papers can be found at https://dl.acm.org/citation.cfm?id=1559795

Call for Papers: 23rd International Conference on Database Theory (ICDT 2020)

Copenhagen, Denmark, March-April 2020 (exact dates TBA)

About ICDT

ICDT is an international conferences series that addresses the principles and theory of data management. Since 2009, it is annually and jointly held with EDBT, the international conference on extending database technology. For general information, see `https://databasetheory.org/icdt-pages`.

Topics of Interest

Every topic related to the principles of data management is relevant to ICDT. Particularly welcome are contributions that connect data management to theoretical computer science, and those that connect database theory and database practice. Examples of relevant topics include:

- Data mining, information extraction, information retrieval, and machine learning for databases
- Data models, design, structures, semantics, query languages, and algorithms for data management
- Distributed databases, cloud computing
- Databases and knowledge representation
- Graph databases, Web data, and Web services
- Data streams and sketching
- Data-centric process management and workflows
- Data and knowledge integration and exchange, data provenance, views, and data warehouses
- Domain-specific databases (multimedia etc)
- Data privacy, security, recovery

Reach Out Track

With its Reach Out Track, ICDT strives to broaden its scope, calling for novel formal frameworks and directions for database theory and connections between principles of data management and other communities such as Database Systems, Artificial Intelligence, Knowledge Representation, Machine Learning, Programming Languages, Distributed Computing, and Operating Systems.

Submission Cycles and Dates

ICDT has two submission cycles:

1st cycle abstract due:	March 27, 2019
Full paper due:	April 3, 2019
Notification:	May 29, 2019
2nd cycle abstract due:	September 15, 2019
Full paper due:	September 23, 2019
Notification:	December 5, 2019

Program Committee

Marcelo Arenas, Michael Benedikt, Christoph Berkholz, Angela Bonifati, Pierre Bourhis, James Cheney, Graham Cormode, Victor Dalmau, Claire David, Floris Geerts, Bas Ketsman, Daniel Kifer, Leonid Libkin, Carsten Lutz (chair), Sebatian Maneth, Filip Murlak, Reinhard Pichler, Andreas Pieris, Sebastian Rudolph, Thomas Schwentick, Uri Stemmer, Domagoj Vrgoc and Frank Wolter.

Submission Instructions

The proceedings will appear in the Leibniz International Proceedings in Informatics (LIPIcs) series, based at Schloss Dagstuhl. This guarantees that the proceedings will be available under the gold open access model, online and free of charge. Submissions will be via EasyChair at `https://easychair.org/conferences/?conf=icdt2020`. Papers must not exceed 15 pages in length (excluding references), both for regular submissions and for the Reach Out Track. For Reachout Track Submissions, it is reasonable to be shorter in length.

Awards

An award will be given to the Best Paper and to the Best Newcomer Paper where 'newcomer' refers to the field of database theory. The latter award will preferentially be given to a paper authored only by students and in that case be called Best Student-Paper Award.

For more details, see `http://www.informatik.uni-bremen.de/~clu/icdt2020cfp.txt`.