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, Yannis Ioannidis, Christian Jensen, and Jan Van den Bussche.

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:
Jan Van den Bussche (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:
Surajit Chaudhuri (Chair), David Dewitt, Martin Kersten, Maurizio Lenzerini, Jennifer Widom

Jim Gray Doctoral Dissertation Award Committee:
Ashraf Aboulnaga (co-Chair), Chris Jermaine (co-Chair), Paris Koutris, Feifei Li, Qiong Luo, Ioana Manolescu, Lucian Popa, Renée Miller

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)

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)

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)

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** *Winner*: Peter Bailis. *Honorable Mention*: Immanuel Trummer

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

[Last updated : June 30, 2017]

Editor's Notes

Welcome to the December 2017 issue of the ACM SIGMOD Record!

This issue starts with the Database Principles column featuring an article by Pierre Senellart on "Provenance and Probabilities in Relational Databases". This article describes different provenance formalisms, from Boolean provenance to provenance semirings and beyond, which can be used to answer a variety of questions regarding the output of a query. It also discusses representation systems for data provenance, circuits in particular, with a focus on its implementation, and how provenance is practically used for query evaluation in probabilistic databases. The article concludes that practical implementation of provenance management is very much possible because it introduces a relatively low overhead.

The Vision column features an article by Pitoura et al. on "Measuring Bias in Online Information". The work is motivated by the observation that as we live in an information age today, the majority of our diverse information needs are satisfied online by search engines, social networks and media, e-shops, and other online information providers. Bias in online information has recently become a pressing issue, with search engines, social networks and recommendation services being accused of exhibiting some form of bias. In this vision paper, the authors make the case for a systematic approach towards measuring bias, with a focus on formal measures for quantifying the various types of bias and the system components necessary for realizing them. The article closes by highlighting the related research challenges and open problems.

The Systems and Prototypes column features an article by Spyropoulos and Kotidis on the Digree system for scalable graph analytics, an issue critical to many application domains such as social networks and electronic commerce. The Digree system enables distributed execution of graph pattern matching queries in a cloud of interconnected graph databases. The system decomposes a graph query into independent sub-patterns, processes them in parallel on distributed graph databases, and finally synthesizes the results at a master node. By comparing to Graphframes, a package for Apache Spark on 18 VMs, Digree is shown to provide superior performance on real world datasets.

The Distinguished Profiles column features Dan Suciu, Professor at the University of Washington. Dan has two Test of Time Awards from PODS as well as Best Paper Awards from SIGMOD and ICDT. In this interview, Dan talked about his research projects and major research results, from XML to privacy to probabilistic data to data markets finally to scalable query processing. During the interview, Dan also discussed his favorite style of research, "a good combination of both theory and practice," because he believes that "the most difficult theory questions are those that are grounded in practice and that the most interesting systems are those that have a strong theoretical component."

The Open Forum column features an article by Sadiq et al. to call to action for promoting empiricism in data quality research. The authors identify two inter-related dimensions of empiricism that help locate the sweet-spot for empiricism in advancing data quality research and practice. These are the type of metric and the scope of method. A third aspect, namely, the nature of the data, exposes a data continuum that defines the setting in which the data quality metrics and methods can be evaluated. The article further presents the various ways in which the dimensions of empiricism can be positioned, thus providing a lens through which the role of empiricism in data quality re-

search can be studied. In order to gain a deeper insight into each of these positions, the authors reached out to thought leaders in data quality research to help elaborate on the motivation and rationale, key approaches, and possible challenges against each position. The viewpoints presented in this article are extracted from a series of interviews conducted with the experts and are supplemented with a review of relevant literature.

The Reports Column features two articles. The first report summarizes the presentations and discussions of the fourth workshop on Algorithms and Systems for MapReduce and Beyond (BeyondMR'17) held in conjunction with the 2017 SIGMOD/PODS conference. The goal of the workshop was to bring together researchers and practitioners to explore algorithms, computational models, languages and interfaces for systems that provide large-scale parallelization and fault tolerance. The program featured two invited talks, the first on current and future development in big data processing by Matei Zaharia from Databricks and Stanford University, and the second on computational models for big data analytics algorithms by Ke Yi from the Hong Kong University of Science and Technology. The second report summarizes a tutorial on "Commonsense Knowledge in Machine Intelligence" presented at the ACM Conference on Information and Knowledge Management (CIKM) in November 2017. This article is motivated by the viewpoint that the future of computing depends on our ability to exploit big data on the Web to enhance intelligent systems, that is, to endow machines with commonsense knowledge. The overview of the state of the art on Commonsense Knowledge (CSK) in Machine Intelligence provides insights into CSK acquisition, CSK in natural language processing, and applications of CSK, as well as the set of open issues.

On behalf of the SIGMOD Record Editorial board, I hope that you enjoy reading the December 2017 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

December 2017

Past SIGMOD Record Editors:

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

Provenance and Probabilities in Relational Databases: From Theory to Practice

Pierre Senellart
DI ENS, ENS, CNRS, PSL Research University
& Inria Paris
& LTCI, Télécom ParisTech
Paris, France
pierre@senellart.com

ABSTRACT

We review the basics of data provenance in relational databases. We describe different provenance formalisms, from Boolean provenance to provenance semirings and beyond, that can be used for a wide variety of purposes, to obtain additional information on the output of a query. We discuss representation systems for data provenance, circuits in particular, with a focus on practical implementation. Finally, we explain how provenance is practically used for probabilistic query evaluation in probabilistic databases.

1. INTRODUCTION

The central task in data management is *query evaluation*: given a database instance and a query, compute the results of that query on that instance. But what if we want something more than just the query result? We might want to know:

- *why* this specific result was obtained;
- *where* values in the result come from;
- *how* the result was produced from the query;
- how the result would change if some of the input *tuples* were *missing*;
- *how many times* each query result was obtained;
- what *probability* the result has, given a probability distribution on the input data;
- what *minimal security clearance* is needed to see the result, given some security information on the input data;
- what the *most economical way* to obtain this result is, in terms of number of data accesses.

All these questions, and more, can be answered using the tool of *data provenance* [11,12], which is some additional bookkeeping information maintained during query evaluation, that allows answering a large number of such meta-questions on the output of a query. The precise nature and form this provenance information should take depends on the question we want to answer, and on the query language we consider.

We focus in this paper on the setting of relational databases, though provenance and its applications apply as well and are equally important in other settings, such as scientific workflows [19], knowledge graphs [30], or hierarchical data [11].

There has been a large amount of work on the foundations of data provenance in relational database systems: early definitions and implementations of data provenance (called *lineage* at the time) for specific applications [13,48]; the influential framework of where- and why-provenance introduced by Buneman, Khanna, and Tan [11]; the seminal paper on provenance semirings [27] by Green, Karvounarakis, and Tannen; and further extensions thereof [7, 8, 22, 24]. Provenance has also been a particularly useful tool in the area of probabilistic databases [3, 23, 32], where the use of provenance is dubbed the *intensional approach* to probabilistic query evaluation.

The goal of this paper is to review some of the most important definitions of provenance, unifying them in a single framework as much as possible, and to address some of the concrete issues that arise in building a modern provenance-aware database management, and in implementing the intensional approach to probabilistic query evaluation.

Example 1. We will use throughout this paper a very simple running example of a single-table database, in Table 1, that contains information on the personal of a (fictitious) intelligence agency, to illustrate various aspects of data provenance. Note in Table 1 the t_i annotation on each tuple. These will be used in the following as *provenance tokens* associated with the tuple, which means that they will contain the elementary provenance information associated to the tuple. You can think of them for now as simple tuple identifiers.

We will consider the following two example queries

Table 1: Table Personal for the personal of an intelligence agency, used as a running example

id	name	position	city	classification	
1	John	Director	New York	unclassified	t_1
2	Paul	Janitor	New York	restricted	t_2
3	Dave	Analyst	Paris	confidential	t_3
4	Ellen	Field agent	Berlin	secret	t_4
5	Magdalen	Double agent	Paris	top_secret	t_5
6	Nancy	HR	Paris	restricted	t_6
7	Susan	Analyst	Berlin	secret	t_7

on this database. The first query, Q_1, asks for the cities referenced in the Personal table which host at least two different employees of the agency:

```
SELECT DISTINCT P1.city
FROM Personal P1, Personal P2
WHERE P1.city = P2.city AND P1.id < P2.id
```

Obviously, the answer to this query, disregarding provenance, is a single-attribute table, containing the three cities New York, Paris, and Berlin. Our second query, Q_2, asks for the cities that host exactly one employee of the agency:

```
  SELECT DISTINCT city FROM Personal
EXCEPT
  SELECT DISTINCT P1.city
  FROM Personal P1, Personal P2
  WHERE P1.city = P2.city AND P1.id < P2.id
```

Its output, once again disregarding provenance, is the empty table. □

The paper is organized as follows: in Section 2, we first introduce the simple setting of *Boolean provenance* that is in particular used in probabilistic databases and has the advantage of being definable for an arbitrary query language. We move in Section 3 to *semiring provenance*, which captures more information than Boolean provenance but is defined for a specific, monotone, query language. We explore formalisms that go beyond semiring provenance in Section 4. In Section 5, we address the concrete problem of which formalism to use to represent data provenance, and settle on circuits as compact formalism. We explain in Section 6 how probabilistic query evaluation in probabilistic databases can be solved using provenance, and which tools can be used to do this efficiently.

2. BOOLEAN PROVENANCE

Boolean provenance is one of the simplest forms of provenance, while having a major conceptual advantage: it can be defined independently of a specific query language. The notion of Boolean provenance is implicit in the work of Imieliński and Lipski on conditional tables [31], although this predates the notion of provenance itself and the query language was limited to the relational algebra. The term *Boolean provenance* was introduced specifically in the setting of provenance semirings [26], though this was restricted to the monotone case. The generalization to non-monotone queries we give below is straightforward, and was made, e.g., in [3].

We fix a finite set $X = \{ x_1, \ldots, x_n \}$, the elements of which we call *Boolean events* (i.e., variables that can be either $\top$ or $\bot$).

As in [31], we let provenance tokens (the annotations attached to tuples of the input databases) be Boolean functions over X, that is, functions of the form $\varphi : (X \to \{\top, \bot\}) \to \{\top, \bot\}$. They are interpreted under a *possible-world semantics*: every valuation $\nu : X \to \{\top, \bot\}$ denotes a *possible world* of the database; in this possible world, a tuple with annotation φ exists if and only if $\varphi(\nu) = \top$. For a given database D, we denote $\nu(D)$ the set of tuples t with annotation φ_t such that $\varphi_t(\nu) = \top$. It is a subset of the database D.

Example 2. Consider again Table 1 and assume that, for each i, t_i is the indicator function of a distinct Boolean event x_i, i.e., the function that maps a valuation ν to $\top$ if and only if $\nu(x_i) = \top$. Then the set of possible worlds of Personal is the set of all subrelations of Personal. For instance, if ν maps x_1, x_3, x_5, and x_7 to $\top$ and all other variables to $\bot$, then ν(Personal) is the subrelation of Personal where only the tuples of id 1, 3, 5, and 7 survive. □

Let Q be an arbitrary query, i.e., a function that takes as input a finite relational database over a fixed database schema, and produces as output a finite relation over a fixed relational schema. Then the *Boolean provenance* of Q over a database D, denoted $\text{prov}_{Q,D}$, is a function that maps a tuple t of the

output relational schema to the Boolean function that maps a valuation ν to $\top$ if $t \in Q(\nu(D))$, and to $\bot$ otherwise. In other words, given provenance annotations on input database tuples, we obtain as output of a query a new database, namely,

$$\bigcup_{\nu:X\to\{\top,\bot\}} Q(\nu(D)),$$

with new provenance annotations on each tuple t, namely $\text{prov}_{Q,D}(t)$.

Note that if Q is *monotone*, i.e., if $D \subseteq D' \Rightarrow Q(D) \subseteq Q(D')$, then $\bigcup_{\nu:X\to\{\top,\bot\}} Q(\nu(D)) \subseteq Q(D)$, but this is not true for arbitrary queries.

Example 3. We will denote in this example Boolean functions using propositional formulas. See Section 5 for an alternate representation. To simplify, we assume that, once again, every t_i is an indicator function of a different Boolean event, and we write simply t_i for this event instead of x_i.

One can check that the Boolean provenances of Q_1 and Q_2 on Personal are, respectively:

city	
New York	$t_1 \wedge t_2$
Paris	$(t_3 \wedge t_5) \vee (t_3 \wedge t_6) \vee (t_5 \wedge t_6)$
Berlin	$t_4 \wedge t_7$

and:

city	
New York	$(t_1 \wedge \neg t_2) \vee (t_2 \wedge \neg t_1)$
Paris	$(t_3 \wedge \neg(t_5 \vee t_6)) \vee (t_5 \wedge \neg(t_3 \vee t_6)) \vee (t_6 \wedge \neg(t_3 \vee t_5))$
Berlin	$(t_4 \wedge \neg t_7) \vee (t_7 \wedge \neg t_4)$

□

One of the major applications of Boolean provenance is query evaluation in probabilistic databases. Assume that each Boolean event x_i comes with an independent probability $\Pr(x_i)$ of being true. Then we can define the probability of a valuation $\nu : X \to \{\top, \bot\}$ as:

$$\Pr(\nu) = \prod_{\nu(x_i)=\top} \Pr(x_i) \prod_{\nu(x_i)=\bot} (1 - \Pr(x_i)).$$

From there, it is natural to define, for any given Boolean function φ over X:

$$\Pr(\varphi) = \sum_{\substack{\nu:X\to\{\top,\bot\} \\ \varphi(\nu)=\top}} \Pr(\nu).$$

In particular, this defines a probability distribution $\Pr(\text{prov}_{Q,D}(t))$ on provenance annotations of output tuples given a probability distribution on the provenance annotations of input tuples. This observation was first made by Green and Tannen in [28]. When t_i's are indicator functions, one gets the simple model of *tuple-independent databases* [14, 23, 36] that has been widely studied.

Example 4. Assume again every t_i is an indicator function of a different Boolean function, and comes with an independent probability $\Pr(t_i)$ of being true.

Then $\Pr(\text{New York} \in Q_1(\text{Personal})) = \Pr(t_1 \wedge t_2) = \Pr(t_1) \times \Pr(t_2)$.

Note that we were able to compute the probability this way because the Boolean formula was simple enough. We discuss in Section 6 options when Boolean functions are more complex. □

Though Boolean provenance can be formally defined for any query, what we need in practice is efficient algorithms for computing the provenance of a query in a given query language. We also want to capture more with provenance than what Boolean provenance can do. This is what provenance semirings, and extensions thereof, offer.

3. SEMIRING PROVENANCE

Provenance semirings have been introduced in [27] as a formalism for data provenance that has been shown [34] to cover and generalize, using a clean mathematical framework, previous formalisms such as *why-provenance* [11], lineages used in view maintenance [13], or the lineage used by the Trio uncertain management system [9].

A *semiring* $(K, \mathbb{0}, \mathbb{1}, \oplus, \otimes)$ is a set K with distinguished elements $\mathbb{0}$ and $\mathbb{1}$, along with two binary operators:

- $\oplus$, an associative and commutative operator, with identity $\mathbb{0}$;
- $\otimes$, an associative and commutative[1] operator, with identity $\mathbb{1}$.

We further require $\otimes$ to distribute over $\oplus$, and $\mathbb{0}$ to be annihilating for $\otimes$.

Examples of semirings include [27, 28, 34]:

- $(\mathbb{N}, 0, 1, +, \times)$: the *counting* semiring;
- $(\{\bot, \top\}, \bot, \top, \vee, \wedge)$: the *Boolean* semiring;
- $(\{\text{unclassified}, \text{restricted}, \text{confidential}, \text{secret}, \text{top secret}\}, \text{top secret}, \text{unclassified}, \min, \max)$: the *security* semiring;
- $(\mathbb{N} \cup \{\infty\}, \infty, 0, \min, +)$: the *tropical* semiring;

[1]It is almost always required in the literature [27, 34] that the semiring be commutative, which means that $\otimes$ must be commutative. Note that this is only necessary if the cross product operator of the relational algebra is assumed to be commutative, which is the case in the *named* perspective, but not in the *unnamed* one [1]. This assumption has some technical impact, e.g., on the universality of $\mathbb{N}[X]$, but is actually not critical to implement provenance support.

- ($\{$positive Boolean funct. over $X\}, \bot, \top, \vee, \wedge$): the semiring of *positive Boolean functions* over X;
- $(\mathbb{N}[X], 0, 1, +, \times)$: the semiring of *integer polynomials* with variables in X (also called *how*-semiring or *universal* semiring, see further);
- $(\mathcal{P}(\mathcal{P}(X)), \emptyset, \{\emptyset\}, \cup, \Cup)$: *why*-semiring over X ($A \Cup B \coloneqq \{a \cup b \mid a \in A, b \in B\}$).

Now, given a fixed semiring $(K, \mathbb{0}, \mathbb{1}, \oplus, \otimes)$, semiring provenance works as follows: we assume provenance tokens are all in K. We consider a query Q from the *positive relational algebra* [1] (selection, projection, renaming, cross product, union). We define a semantics for the provenance of a tuple $t \in Q(D)$ inductively on the structure of Q, informally as follows (formal definitions can be found in [27]):

- selection and renaming do not affect provenance annotations;
- in the set semantics, the provenance annotations of tuples that are identical after projection are $\oplus$-ed; in the bag semantics [29] that more closely models SQL, projection does not affect provenance annotations, but *duplicate elimination* $\oplus$-es the annotations of merged tuples;
- the provenance annotations of unioned tuples are $\oplus$-ed;
- the provenance annotations of tuples combined in a cross product are $\otimes$-ed.

Example 5. Consider the security semiring and query Q_1, which can be rewritten in the relational algebra as:

$$\Pi_{\mathsf{city}}(\sigma_{\mathsf{id}<\mathsf{id2}}(\Pi_{\mathsf{id},\mathsf{city}}(\mathsf{Personal}) \bowtie \rho_{\mathsf{id}\to\mathsf{id2}}(\pi_{\mathsf{id},\mathsf{city}}(\mathsf{Personal})))))$$

(the join operator $\bowtie$ being a combination of a cross product, selection, and projection). Using the inductive definition of the provenance of a tuple in a query result, and assuming that the initial provenance tokens t_i are equal to the classification attribute of the tuple, one can compute the provenance of the output of the query as:

city	
New York	restricted
Paris	confidential
Berlin	secret

Similarly, if we consider the counting semiring and query Q_1, assuming the initial provenance tokens t_i are equal to the id attribute of the tuple, one can compute the provenance of the output of the query as:

city	
New York	2
Paris	63
Berlin	28

□

Indeed, simply using this inductive definition of semiring provenance, one can use different semirings to compute different meta-information on the output of a query, with polynomial-time overhead in data complexity:

counting semiring: the number of times a tuple can be derived;

Boolean semiring: if a tuple exists when a subdatabase is selected;

security semiring: the minimum clearance level required to get a tuple as a result;

tropical semiring: minimum-weight way of deriving a tuple (as when computing shortest paths in a graph);

positive Boolean functions: Boolean provenance, *as previously defined*;

integer polynomials: universal provenance, see further;

why-semiring: why-provenance of [11], set of combinations of tuples needed for a tuple to exist.

However, [27] makes two important observations that lead to a different way to compute provenance annotations, instead of doing it one semiring at a time. First, semiring homomorphisms *commute* with provenance computation: if there is a homomorphism from semiring K to semiring K', then one can compute the provenance in K, apply the homomorphism, and obtain the same result as when computing provenance in K'. Second, the integer polynomial semiring $\mathbb{N}[X]$ is the unique *universal* semiring in the following sense: there exists a unique homomorphism to any other commutative semiring that respects a given valuation of the variables. Combining these two facts, we have that *all computations can be performed in the universal semiring*, with homomorphisms only applied when the provenance for a given semiring is required. This suggests a way to implement provenance computation in a DBMS, discussed in Section 5.

Note that two queries that are equivalent in the usual sense [1] can have different semiring provenance, as semiring provenance captures more than logical equivalence. Indeed, two queries are logically equivalent if and only if they have the same *Boolean provenance* on every database.

Provenance semirings only capture the positive relational algebra, a relatively small fragment of SQL. We next discuss how to go beyond this fragment,

and investigate if all interesting forms of provenance are captured by provenance semirings.

4. BEYOND SEMIRING PROVENANCE

We now discuss some extensions of the provenance semiring framework.

Semirings with monus. Semiring provenance can only be defined for the positive fragment of the relational algebra, excluding non-monotone operations such as difference. However, some semirings can be straightforwardly equipped with a *monus* operator $\ominus$ [6, 24], that captures non-monotone behavior. Such an operator must verify the following properties, for all $a, b, c \in K$:

- $a \oplus (b \ominus a) = b \oplus (a \ominus b)$;
- $(a \ominus b) \ominus c = a \ominus (b \oplus c)$;
- $a \ominus a = \mathbb{0} - a = \mathbb{0}$.

This is the case for the Boolean function semiring, which, equipped with the monus operator $a \ominus b = a \wedge \neg b$, forms a *semiring with monus*, or *m-semiring* for short. This is also the case [7] for the why-semiring with set difference, the integer polynomial and counting semirings with truncated difference on scalar values ($a \ominus b = \max(0, a - b)$), etc. Indeed, most natural semirings (though not all [5]) can be extended to m-semirings.

Once such an m-semiring is defined, provenance of the full relational algebra can be captured in that m-semiring. For Boolean functions, it coincides with the Boolean provenance introduced in Section 4.

Note, however, that sometimes some seemingly natural axioms, such as distributivity of $\otimes$ over $\ominus$, fail over m-semirings [7], which implies that two very similar queries may return different provenances.

Another important difference between m-semirings and semirings is that $\mathbb{N}[X]$ is not a universal m-semiring [24]. There does indeed exist a unique universal m-semiring [24], but it is simply the *free m-semiring*, i.e., the m-semiring of free terms constructed using $\oplus$, $\otimes$, $\ominus$, quotiented by the equivalence relations imposed by the m-semiring structure.

We will illustrate in Section 5 how computation is performed using m-semirings.

Provenance for aggregates. One of the most natural ways to extend the relational algebra is to add aggregation capabilities [37]. There have been attempts at defining provenance formalisms for aggregate queries [8, 22]. This is feasible in the case of associative and commutative aggregation, though it requires moving from annotations at the level of tuples to annotations at the level of values, as elements of a *semimodule* that combines provenance semiring annotation with scalar values from the range of the aggregation function. We believe the representation systems of Section 5 could be extended to provide somewhat compact representations of these semimodule elements, but leave this for future work.

Where-provenance. One notable provenance formalism that was introduced early on [11] is where-provenance. The where-provenance is a bipartite graph that connects values in the output relation to values in the input relation to indicate where a specific value may come from in the input. It was shown [12] that where-provenance *cannot* be captured by semiring provenance: there is no semiring for which semiring provenance allows reconstructing the where-provenance of a query. This is, intuitively, for two reasons:

- since where-provenance is assigned to individual values instead of tuples, it is affected either by renaming (in the named perspective) or by projection (in the unnamed perspective), as there needs to be a way to keep track of which value of a given tuple has which where-provenance;
- where-provenance is affected by joins differently as by a combination of cross product, selection, and projection: a value that results from a join of two relations has where-provenance pointing to the joined value in both relations.

However, a system keeping track of semiring provenance could be relatively straightforwardly extended to keep track of where-provenance: instead of dealing with values in a semiring (or in an m-semiring), just maintain value in a free algebra of terms, whose operator includes, in addition to $\oplus$, $\otimes$, and perhaps $\ominus$, operators to record projections (or renaming) and joined values.

Recursive queries. Query languages considered so far are unable to express recursive queries, such as shortest distance in a graph. It is also possible to define provenance notions for such queries as extension to semiring provenance, as long as the provenance formalism can express cycles. This was done in the original work on provenance semirings [27] for *ω-continuous semirings*, showing that the semiring $\mathbb{N}^\infty[[X]]$ of formal power series with integer coefficients is a universal ω-continuous semiring. An alternative approach is the use of semirings with Kleene stars [21], such as k-closed semirings [39], for which efficient algorithms for provenance computation can be designed [39]. We leave details for further work, though we note that the representation systems we are introducing next – circuits – need to

be amended in the case of recursive queries, using either equation systems as in [27] or *cycluits* (cyclic circuits) as in [2].

5. PROVENANCE CIRCUITS

We now discuss concrete representations for provenance annotations. As we have seen, leaving the case of provenance for aggregates, where-provenance, and recursive queries, to future work, provenance annotations can be Boolean functions (see Section 2) useful for probabilistic databases, semiring values (see Section 3), or m-semiring values (see Section 4). In addition, positive Boolean provenance is a special case of semiring provenance, and non-monotone Boolean provenance is a special case of m-semiring provenance. Finally, there exist a universal semiring and a universal m-semiring.

In some semirings (the Boolean, counting, and security semirings, for instance), provenance annotations are *elementary*, i.e., they are easily representable with enumerations or native types. Other semirings, such as $\mathbb{N}[X]$ or the Boolean function semirings have complex annotations, for which a compact representation needs to be found.

In many previous works [27, 31, 45], provenance annotations have been represented as formulas, e.g., propositional formulas for Boolean provenance. But this leads to suboptimal representations, as (Boolean) formulas can be less compact than (Boolean) *circuits* [4, 47]. We therefore argue in favor of using *provenance circuits*, arithmetic circuits whose gates are the operators of the (m-)semiring as in [3, 20], as a compact representation system for provenance.

Example 6. Consider queries Q_1 and Q_2 on Personal. We can represent the provenance annotations of their output as references to gates in the universal m-semiring circuit shown in Figure 1. The output of Q_1 is as follows:

city	
New York	g_1
Paris	g_2
Berlin	g_3

while that of Q_2 is:

city	
New York	g_4
Paris	g_5
Berlin	g_6

Indeed, by developing the circuit, one can for instance verify that the provenance of "New York" for Q_2 on Personal is $(t_1 \oplus t_2) \ominus (t_1 \otimes t_2)$. As can be seen on Figure 1, a significant amount of sharing can be obtained for provenance within and across queries by using provenance circuits. □

This suggests a practical way for computing provenance of queries over a relational database: inductively construct a provenance circuit over input tuples for each operation performed in a query, reusing parts of the circuit that have been constructed by subqueries. By constructing this circuit in the universal m-semiring, it then becomes easy to instantiate it to a wide variety of semirings and m-semirings.

Example 7. Consider the query Q_1 on Personal, for which we want to compute the security and counting semiring annotations. Since we have already computed in Figure 1 a circuit for this query in the universal m-semiring, we can directly obtain a circuit whose evaluation returns the provenance of Q_1 in either of these semirings by applying the appropriate semiring homomorphisms. This is what is shown in Figures 2 and 3. One can verify that the evaluation of these queries returns the provenance annotations already computed in Example 5.

Similarly, one can compute the Boolean circuit of Figure 4 by applying the m-semiring homomorphism from the universal m-semiring of Figure 1 to the Boolean function semiring. □

This approach of incremental provenance computation in the universal m-semiring, with specialization to arbitrary semirings and m-semirings on demand, is that taken by ProvSQL [43], a lightweight add-on to the PostgreSQL database management system for support of (m-)semiring provenance computation on relational databases. To our knowledge, this is the only publicly available system for management of data with semiring provenance, with support of a large subset of the SQL query language. The closest such software may be orchestra [25], which is unfortunately unavailable.

6. PROBABILITY EVALUATION

As discussed in Section 2, Boolean provenance is a very important tool to compute the probability of a query in probabilistic databases, an intractable (#P-hard) problem in general [45]. It allows separating the concerns between query evaluation on the one hand, which produces a Boolean provenance annotation in polynomial time, and probability evaluation of the provenance annotation on the other hand, itself a #P-hard problem.

Let us thus assume we have obtained a Boolean circuit of the data provenance of a query over a probabilistic database. What can, then, be done to evaluate the probability of the provenance annotation, given the intractability of the problem?

Brute-force algorithms. The first possible way is to resort to an exponential-time enumera-

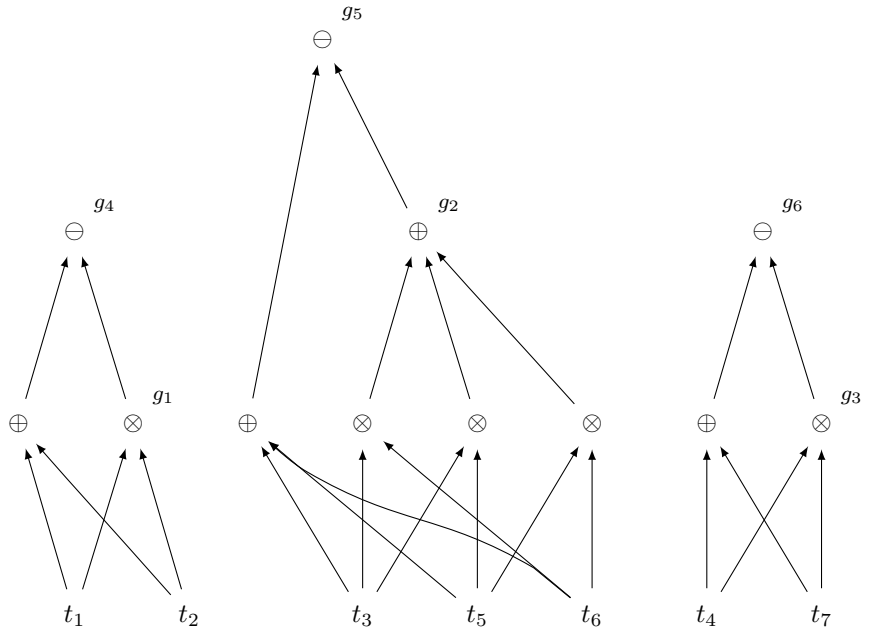


Figure 1: Provenance circuit for queries Q_1 and Q_2 in the universal m-semiring

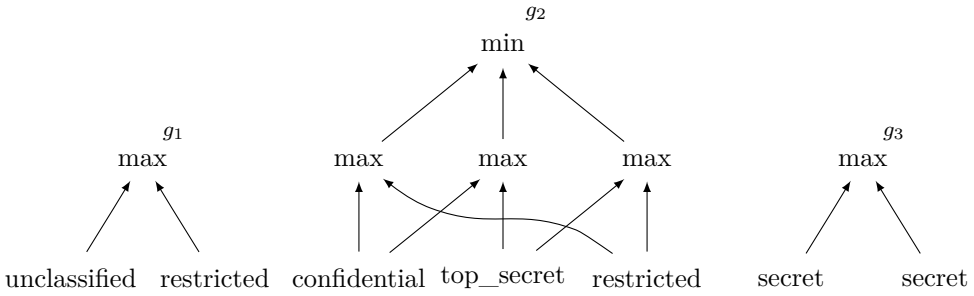


Figure 2: Provenance circuit for query Q_1 in the security semiring

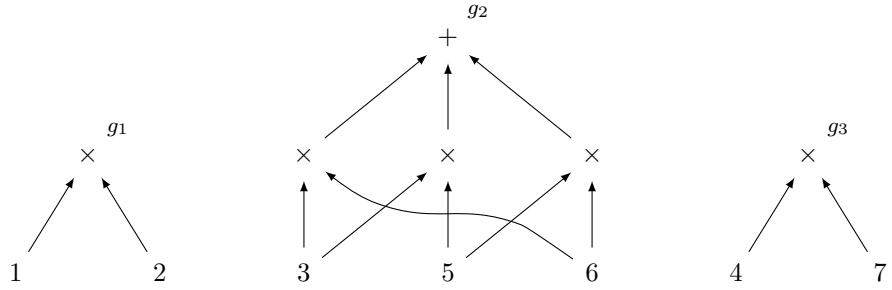


Figure 3: Provenance circuit for query Q_1 in the counting semiring

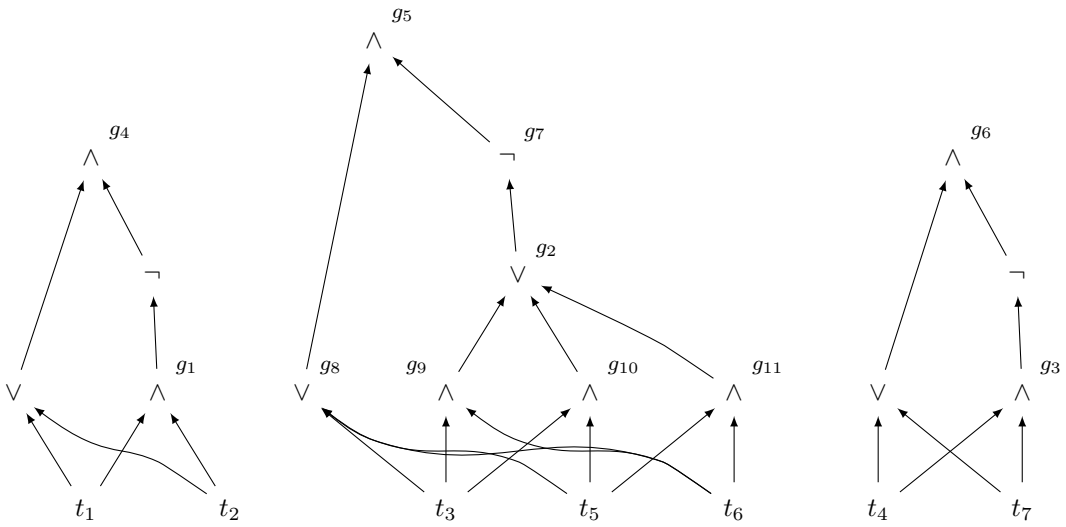


Figure 4: Boolean circuit for queries Q_1 and Q_2

tion of all possible valuations of the Boolean events occurring in the provenance annotation, and summing the probabilities of valuations mapped to $\top$ by the provenance annotation to compute the overall probability. This is rarely feasible, but note that it is at least more efficient than enumerating all possible worlds of the initial database.

Approximations. One can always resort to approximating the probability of the query, either by Monte-Carlo sampling, which is always feasible but fairly slow, or by more refined approximation techniques, as in [42, 44].

Exploiting the query structure. Jha and Suciu [33] have shown that, when queries have specific forms, it is possible to construct Boolean provenance circuits of certain types, that allow for efficient probability evaluation. In particular, if a union of conjunctive queries (UCQ) is *inversion-free*, an ordered binary diagram (OBDD [10]) for it can be obtained. If some more general property is satisfied, then it admits a deterministic decomposable negation normal form (d-DNNF [16]). Both OBDDs and the more general class of d-DNNFs allow for efficient (linear-time) probabilistic query evaluation. Of course, this approach is inapplicable if the query is not of the specific form required. Note also that this approach is specific to Boolean provenance, which means it precludes computation of provenance in a more general (m-)semiring before specializing to the Boolean function case.

Exploiting the data structure. An alternative is to exploit the fact that the structure of the data is not arbitrary. Indeed, if the data has the structure of a tree, or has a low treewidth, meaning that its structure is close to that of a tree, it has been shown [2, 3] that a bounded-treewidth provenance circuit can be constructed, which in turn supports tractable query evaluation. This line of technique has been successfully applied to synthetic [40] and real-world [38] data, for specific kinds of queries.

When none of this is feasible, one can resort to general *knowledge compilation* techniques [18]. Knowledge compilation is the problem of transforming Boolean functions of a certain form into another, more tractable, form. Over the years, a wide variety of techniques, results, heuristics, and tools have emerged from the knowledge compilation community. In particular, tools such as C2D [17], DSHARP [41], and D4 [35] compile arbitrary formulas in *conjunctive normal form* into d-DNNFs.

One practical approach for probabilistic query evaluation is therefore to produce a Boolean provenance circuit, transform it into a conjunctive normal form in linear time using the standard Tseitin transformation [46], and feed it to a knowledge compiler. This approach is used in PROVSQL.

Example 8. Consider the middle connected component of the Boolean circuit of Figure 4, and, in particular, gate g_5 which yields the Boolean provenance of "Paris" in the output of query Q_2 on Personal. One can transform this part of the circuit into the following equivalent conjunctive normal form, where variables are inputs of the circuit along with its

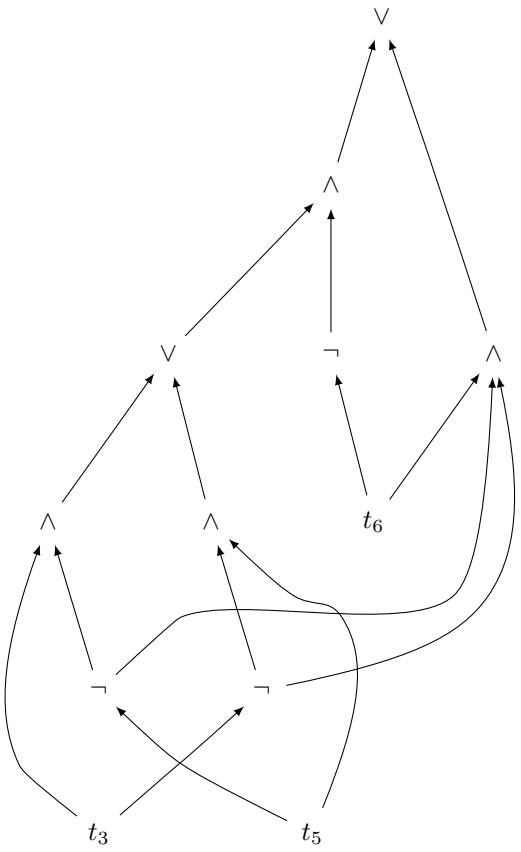


Figure 5: d-DNNF for tuple "Paris" in the output of query Q_2

internal gates, using Tseitin transformation:

$$
\begin{array}{llll}
 & g_5 \vee \bar{g}_8 \vee \bar{g}_7 & \wedge & \bar{g}_5 \vee g_8 \\
\wedge & \bar{g}_5 \vee g_7 & \wedge & \bar{g}_8 \vee t_3 \vee t_5 \vee t_6 \\
\wedge & g_8 \vee \bar{t}_3 & \wedge & g_8 \vee \bar{t}_5 \\
\wedge & g_8 \vee \bar{t}_6 & \wedge & g_7 \vee g_2 \\
\wedge & \bar{g}_7 \vee \bar{g}_2 & \wedge & \bar{g}_2 \vee t_9 \vee t_{10} \vee t_{11} \\
\wedge & g_2 \vee \bar{t}_9 & \wedge & g_2 \vee \bar{t}_{10} \\
\wedge & g_2 \vee \bar{t}_{11} & \wedge & g_9 \vee \bar{t}_3 \vee \bar{t}_6 \\
\wedge & \bar{g}_9 \vee t_3 & \wedge & \bar{g}_9 \vee t_6 \\
\wedge & g_{10} \vee \bar{t}_3 \vee \bar{t}_5 & \wedge & \bar{g}_{10} \vee t_3 \\
\wedge & \bar{g}_{10} \vee t_5 & \wedge & g_{11} \vee \bar{t}_6 \vee \bar{t}_6 \\
\wedge & \bar{g}_{11} \vee t_6 & \wedge & \bar{g}_{11} \vee t_6 \\
\wedge & g_5 & &
\end{array}
$$

Once such a formula obtained, it can be given as input to a knowledge compiler. For example, D4 outputs the d-DNNF in Figure 5. A d-DNNF is a special case of a Boolean circuit, where every ¬-gate is directly connected to an input, every ∧-gate has children with disjoint sets of descendant leaves, and every ∨-gate is such that only one of its child can be true in any possible world. These restrictions make it possible to compute the probability of a gate in linear-time, given a probability distribution on input gate: ∧-gates become products, while ∨-gates become sums. The computation of $\Pr(g_5)$ from the d-DNNF in Figure 4 is shown in Figure 6. □

Note that this intensional (provenance-based) approach to probabilistic query evaluation is not the only one. The *extensional approach*, which directly manipulates probabilities as the query is evaluated, without an intermediate provenance representation, has also been successfully used [14, 15]. It is an open problem [15, 33] whether there are cases where the extensional approach succeeds but no compact d-DNNF is obtainable.

7. CONCLUSION

Data provenance is a major tool to obtain additional information on query output. It allows answering questions about:
why in the why-semiring;
where using where-provenance;
how using integer polynomials;
missing tuples using Boolean provenance;
how many times in the counting semiring;
probability using probability evaluation of Boolean provenance;
minimal security clearance in the security semiring;
most economical way in the tropical semiring.

Practical implementation of provenance management is very much possible, since it introduces a relatively low overhead. One avenue for practical implementations is to perform all computations in a universal structure, such as the universal m-semiring, and only specialize when needed. Using these provenance representations, it is also possible to perform query evaluation on probabilistic databases, for instance using knowledge compilation to obtain Boolean provenance representations on which probabilistic evaluation is efficient.

8. REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
[2] Antoine Amarilli, Pierre Bourhis, Mikaël Monet, and Pierre Senellart. Combined tractability of query evaluation via tree automata and cycluits. In *ICDT*, 2017.
[3] Antoine Amarilli, Pierre Bourhis, and Pierre Senellart. Provenance circuits for trees and treelike instances. In *ICALP*, 2015.

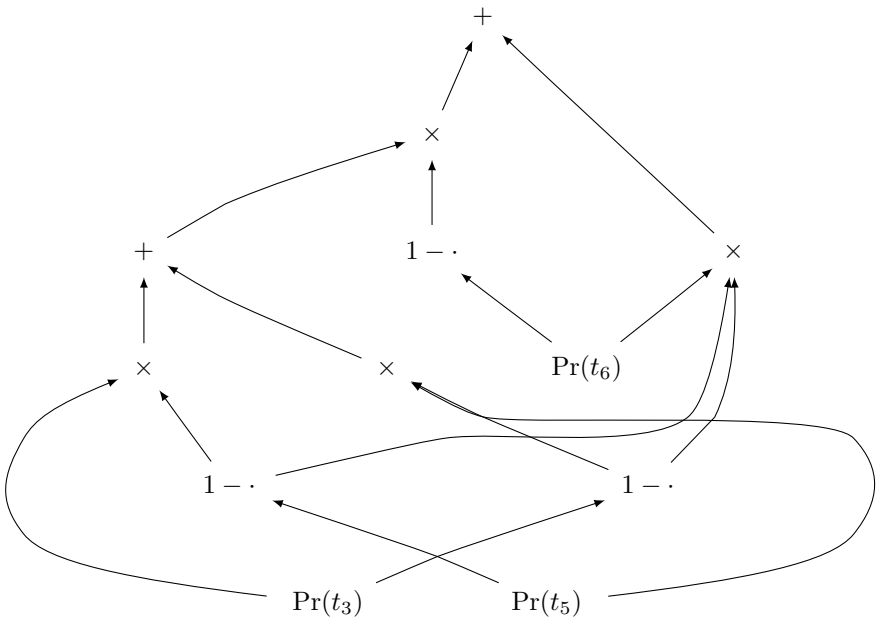


Figure 6: Probability computation following the d-DNNF of Figure 5

[4] Antoine Amarilli, Pierre Bourhis, and Pierre Senellart. Tractable lineages on treelike instances: Limits and extensions. In *PODS*, 2016.

[5] Antoine Amarilli and Mikaël Monet. Example of a naturally ordered semiring which is not an m-semiring. `http://math.stackexchange.com/questions/1966858`, 2016.

[6] K. Amer. *Algebra Universalis*, 18(1), 1984.

[7] Yael Amsterdamer, Daniel Deutch, and Val Tannen. On the limitations of provenance for queries with difference. In *TaPP*, 2011.

[8] Yael Amsterdamer, Daniel Deutch, and Val Tannen. Provenance for aggregate queries. In *PODS*, 2011.

[9] Omar Benjelloun, Anish Das Sarma, Alon Halevy, and Jennifer Widom. ULDBs: Databases with uncertainty and lineage. In *VLDB*, 2006.

[10] Randal E. Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3), 1992.

[11] Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. Why and where: A characterization of data provenance. In *ICDT*, 2001.

[12] James Cheney, Laura Chiticariu, and Wang Chiew Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4), 2009.

[13] Yingwei Cui and Jennifer Widom. Practical lineage tracing in data warehouses. In *ICDE*, 2000.

[14] Nilesh Dalvi and Dan Suciu. Efficient query evaluation on probabilistic databases. *The VLDB Journal*, 16(4), 2007.

[15] Nilesh Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6), 2012.

[16] Adnan Darwiche. On the tractable counting of theory models and its application to truth maintenance and belief revision. *J. Applied Non-Classical Logics*, 11(1-2), 2001.

[17] Adnan Darwiche. New advances in compiling CNF to decomposable negation normal form. In *ECAI*, 2004.

[18] Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *J. Artificial Intelligence Research*, 17(1), 2002.

[19] Susan B Davidson and Juliana Freire. Provenance and scientific workflows: challenges and opportunities. In *SIGMOD*, 2008.

[20] Daniel Deutch, Tova Milo, Sudeepa Roy, and Val Tannen. Circuits for Datalog provenance. In *ICDT*, 2014.

[21] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of weighted automata*. Springer Science & Business Media, 2009.

[22] Robert Fink, Larisa Han, and Dan Olteanu.

Aggregation in probabilistic databases via knowledge compilation. *PVLDB*, 5(5), 2012.
[23] Norbert Fuhr and Thomas Rölleke. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems*, 15(1), 1997.
[24] Floris Geerts and Antonella Poggi. On database query languages for K-relations. *J. Applied Logic*, 8(2), 2010.
[25] Grigoris Green, Todd J. and Karvounarakis, Zach Ives, and Val Tannen. Provenance in ORCHESTRA. *IEEE Data Eng. Bull*, 33(3), 2010.
[26] Todd J Green. Containment of conjunctive queries on annotated relations. *Theory of Computing Systems*, 49(2), 2011.
[27] Todd J Green, Grigoris Karvounarakis, and Val Tannen. Provenance semirings. In *PODS*, 2007.
[28] Todd J. Green and Val Tannen. Models for incomplete and probabilistic information. *IEEE Data Eng. Bull.*, 29(1), 2006.
[29] Stéphane Grumbach and Tova Milo. Towards tractable algebras for bags. *J. Computer and System Sciences*, 52(3), 1996.
[30] Olaf Hartig. Provenance information in the web of data. In *LDOW*, 2009.
[31] Tomasz Imielinski and Witold Lipski Jr. Incomplete information in relational databases. *J. ACM*, 31(4), 1984.
[32] Abhay Jha, Dan Olteanu, and Dan Suciu. Bridging the gap between intensional and extensional query evaluation in probabilistic databases. In *EDBT*, 2010.
[33] Abhay Jha and Dan Suciu. Knowledge compilation meets database theory: compiling queries to decision diagrams. *Theory of Computing Systems*, 52(3), 2013.
[34] Grigoris Karvounarakis and Todd J Green. Semiring-annotated data: queries and provenance? *ACM SIGMOD Record*, 41(3), 2012.
[35] Jean-Marie Lagniez and Pierre Marquis. An improved decision-DNNF compiler. In *IJCAI*, 2017.
[36] Laks VS Lakshmanan, Nicola Leone, Robert Ross, and Venkatramanan Siva Subrahmanian. Probview: A flexible probabilistic database system. *ACM Transactions on Database Systems*, 22(3), 1997.
[37] Leonid Libkin and Limsoon Wong. Query languages for bags and aggregate functions. *J. Computer and System sciences*, 55(2), 1997.
[38] Silviu Maniu, Reynold Cheng, and Pierre Senellart. An indexing framework for queries on probabilistic graphs. *ACM Transactions on Database Systems*, 42(2), 2017.
[39] Mehryar Mohri. Semiring frameworks and algorithms for shortest-distance problems. *J. Automata, Languages and Combinatorics*, 7(3), 2002.
[40] Mikaël Monet. Probabilistic evaluation of expressive queries on bounded-treewidth instances. In *SIGMOD/PODS PhD Symposium*, 2016.
[41] Christian J Muise, Sheila A McIlraith, J Christopher Beck, and Eric I Hsu. Dsharp: Fast d-DNNF compilation with sharpSAT. In *Canadian Conference on AI*, 2012.
[42] Dan Olteanu, Jiewen Huang, and Christoph Koch. Approximate confidence computation in probabilistic databases. In *ICDE*, 2010.
[43] Pierre Senellart. ProvSQL. `https://github.com/PierreSenellart/provsql`, 2017.
[44] Asma Souihli and Pierre Senellart. Optimizing approximations of DNF query lineage in probabilistic XML. In *ICDE*, 2013.
[45] Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. *Probabilistic Databases*. Morgan & Claypool, 2011.
[46] G Tseitin. On the complexity of derivation in propositional calculus. *Studies in Constrained Mathematics and Mathematical Logic*, 1968.
[47] Ingo Wegener. *The complexity of Boolean functions*. Wiley, 1987.
[48] Allison Woodruff and Michael Stonebraker. Supporting fine-grained data lineage in a database visualization environment. In *ICDE*, 1997.

On Measuring Bias in Online Information

Evaggelia Pitoura, Panayiotis Tsaparas
Computer Science and Engineering Dept.
University of Ioannina, Greece
{pitoura,tsap}@cs.uoi.gr

Giorgos Flouris, Irini Fundulaki, Panagiotis Papadakos
Institute of Computer Science, FORTH, Greece
{fgeo,fundul,papadako}@ics.forth.gr

Serge Abiteboul
INRIA & ENS, Paris, France
serge.abiteboul@inria.fr

Gerhard Weikum
Max Planck Institute for Informatics, Germany
weikum@mpi-inf.mpg.de

ABSTRACT

Bias in online information has recently become a pressing issue, with search engines, social networks and recommendation services being accused of exhibiting some form of bias. In this vision paper, we make the case for a systematic approach towards measuring bias. To this end, we discuss formal measures for quantifying the various types of bias, we outline the system components necessary for realizing them, and we highlight the related research challenges and open problems.

1. INTRODUCTION

Today, the majority of our diverse information needs are satisfied online, by search engines, social networks and media, news aggregators, e-shops, vertical portals, and other online information providers (**OIPs**). These providers use sophisticated algorithms to produce a ranked list of results tailored to our profile. These results play an important role in guiding our decisions and shaping our opinions, and in general in our view of the world.

There are increasingly frequent reports of OIPs exhibiting some form of bias. For instance, in the recent US presidential elections, Google was accused of being biased against Donald Trump[1] and Facebook of contributing to the post-truth politics[2]. Google search has been accused of being sexist or racist when returning images for queries such as "nurse" or "hair-styling"[3], and prejudiced when answering queries about holocaust[4]. Similar accusations have been made for Flickr, Airbnb and LinkedIn. In fact, the problem of understanding and addressing bias is considered a high-priority problem for machine learning algorithms and AI for the next few years[5].

According to the Oxford English Dictionary[6], bias is *"an inclination or prejudice for or against one person or group, especially in a way considered to be unfair"*, and as *"a concentration on or interest in one particular area or subject"*. When it comes to bias in OIPs, we make the distinction between *user bias* and *content bias*. User bias appears when different users receive different content based on user attributes that should be protected, such as gender, race, ethnicity, or religion. Content bias refers to biases in the information received by any user, where some aspect is disproportionately represented in a query result or in news feeds.

The problem has attracted some attention in the data management community as well [27]. In this paper, we make the case for a systematic approach to addressing the problem of bias in the data provided by the OIPs. Addressing bias involves many steps. Here, we focus on the very first step, that, of defining and measuring bias.

2. RELATED WORK

In the field of machine learning, there is an increasing concern about the potential risks of data-driven approaches in decision making algorithms [2, 15, 25, 27], raising a call for equal opportunities by design [19]. Biases can be introduced at different stages of the design, implementation, training and deployment of machine learning algorithms. There are reports for discriminatory ads based on either race [28], or gender [9], and recommendation algorithms showing different prices to different users [17]. Consequently, there are efforts for defining principles of

[1] https://www.theguardian.com/us-news/2016/sep/29/donald-trump-attacks-biased-lester-holt-and-accuses-google-of-conspiracy
[2] https://www.theguardian.com/us-news/2016/nov/16/facebook-bias-bubble-us-election-conservative-liberal-news-feed
[3] http://fusion.net/story/117604/looking-for-ceo-doctor-cop-in-google-image-search-delivers-crazy-sexist-results/
[4] http://www.bbc.com/news/technology-38379453
[5] https://futureoflife.org/ai-principles/
[6] https://en.oxforddictionaries.com/definition/bias.

accountable algorithms[7], for auditing algorithms by detecting discrimination [7, 26] and for debiasing approaches [1, 37]. There is a special interest for racial fairness and fair classifiers [18, 34, 35, 6], ensuring that groups receive ads based on population proportions [9] and reducing the discrimination degree of algorithms against individuals of a protected group [13]. Other efforts try to ensure temporal transparency for policy changing events in decision making systems [12]. Recently, practical tools for addressing bias have appear, e.g., for removing discriminating information[8], or for showing political biases of Facebook friends and news feed[9].

Regarding search engines and social media, [23] examines how bias can be measured in search, while [21] tries to quantify bias in Twitter data. There are also studies that look at individual aspects of bias, such as geographical [29], or temporal [5], and if search engines can partially mitigate the rich-get-richer nature of the Web [14]. The presence of bias in media sources has been studied based on human annotations [4], message impartiality [33] and through affiliations [31].

Another branch of research focuses on how bias can affect users. According to field studies, biased search algorithms could shift the voting preferences of undecided voters by as much as 20% [10]. Since most users try to access information that they agree with [20], personalization and filtering algorithms lead to echo chambers and filter bubbles that reinforce bias [3, 16]. This is also evident in social media, where platforms strengthen users existing biases [22], minimizing the exposure to different opinions [30].

3. TYPES OF BIAS

We consider bias in terms of *topics*. In particular, we would like to test whether an OIP is biased with respect to a given topic. A topic may be a very general one, such as, politics, or a very specific one down to the granularity of a single search query. For example, we may want to test whether an OIP provides biased results for events such as"Brexit" and "US Elections", people such as "Donald Trump", general issues such as "abortion" and "gun control", transactional queries such as "air tickets", "best burger", or even topics such as "famous people". An OIP may be biased with respect to one topic and unbiased with respect to another one.

We distinguish between two types of bias, namely *user* and *content* bias. User bias refers to bias against the users receiving the information, while content bias looks at bias in the information delivered to users.

For user bias, we assume that some of the attributes that characterize the user of an OIP are *protected* (e.g. race, gender, etc.). User bias exists when the values of these attributes influence the results presented to users. For example consider the case of a query about jobs, where women receive results of lowered paid jobs than men. User bias can also appear due to hidden dependencies between protected and unprotected attributes, even when such protected attributes are not used directly in computing the results (e.g., see [11]). For instance, the home location of users may imply their race.

Content bias refers to bias in the results provided by the OIP and may appear even when there is just a single user. For example, an instance of this kind of bias occurs when an OIP promotes its own services over the competitive ones, or, when the results for queries about a political figure take an unjustifiable favorable, or unfavorable position towards this politician (independently of the user receiving the results).

In most cases, the OIP content is presented in the form of a ranked list of results. Results are often complex objects, such as news feeds, web pages, or, even physical objects, in the case of recommendations. We assume that results can be described by features, or attributes, either explicitly provided, or intentionally extracted. In analogy to protected attributes for users, we consider *differentiating attributes* for topics. For instance, for a controversial topic such as "abortion" or "gun control", the differentiating attribute could be the stance (pro, or against). For a topic such as "famous people", we may want to test whether the results are biased towards men over women, or, favor people from specific countries, or, over-represent, say, artists over scientists. Finally, for a topic such as "US Elections", a differentiating attribute may be the political party (with values, "Democrats" or "Republicans").

In a sense, addressing user bias can be regarded as a counterweight to machine-learning and personalization algorithms that try to differentiate the needs of various user groups, so that these algorithms do not discriminate over specific protected attributes. On the other hand, addressing content bias has some similarity to result diversification [8]. However, diversity is related to coverage, since we want all various aspects of a topic, even the rarest ones, to appear in the result. For content bias, we want the differentiating attributes to be represented proportionally to a specific "ground truth".

A commonly encountered case is the case of a

[7]http://www.fatml.org
[8]http://www.debiasyourself.org/
[9]http://politecho.org/

combined user and content bias appearing when a specific facet is over-represented in the results presented to a specific user population, e.g., democrats get to see more pro-Clinton articles than republicans. This type of bias is also related to *echo chambers*, i.e., the situation in which information, ideas, or beliefs are amplified, exaggerated or reinforced inside groups of equally-minded people. Since similar people may be interested in specific aspects of a topic, the content they create, consume, or prefer is biased towards these aspects. Then, the information presented to them may reflect this bias and lead to bias amplification, creating a bias-reinforcement cycle. In such cases, there is often some relation between the protected attributes of the users and the differentiating attributes of the topic.

4. BIAS MEASURES

In this section, we present measures for user and content bias. Our goal is not to be overly formal, but instead we provide such measures as a means to make the related research challenges more concrete.

We assume that the information provided by an OIP is in the form of a ranked list R. In the core of each bias measure lies a definition of similarity between lists of results. For now, let us assume a distance function $D_R(R_1, R_2)$ between two ranked lists of results R_1 and R_2. D_R can be defined by employing some existing distance metric between ranked lists. We will revisit this issue when we talk about content bias.

To simplify the discussion, in the following, we assume that the topic for which we want to measure bias is a single query q. We can generalize the definitions to a set of queries by adopting some aggregation measure of the metrics for a single query.

User Bias. Let U be the user population of the OIP. For simplicity, assume a binary protected attribute that divides users into a protected class P and an unprotected class $\bar{P}$. For example, if the protected attribute is gender, P may denote the set of women and $\bar{P}$ the set of men. Intuitively, we do not want the information provided to users to be influenced by their protected attributes.

The problem of user bias is somehow related to fairness in classification, where individuals are classified in a positive or negative class. Example applications include hiring, school admission, crime risk factor estimation, medicine (e.g., suitability for receiving a medical treatment) and advertisement.

There are two general approaches to fairness: *group* and *individual* fairness [9]. Group fairness imposes requirements on the protected and unprotected class as a whole. A common example of group fairness is *statistical parity* where the proportion of members in the protected class that receive positive classification is required to be identical with their proportion in the general population. Individual fairness requires similar people to be treated similarly. Both approaches have drawbacks. Group fairness does not take into account the individual merits of each group member and may lead in selecting the less qualified members of a group. Individual fairness assumes a similarity metric between individuals that is classification-task specific and hard to define.

A technical difference between fairness and user bias is that most work in fairness focuses on classification tasks, while, in our case, results are ranked. Very recent work addresses fair ranking (where the output is a ranked list of individuals) by adopting a group based approach that asks for a proportional presence of individuals of the protected class in all prefixes of the ranked list [32, 36]. A conceptual difference between the two problems is that in the case of fairness, users are the ones who are being classified (or ranked), whereas in user bias, the users are the ones who receive ranked information.

An individual-based approach to user bias assumes that it is possible to define an appropriate distance measure D_u between the users in U. The distance should capture when two users are considered similar for the *topic* under consideration. For instance, if the topic is jobs, individuals with the same qualifications should be considered similar independently of their gender. The following definition reflects the premise that similar users should receive similar result lists.

Definition 1 (Individual User Bias). *An online information provider is individual user unbiased if for any pair of users u_1 and u_2, it holds: $D_R(R_{u_1}, R_{u_2}) \leq D_u(u_1, u_2)$, where R_{u_1} and R_{u_2} are the result lists received by u_1 and u_2 respectively.*

There are many ways of expressing group-based user bias. We will discuss one. Let $\mathcal{R}_{\mathcal{P}}$ be the union of the result lists seen by the members of the protected class and $\mathcal{R}_{\bar{\mathcal{P}}}$ be the union of the result lists seen by the members of the non-protected class. We could aggregate the results in each of them to create two representative ranked lists, R_P and $R_{\bar{P}}$, for $\mathcal{R}_{\mathcal{P}}$ and $\mathcal{R}_{\bar{\mathcal{P}}}$, respectively. We can define now user bias using these representative ranked lists.

Definition 2 (Group User Bias). *An online information provider is group user unbiased if it holds: $|D_R(R_P, R_{\bar{P}})| \leq \epsilon$, for some small $\epsilon \geq 0$.*

Aggregating result lists is just one possibility. Another is to require the probability that a member of

P receives any of the lists in $\mathcal{R}_\mathcal{P}$ to be the same with the probability that a member of $\bar{P}$ receives it (and vice versa). All group-based definitions ignore the profiles of individual users; i.e., they do not capture the fact that a result list should be relevant to the specific individual in the group who receives it.

Content Bias. Let us first assume that there is just one user. Let A be a differentiating attribute, and let $\{a_1, ..., a_m\}$ be the values of A. For example, in the case of a query about elections, $a_1, ..., a_m$ may correspond to the different parties that participate in the elections. We also assume that each result is annotated with the values of attribute A.

A distinctive characteristic of content bias is that content bias can be defined only relatively to some "ground truth", or "norm". But what should be the "ground truth"? One option is to consider the actual data used by the OIP for computing the content delivered to users as the ground truth. For example, this is the approach taken in [21] that compares the political bias of Twitter search with the bias in all tweets that contain the search terms. However, user-generated content may include biases inflicted by the design and affordances of the OIP platform, or by behavioral norms emerging on the platform. Bias could also be introduced by the OIP during the acquisition of data (e.g. during crawling and indexing for a search engine). See [24] for a complete analysis of the different biases and pitfalls associated with social data. There are also cases, where the actual data used by the OIP may not be available.

Ideally, we would like to have an indisputably unbiased ranked list of results. Such lists could be constructed using an aggregation of OIPs and other external sources such as knowledge bases, or domain experts. Crowdsourcing could also be utilized in creating such lists. In some cases, an estimation of the distribution of values of the differentiating attributes in the general population may be available. For example, for the election query, we could use external sources, such as polls, to estimate the actual party popularity and user intention to vote. One could also think of creating bias benchmarks consisting of reference sample topics and result lists similar to TPC benchmarks for evaluating database system performance, and TREC tracks for evaluating relevance in information retrieval.

Given the ground truth as an "ideal unbiased ranking" R_T, we could define content bias looking at its distance from the ground truth.

DEFINITION 3 (CONTENT BIAS). *An online information provider is content unbiased if it holds: $D_R(R_u, R_T)| \leq \epsilon$, for some small $\epsilon \geq 0$.*

One way of defining D_R is using the distribution of the values of the differentiating attribute in an ideal ranking. Assume that we have the "ground truth" in the form of probabilities $Pr_T(a_i)$ for all the attribute values which captures the relative popularity of each value (e.g., the support of a party as measured by polls). Let $Pr(u, a_i)$ be the probability that user u receives a result annotated with value a_i (e.g., one possible definition is this to be defined as the fraction of the top-k results that are about a_i). The following equation could serve as a definition of D_R.

$$D_R(R_u, R_T) = \max_i |Pr(u, a_i) - Pr_T(a_i)| \quad (1)$$

Combined User-Content Bias. We can refine user bias, using content-aware distance definitions, such as the one in Equation (1). For example, in Definition 1, we could use:

$$D_R(R_{u_1}, R_{u_2}) = \max_i |Pr(u_1, a_i) - Pr(u_2, a_i)| \quad (2)$$

Equation (2) looks at the relative bias of the content seen by two users. Although both users may receive biased content with respect to ground truth, there is no user bias if the content is equally biased.

One way of identifying echo chambers is by measuring the content bias in the result lists seen by different groups. For instance, adopting the representative list approach to user bias, we may look at the distance of R_P and $R_{\bar{P}}$ from the ground truth to test, for example, whether specific attribute values are over-represented in the results shown to a population group.

5. A SYSTEM FOR MEASURING BIAS

We now look at some of the challenges involved in realizing a system for measuring the bias of an OIP. The OIP may be a search engine, a recommendation service, or the news feed service of a social network. In Figure 1, we present the main components of BIASMETER, a system for measuring bias. We treat the OIP as a black-box and assume that BIASMETER can access it only through the interface provided by the OIP, e.g., through search queries. For simplicity, we assume that the set of protected and differentiating attributes are given as input.

Given the topic T and the differentiating attributes A, the goal of the *query generator* is to produce an appropriate set of queries to be submitted to the OIP under consideration. For instance, if the OIP is a search engine, to test about the topic "US elections", the generator may produce queries referring to specific political parties. To produce queries that best represent the topic and the attributes, the query-generator may need to use background knowledge, such as a related knowledge base.

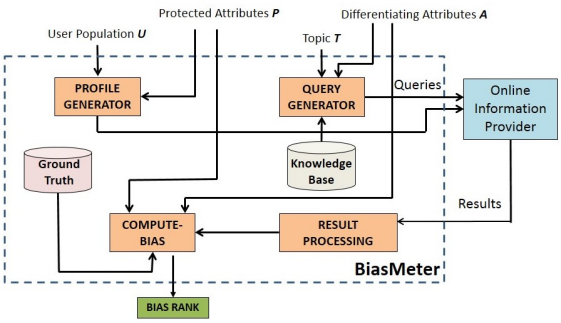


Figure 1: System components.

The *profile generator* takes as input the user population U and the set of protected attributes P and produces a set of user profiles appropriate for testing whether the OIP discriminates over users in U based on the protected attributes in P. For example, if we want to test gender bias in job search queries, we need samples of men and women with similar characteristics with respect to other attributes (e.g., grades, skills, background, ethnicity) to avoid differences in results due to attribute correlations.

There are many issues of both a theoretical and a practical nature in generating profiles. For example, we must ensure that the profiles are an appropriate sample of U that represents all values of the protected attributes. Furthermore, we should ensure that the characteristics of the users in the sample are similar with respect to all other attributes, so as to avoid the effect of confounding factors. This raises issues similar to those met when selecting people for opinion polls, surveys, etc. From a more practical view, we need to assemble users with the specific profiles and ask them to issue the queries (for example using a crowd-sourcing platform, such as Mechanical Turk), or generate artificial accounts of such users. An important step to automated profile generation is offered by AdFisher, a tool for testing discrimination in Google Ads [7].

The *result processing* component takes as input the results from the OIP and applies machine learning and data mining algorithms such as topic modeling and opinion mining to determine the values of the differentiating attributes. For example, for a topic such as "gun control", we need to determine whether a specific result takes a positive, neutral or negative stand.

Finally, the *compute-bias* component calculates the bias of the OIP, using bias metrics and the *ground-truth*. Note that the cause of bias is not specified in the result; we just detect bias with respect to specific user and content attributes.

6. RESEARCH CHALLENGES

Obtaining the ground truth. Defining the ground truth is the most formidable task in identifying bias. One approach could be a human-in-the-loop approach where humans take the role of data processors characterizing the bias of online information, similarly to humans evaluating the relevance of search results. One can even envision novel crowdsourcing platforms specifically targeting bias evaluation. However, such tasks are hindered by strong cognitive biases, such as confirmation bias, that may lead users in discrediting as biased any information that does not fit their own beliefs. Furthermore, bias, as opposed to relevance, may involve political, ideological, or, even, ethical connotations. Besides crowdsourcing, one can envision a form of data-driven validation that integrates information from large data repositories, knowledge bases, and multiple OIPs. Besides this long-term quest for ground truth, a more realistic approach is to rely on comparative evaluations. For instance one could compare the bias between the results of two OIPs or between the results of an OIP and content found in traditional media.

Defining bias measures. Bias is multifaceted. We abstracted the many forms of bias, through the notions of protected attributes for users and differentiating attributes for content. However, there are often correlations among the attributes making it hard to single out the effects of each of them in the results. Furthermore, our measures are high level, and a lot of work is needed to come up with rigorous mathematical formulations.

Engineering and technical challenges. To measure bias with respect to a protected attribute P (e.g. gender), we need to generate large samples of user accounts for the different values of P (e.g., women and men), making sure that the distribution of the characteristics for the other attributes is near identical. Careful statistical analysis is also required to ensure statistical significance of our results. In addition, the query generation and result processing components involve a variety of data mining and machine learning algorithms for identifying keywords to describe an information need, or understanding the topic and stance of a specific result. To this end, we need modules for knowledge representation, record linkage, entity detection and entity resolution, sentiment detection, topic modeling, and more.

Auditing. Bias detection can be simplified, if access is given to the internals of the OIP (e.g., for sampling users with specific demographics, or getting non personalized results). Clearly, this is impossible for an entity outside the OIP and it requires the cooperation of law and policy makers. Such access

would also help in differentiating between bias in the source data and bias in the results.

7. CONCLUSIONS

In this paper, we argue about the importance of a systematic approach for measuring bias in the information from online information providers. As more people rely on online sources to get informed and make decisions, this is of critical value. There are many research challenges to be addressed, some of which we have highlighted in this paper. Measuring bias is just the first step; many steps are needed to counteract bias including identifying sources of bias and developing approaches for debiasing.

8. REFERENCES

[1] G. Adomavicius, J. Bockstedt, C. Shawn, and J. Zhang. *De-biasing user preference ratings in recommender systems*, volume 1253, pages 2–9. CEUR-WS, 2014.

[2] S. Barocas and A. D. Selbst. Big Data's Disparate Impact. *SSRN eLibrary*, 2014.

[3] E. Bozdag. Bias in algorithmic filtering and personalization. *Ethics and Inf. Technol.*

[4] C. Budak, S. Goel, and J. M. Rao. Fair and balanced? quantifying media bias through crowdsourced content analysis. *Public Opinion Quarterly*, 80(S1).

[5] A. Chakraborty, S. Ghosh, N. Ganguly, and K. P. Gummadi. Can trending news stories create coverage bias? on the impact of high content churn in online news media. In *Computation and Journalism Symposium*, 2015.

[6] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq. Algorithmic decision making and the cost of fairness. *CoRR*, abs/1701.08230, 2017.

[7] A. Datta, M. C. Tschantz, and A. Datta. Automated experiments on ad privacy settings. *Proceedings on Privacy Enhancing Technologies*, 2015(1):92–112, 2015.

[8] M. Drosou and E. Pitoura. Search result diversification. *SIGMOD Record*, 39(1):41–47, 2010.

[9] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel. Fairness through awareness. In *ITCS*.

[10] R. Epstein and R. E. Robertson. The search engine manipulation effect (seme) and its possible impact on the outcomes of elections. *PNAS*, 112(20), 2015.

[11] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *KDD*, pages 259–268, 2015.

[12] M. Ferreira, M. B. Zafar, and K. P. Gummadi. The case for temporal transparency: Detecting policy change events in black-box decision making systems. *arXiv preprint arXiv:1610.10064*, 2016.

[13] B. Fish, J. Kun, and Á. D. Lelkes. A confidence-based approach for balancing fairness and accuracy. In *SDM*, pages 144–152, 2016.

[14] S. Fortunato, A. Flammini, F. Menczer, and A. Vespignani. Topical interests and the mitigation of search engine bias. *Proceedings of the National Academy of Sciences*, 103(34):12684–12689, 2006.

[15] S. Hajian, F. Bonchi, and C. Castillo. Algorithmic bias: From discrimination discovery to fairness-aware data mining. In *KDD*, pages 2125–2126. ACM, 2016.

[16] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson. Measuring personalization of web search. In *WWW*, pages 527–538. ACM, 2013.

[17] A. Hannak, G. Soeller, D. Lazer, A. Mislove, and C. Wilson. Measuring price discrimination and steering on e-commerce web sites. In *Internet Measurement Conference*, pages 305–318, 2014.

[18] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. In *NIPS*, pages 3315–3323, 2016.

[19] W. House. Big data: A report on algorithmic systems, opportunity, and civil rights. *Washington, DC: Executive Office of the President, White House*, 2016.

[20] D. Koutra, P. N. Bennett, and E. Horvitz. Events and controversies: Influences of a shocking news event on information seeking. In *WWW*, pages 614–624, 2015.

[21] J. Kulshrestha, M. Eslami, J. Messias, M. B. Zafar, S. Ghosh, I. Shibpur, I. K. P. Gummadi, and K. Karahalios. Quantifying search bias: Investigating sources of bias for political searches in social media. In *CSCW*, 2017.

[22] Z. Liu and I. Weber. Is twitter a public sphere for online conflicts? a cross-ideological and cross-hierarchical look. In *SocInfo*, pages 336–347, 2014.

[23] A. Mowshowitz and A. Kawaguchi. Measuring search engine bias. *Information Processing & Management*, 41(5):1193–1205, 2005.

[24] A. Olteanu, C. Castillo, F. Diaz, and E. Kiciman. Social data: Biases, methodological pitfalls, and ethical boundaries. In *SSNR Preprint*, 2017.

[25] A. Romei and S. Ruggieri. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, 29(05):582–638, 2014.

[26] C. Sandvig, K. Hamilton, K. Karahalios, and C. Langbort. Auditing algorithms: Research methods for detecting discrimination on internet platforms. *Data and discrimination: converting critical concerns into productive inquiry*, 2014.

[27] J. Stoyanovich, S. Abiteboul, and G. Miklau. Data, responsibly: Fairness, neutrality and transparency in data analysis. In *EDBT*, 2016.

[28] L. Sweeney. Discrimination in online ad delivery. *Queue*, 11(3):10, 2013.

[29] L. Vaughan and M. Thelwall. Search engine coverage bias: evidence and possible causes. *Information processing & management*, 40(4):693–707, 2004.

[30] I. Weber, V. R. K. Garimella, and A. Batayneh. Secular vs. islamist polarization in egypt on twitter. In *ASONAM*, pages 290–297, 2013.

[31] F. M. F. Wong, C. W. Tan, S. Sen, and M. Chiang. Quantifying political leaning from tweets and retweets. *ICWSM*, 13:640–649, 2013.

[32] K. Yang and J. Stoyanovich. Measuring fairness in ranked outputs. In *SSDM*, pages 22:1–22:6, 2017.

[33] M. B. Zafar, K. P. Gummadi, and C. Danescu-Niculescu-Mizil. Message impartiality in social media discussions. In *ICWSM*.

[34] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi. Learning fair classifiers. *arXiv preprint arXiv:1507.05259*, 2015.

[35] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *WWW*, 2017.

[36] M. Zehlike, F. Bonchi, C. Castillo, S. Hajian, M. Megahed, and R. A. Baeza-Yates. Fa*ir: A fair top-k ranking algorithm. In *CIKM*, 2017.

[37] J. Zhao, T. Wang, M. Yatskar, V. Ordonez, and K. Chang. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. *CoRR*, abs/1707.09457, 2017.

Digree: Building A Distributed Graph Processing Engine out of Single-node Graph Database Installations*

Vasilis Spyropoulos
Athens University of Economics and Business
Athens, Greece
vasspyrop@aueb.gr

Yannis Kotidis
Athens University of Economics and Business
Athens, Greece
kotidis@aueb.gr

ABSTRACT

In this work we present *Digree*, a system prototype that enables distributed execution of graph pattern matching queries in a cloud of interconnected graph databases. We explain how a graph query can be decomposed into independent sub-patterns that are processed in parallel by the distributed independent graph database systems and how the results are finally synthesized at a master node. We experimentally compare a prototype of our system against a popular big data engine and show that *Digree* provides significantly faster query execution.

1. INTRODUCTION

Attempts to utilize relational databases and big data systems for storing and querying graph datasets are often hindered by the fact that neither technology natively supports navigational primitives over the graph structure. For instance, evaluating simple path expressions requires costly joins between tables storing adjacency list data in a relational system. Native graph databases permit much faster execution of navigational primitives because they promote object relationships as first class citizens in their storage model. Moreover, they offer declarative access to the underlying graph via high-level languages like Cypher.

In this work we utilize graph databases as local worker nodes in a distributed system, *Digree*, which is used to manage large graph datasets. User queries, in the form of graph patterns are decomposed into smaller elements, utilizing the capabilities of the underlying graph databases to process path expressions. These expressions are executed in parallel by all worker nodes and their results are transmitted to a master node, where intermediate answers are consolidated in order to form the final result set to the user query. Key to the success of the proposed architecture are (i) the efficiency of the native local graph databases in processing path expressions and (ii) the increased parallelism offered by the query decomposition process that enables all worker nodes to contribute in evaluating a user expression.

In our prior work [1], we have formally described the query decomposition phase and the subsequent synthesis of the intermediate results and proven their correctness. In this paper, we first illustrate these processes using a simple running example. We then discuss a new greedy heuristic that leads to an efficient decomposition of a user pattern query. We additionally present new optimizations that we employ in order to expedite the execution of complex graph patterns. The first optimization termed early termination is used to detect when the distributed execution will return an empty result before all distributed processing is concluded. This is important since often query patterns cannot be matched against the data graph and early identification of such scenarios helps avoid unnecessary utilization of resources in the distributed system. The second optimization is used to push additional filters towards the local workers in order to reduce selectivity and, consequently, the sizes of intermediate results. We then describe the architecture of our *Digree* system prototype and compare its performance against a popular big data system extended to support graph pattern matching queries.

2. OVERVIEW

2.1 Data Model

The scale of modern graph datasets such as those encountered in social network applications, easily overwhelms single node installations. This necessitates multi-node deployments that partition the large graphs intro smaller chunks that are managed

*This research is financed by the Research Centre of Athens University of Economics and Business, in the framework of the project entitled 'Original Scientific Publications'

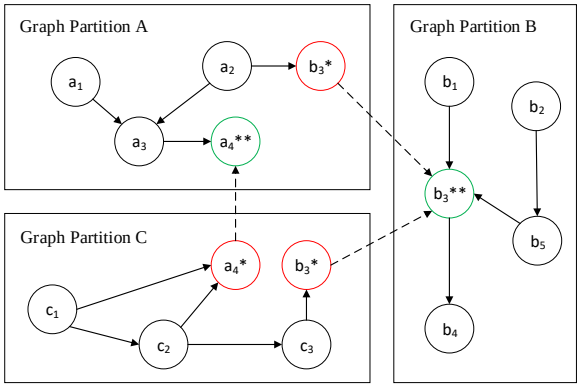


Figure 1: Example of a distributed Graph Dataset. A single asterisk by the name of a node indicates the special label *REF*. A double asterisk indicates the label *REFED*.

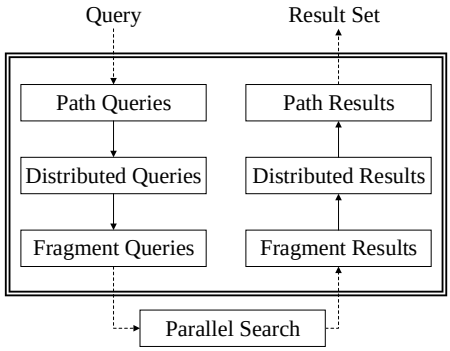


Figure 2: A high-level overview of the process we follow in order to answer a graph pattern query. Downward and upward directions respectively show the query decomposition and results combination processes.

by different servers [2]. Following this premise, *Digree* manages graph datasets that are partitioned across a number of worker nodes. Each graph partition is managed by a local graph database, which in our implementation is Neo4j. From an architecture perspective, *Digree* can utilize any graph database engine or combination of graph databases running at the worker nodes, as long as they implement a basic API for querying path expressions.

The graph data model that we employ is one of the most widely adopted models, namely the *labeled property graph model*, which is made up of nodes, relationships, properties and labels. *Digree* assumes that when a node u in graph partition GP_1 has an outgoing edge to a node v that exists in another partition GP_2 then:

- in GP_1 we maintain a local reference v' to v. We apply to it the label REF, indicating that this node is a REFerence to "remote" node v.
- all nodes in GP_1 that have outgoing edges to v are using the same single reference v'.
- at GP_2 we append to node v the label $REFED$, indicating that the node is REFErenceD by a remote node.

An example is shown in Figure 1. We note that the partitioning process is happening during ingestion of a new dataset and is orthogonal to the techniques we present here. Any partitioning algorithm, including dynamic partitioning techniques [2, 3] can be used as long as special care is taken in order to assign the aforementioned labels into the boundary nodes. These nodes can be located by following cross-edges between the graph partitions in a static scheme or when performing node migration in a dynamic partitioner [2]. Moreover, this labeling is performed at the system level in a manner that is transparent to the applications using the data that need not be concerned with the details of the graph partitioning layout.

2.2 Query Processing

Given a *pattern query*, i.e. a directed graph with vertices and edges possibly with labels and properties, the fundamental task is to find subgraphs of the database that are isomorphic (structurally and semantically) to the pattern query [4].

Digree takes as input a pattern query and decomposes it into smaller elements that are processed in parallel by the worker nodes. All computed results are shipped to a designated master node that combines the partial results to produce the global result set. Figure 2 presents an overview of the operations that decompose an input pattern query into smaller sub-patterns that are processed independently by the graph databases. The results to these sub-pattern queries are fused by *Digree* in order to compile the final result set. In what follows we use a running example to illustrate this process.

Taking as input the graph pattern query (from now on referred to as base query) depicted in Figure 3, *Digree* first decomposes it into a set of path queries. A possible decomposition consisting of two path queries is shown in Figure 4 and consists of path queries $pq1$ and $pq2$. Node B, depicted in orange indicates the location where results from these two path queries need to be "joined" in order to produce matching patterns to the input base query.

Next, we take an intermediate step where for each of the path queries we find out on which nodes

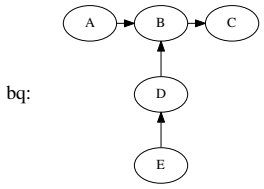


Figure 3: Running Example: The letters inside the depicted nodes are variable names so we can refer to specific nodes of the pattern query. Nodes may also have multiple labels and/or properties which we omit in this simplified example.

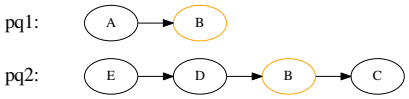


Figure 4: Two path queries obtained from the base query of Figure 3. Orange nodes denote locations where partial results need to be joined.

they should be "taken apart", in order to account for nodes that are possibly stored in different partitions. We refer to these nodes as break-points. This process forms a new set of queries that we call distributed queries and are shown in Figure 5 (the break-points are colored in blue). From path query $pq1$ we generate just one distributed query $dq1$, while path query $pq2$ gives us four different distributed queries, namely $dq2$, $dq3$, $dq4$ and $dq5$.

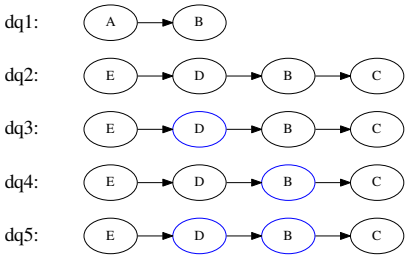


Figure 5: Resulting set of Distributed Queries. Nodes in blue are break-points. These nodes refer to possible locations where a path may be split and the resulting subpaths may be stored in different nodes.

As one can see, we take all combinations of break-point choices which, for every respective path query of length k, results in 2^{k-2} distributed queries, since break-points cannot exist at path start and terminal nodes. The results gathered for each distributed query should be unioned in order to generate the result set for the respective path query.

Each distributed query depending on the break-points it contains generates a number of fragment queries. These are the actual queries that will be submitted to the underlying graph databases. The fragment queries of our example are shown in Figure 6. Essentially, each distributed query represents a possible layout of the path it came from in the distributed setting. For example distributed query $dq4$ ($E \rightarrow D \rightarrow B \rightarrow C$), where the break-point is node B, aims to retrieve all instances of the path where the part $E \rightarrow D$ lies in one database while the fragment $B \rightarrow C$ lies in another one. Though, due to the data model, if such a result exists then node B should exist in both partitions, in the first labeled as REF and in the second as $REFED$. Thus, the fragment queries that will be generated will be $fq5$: $E \rightarrow D \rightarrow B^{REF}$ and $fq6$: $B^{REFED} \rightarrow C$. In the results combination process all fragment results for such fragment sets will be joined on their common node (in our example that one is B). One such joinable pair of fragment results for the example shown in Figure 1 would be $a1 \rightarrow a3 \rightarrow b3^{*}$ and $b3^{**} \rightarrow b4$.

One can see that there are duplicate fragment queries, e.g. $fq3$ and $fq7$. It is sufficient to execute just one instance of those and use the results for all instances.

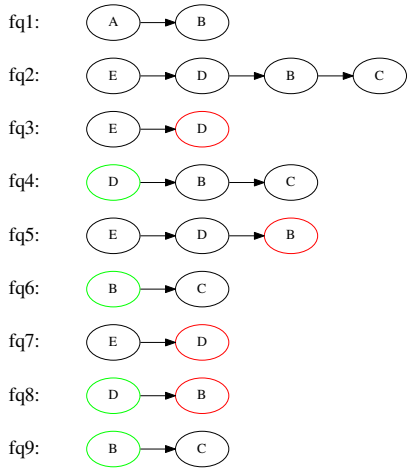


Figure 6: The Fragment Queries that will be submitted to the graph databases. For duplicate sets (such as $fq3$-$fq7$ and $fq6$-$fq9$) we submit just one query and reuse its results.

The full decomposition from the base query all the way down to it's fragment queries queries can be seen in Figure 7. Most of these tasks can be executed in parallel. Fragment queries are submitted to the worker nodes and when all fragment results that correspond to a distributed query's decomposition are gathered the computation of the distributed

query may start. Parallelization is also possible at the "higher" layer when all distributed results that are required to answer a path query are gathered.

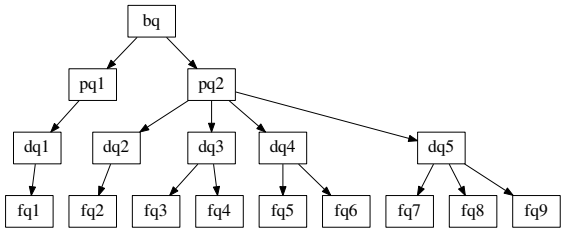


Figure 7: The decomposition mapping for Base Query into Path Queries, Distributed Queries and Fragment Queries (referring to Figures 3, 4, 5 and 6).

2.3 Key Intuition

While most of the approaches in the literature use some kind of decomposition, be it edge-level or subgraph-level of the user query and the subsequent combination of the partial results so as to answer distributed queries, there is a significant advantage in our setting. This is a result of the use of REF and $REFED$ labels at the partitions' boundary nodes. Due to the existence of these special labels we expect, and usually achieve, low selectivity while answering the fragment queries. Take for example fragment queries $E \rightarrow D$, $B \rightarrow C$ and $D \rightarrow B$ (Figure 6). These all contain nodes labeled either REF or $REFED$ and this not only allows for much more efficient execution of the query (based on available indexes at the local nodes) but may significantly reduce the number of results. Other systems that opt to get results for each edge of the input query and then join them using some execution plan [5] would execute the respective plain edge queries ($E \rightarrow D$, $B \rightarrow C$ and $D \rightarrow B$) and end up with a potentially much higher number of results to join at the next step. In our setting the query results that are local to a partition are collected by the execution of longer, more selective paths that contain them, e.g. local results for $E \rightarrow D$ are discovered by the execution of fragment queries $fq2$ ($E \rightarrow D \rightarrow B \rightarrow C$) and $fq5$ ($E \rightarrow D \rightarrow B$).

3. OPTIMIZATIONS

3.1 Path Queries Selection

If we consider the decomposition of the base query into path queries, we are faced with many alternative selections. Consider for example the base query from Figure 3. Instead of decomposing it into the

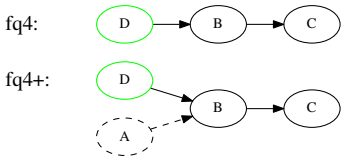


Figure 8: Fragment query $fq4$ and it's augmented version $fq4+$.

path queries $pq1$ and $pq2$ as shown in Figure 4 someone could decompose it into paths $A \rightarrow B \rightarrow C$ and $E \rightarrow D \rightarrow B$, a choice that would affect the further decomposition but also the efficiency of the execution. As the input graph query grows larger the number of choices is increasing and the cost analysis needed to select the best one is non trivial. The design and implementation of such a query optimizer is a work in progress of our own but preliminary results have shown that the system is favoured by the choice of longer path queries. Based on that, for the prototype of *Digree* we decided to use a heuristic algorithm that (i) creates an empty solution list, (ii) enumerates all possible (non-trivial) paths, (iii) adds them in a priority queue in decreasing order by their length, (iv) removes the first path from the queue and adds it to the solution list, and (v) until the base query is covered removes the next path from the queue and adds it to the list if it does not overlap with any of the paths already in the queue.

3.2 Early Termination

Because of the way that the queries are broken into path queries we can assert that if any of the path queries result set is empty, then the result set for the base query is also empty since it is computed by joining the path results. Before submitting the fragment queries to the partitions we sort them by shortest ancestor path query. That way, we are able to get path results early in the query execution and increase the chance to find out that one of them, and consequently the base query, has an empty result set. In that case an interrupt signal is sent throughout the system and the empty result set answer is returned to the user.

3.3 Fragment Query Augmentation

When fragment queries are submitted to the local nodes, they are augmented with additional structural conditions from the base query in order to help the underlying graph database system to answer the query more efficiently.

A node in a fragment query is in one of three states. It is either a REF node, a $REFED$ node, or a pure-local node. A pure-local node can be aug-

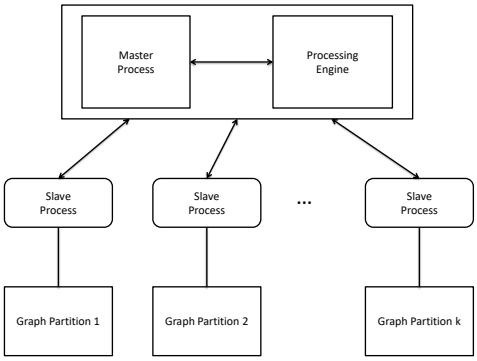


Figure 9: *Digree* architecture overview

mented by any other node (and the related edge) from the base query that is its neighbour, connected either by an incoming or outgoing edge since all of them should exist in the same partition. A $REFED$ node can be augmented by its outgoing edges neighbours since due to our data model all of them should exist in the same partition, but not by its incoming neighbours since at least one of them lies in another partition and we have no way knowing about it. Last, the REF nodes cannot augment since they are actually pseudo-nodes referencing a node existing in another partition.

For example, consider the base query in Figure 3 and the fragment query $fq4$ in Figure 6. Applying this technique, we can augment $fq4$ by the addition of node A and the respective edge $A \rightarrow B$, as shown in Figure 8. It is not possible to further augment $fq4$ by the use of node E since this is an incoming node to node D (the respective edge is $E \rightarrow D$) which, in the context of $fq4$, is a $REFED$ node.

4. SYSTEM ARCHITECTURE

Digree is designed as a distributed system that consists of two main processes, namely the master process and the slave process, and a main processing engine. A deployment should consist of a single instance of the master process, one slave process for each of the managed databases/partitions and one processing engine. The processing engine is a layer over a data management system (e.g. a relational database system, a graph database or a big data system) and handles the temporary storage of the partial results and the operations (union, join) that need to be applied to them. An overview of *Digree* architecture is shown in Figure 9.

Master process receives the user graph query and performs the decomposition all down to the level of fragment queries. Fragment queries are then submitted to the slave processes which are simple implementations of the managed graph databases API. In order to manage a different graph storage one needs to implement a new class of slave process so that it can communicate with it at least a simple path expression (query). When a fragment query is answered at a partition the slave process takes care that the results are transferred to the processing engine and also that a signal is sent to the master process. The master, depending on the signals it receives from the slaves, decides when a results combination process can start (e.g. union distributed results to compute a path result set) and triggers the appropriate operation at the processing engine.

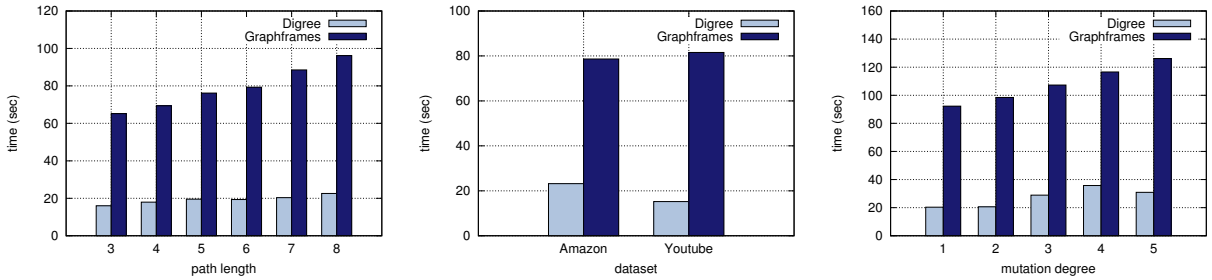
All *Digree* processes, including the processing engine, can be deployed in either in a single machine or in a distributed setting. The master and slave processes are implemented using the actor model in such a way that the operations on the collected partial results begin the soonest possible in a non-blocking fashion.

5. EXPERIMENTAL EVALUATION

We deployed our system on a cluster consisting of 18 Linux virtual machines (VMs). Each VM had 4 cores and 8 GB of memory. We used one VM to run the master process, one VM to host a PostgreSQL database server acting as the processing engine and the rest 16 VMs to host the partitions of the graph database (one Neo4j database per node) and the slave processes. We compared *Digree* to the motif finding feature of Graphframes [6], a package for Apache Spark which provides DataFrame-based Graphs. Graphframes was deployed on the same cluster using the default Spark setup. We used two real world datasets for our experiments which are the Amazon product network[1] [7] and the Youtube video graph.[2] The details of the datasets are shown in Table 1. For partitioning the datasets we used the popular METIS [8] algorithm. All measurements are made after the datasets have been loaded and partitioned on the respective systems. In [1] we present additional experiments running *Digree* on a much larger twitter dataset consisting of over 35 million nodes and 900 million edges, having 232 different labels. Graphframes however, in the aforementioned cluster, could not handle that dataset so we do no present the respective results here.

In the first experiment of Figure 10(a), we evaluate how each system scales while answering simple path queries of increasing length. For each dataset and for path lengths from 3 to 8 we created 5 graph queries of random labels. In the figure we report

[1]https://snap.stanford.edu/data/
[2]http://netsg.cs.sfu.ca/youtubedata/



(a) path length (b) datasets (c) mutation degree

Figure 10: Average execution time

Table 1: Datasets Overview

	#nodes	#edges	#labels
Amazon	548.552	1,788,725	11
Youtube	155,513	2,969,826	14

the average execution time of these queries. We also used seven graph pattern matching queries from [5]. For each pattern query we created 5 randomly labeled instances and ran all of them on the two systems. We present average execution times for each dataset in Figure 10(b). In all cases, *Digree* is significantly faster.

We then took the pattern queries from the last experiment and created a number of mutations for each of those. These mutations have been created by randomly choosing a vertex from the graph query and attaching a new vertex to it, randomly incoming or outgoing. We refer to the number of vertices added as the mutation degree. As shown in Figure 10(c), *Digree* outperforms Graphframes which presents a steady linear increase in execution time while *Digree* handles better the 5-degree mutation than the 4-degree one.

6. RELATED WORK

A number of systems were developed for distributed graph processing such as Google Pregel [9], Apache Giraph [10] and GraphX [11]. These systems are not specialized in graph pattern matching but instead provide a programming model to develop and deploy graph algorithms. In [5] the authors explore relational optimizations for graph pattern matching. The work of [2] also suggests bulding a distributed graph database out of local Neo4j installations. The authors propose a novel lightweigt dynamic repartitioner that increases data locality while maintaining load balance. This work is complementary to ours as it focuses on maintaining a good partitioning scheme in evolving datasets, while *Digree* focuses on parallel processing of complex query patterns over the resulting partitions.

7. CONCLUSION

In this paper we presented *Digree*, a distributed graph processing engine that exploits the efficient graph processing primitives provided by local graph databases, while at the same time benefits from the increased parallelism offered by the proposed query decomposition process. We have compared *Digree* against a popular big data system and shown that it consistently provides better performance.

8. REFERENCES

[1] V. Spyropoulos, C. Vasilakopoulou, and Y. Kotidis, "Digree: A Middleware for a Graph Databases Polystore," in *Proc. of IEEE BigData*, 2016.
[2] D. Nicoara, S. Kamali, K. Daudjee, and L. Chen, "Hermes: Dynamic Partitioning for Distributed Social Network Graph Databases," in *Proc. of EDBT*, 2015.
[3] I. Filippidou and Y. Kotidis, "Online and On-demand Partitioning of Streaming Graphs," in *Proceedings of the IEEE Big Data Conference*, 2015.
[4] B. Gallagher, "Matching Structure and Semantics: A survey on Graph-based Pattern Matching," *AAAI FS*, vol. 6, 2006.
[5] J. Huang, K. Venkatraman, and D. J. Abadi, "Query Optimization of Distributed Pattern Matching," in *Proceedings of ICDE*, 2014.
[6] A. Dave, A. Jindal, L. E. Li, R. Xin, J. Gonzalez, and M. Zaharia, "GraphFrames: An Integrated API for Mixing Graph and Relational Queries," in *Proceedings of GRADES*, 2016.
[7] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," *ACM Trans. Web*, 2007.
[8] G. Karypis and V. Kumar, "Analysis of multilevel graph partitioning," in *Proc. of IEEE SC Conf.*, 1995.
[9] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: A System for Large-scale Graph Processing," in *Proceedings of ACM SIGMOD*, 2010.
[10] M. Han and K. Daudjee, "Giraph Unchained: Barrierless Asynchronous Parallel Execution in Pregel-like Graph Processing Systems," *Proc. VLDB Endow.*, vol. 8, May 2015.
[11] R. S. Xin, J. E. Gonzalez, M. J. Franklin, and I. Stoica, "GraphX: A Resilient Distributed Graph System on Spark," in *GRADES*, 2013.

Dan Suciu Speaks Out on Research, Shyness and Being a Scientist

Marianne Winslett and Vanessa Braganholo

Dan Suciu
https://homes.cs.washington.edu/~suciu/

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 Dan Suciu, who is a professor at the University of Washington. Dan has two Test of Time Awards from PODS as well as Best Paper Awards from SIGMOD and ICDT. Dan's Ph.D. is from the University of Pennsylvania.

So, Dan, welcome!

Thank you.

Can we put a price on individuals' privacy?

Oh, that's tough. Today users give away their private information to Internet companies like Google or Yahoo for free, but this has to change. Users need to have control over their private data. So, it's not clear how this would happen, but at the University of Washington, we've started a number of projects that look into how to price data. In particular, we've looked at how to cover the entire continuum between free and differentially private data and paying the users the full price for access to their private data if they're willing to release that. It is still way too early to see what the right business model would be to allow users to monetize their private data. So far, we do studies in academia. We are still waiting for somebody to come up with the right business model.

We need to train the future scientists [..] in a way that will help them develop their careers for many years. For that, we need a good combination of both theory and practice.

I know it's a hard question but what can you tell me about how we should price it? What kind of methodology should we use?

Clearly, we need to allow users to opt-in and there are some technical challenges here because sometimes the user's decision of whether to opt-in or not might actually reveal something about their private data. This used to be one of the most technically challenging problems in pricing private data. In addition to that, the problem that we did address and that seems more within our reach is how to adjust the price according to the amount of perturbation that we add to the query. So, if the data analyst wants to have very precise data, then he would pay to have the perturbation removed. If he is willing to cope with differentially private data, then he would not pay, and then he would get differentially private perturbed query answers.

So what is my email address worth?

Ah, I think that kind of depends on you, and I don't think your email address is worth in isolation. Maybe it is, but you're part of a larger population. The question is, what is it worth to the analyst to get statistical information about that population? There is a game between users who would like to be compensated for access to their data and the analyst who would like to pay for access to aggregated data over a larger population. So, I don't have a good answer to this. I think that there are developing techniques that would allow the users to choose their price.

It's like there are two categories. There are the ways to contact me and then there are facts about me.

These are indeed different kinds of private information. I don't think we have thought too much about how to distinguish between these kinds of information. So nope, I can't tell you much more about this.

Channels vs. attributes so to speak. Okay! What about the temporal element?

Temporal element in data... you got me here. We know the techniques to deal with temporal data. Maybe you're asking about archiving data and how to keep it for a long time?

Well once something is out there, it's out there.

Ah, how to retract data?

Is that the solution? If I change my mind, update it or whatever?

That is again, a difficult technical challenge. I know there has been research, not inside the database community, but in the operating systems community. There was a project run by some of my colleagues called Vanish[1] that allowed people to produce data that would completely vanish from the cloud after a few hours. So the idea is that you can send an email to your friend, but this email is private, and then it will completely disappear. Every trace of this email will completely disappear from the cloud after a few hours that your friend has read this email. It's a technically challenging problem. Now we also see legislation in Europe that tries to force companies to remove data. I think the jury is still out for what the right model is -- if the data should be kept forever and how much should the users have control over when their data is being deleted.

[1] The VANISH project. https://vanish.cs.washington.edu

Differential privacy also gets more difficult when there's a periodic release of aggregated information compared to a one-time release.

That's a major limitation of differential privacy. They talk about a privacy budget. You can only ask as many queries as is allowed by a privacy budget. There is no story of what happens after this privacy budget has been exhausted. So, then the analyst should theoretically never have access to the data because he/she has exhausted the privacy budget. There is some hope if you also take into account data churning. The data is never static. It always gets updated. Old data is being removed, and new data is being added. So, then your privacy budget would maybe be renewed, but this is still work in progress, and this is not something we do in our group. In our group, we try to solve the problem by allowing users to place a price on this data and that will simplify the privacy budget because now the budget is really a monetary budget (as much money as you have). This is how much you can get access to the private data.

Ok great. Tell me about the work that you received the Test of Time Awards for.

The Test of Times in PODS? There were two papers. Both were about XML. The first was on type checking XML transformations[2]. Here the question that we asked was: If you are given the schemas for your input data and for your output data, can you automatically check if a given XML transformation will indeed map every input data conforming to the input schema to an output data that conforms to the output schema? It turns out this is decidable. You can do this for a non-trivial fragment of XPath. Today XQuery does it differently. It uses type inference, which is not a complete decision procedure. We had a complete decision procedure for a more restricted fragment. So, this was one work.

The other[3] was for a very simple and fundamental problem, which is, you're given two XPath expressions and you need to check if they're equivalent. They may be syntactically different perhaps, but are they actually semantically equivalent? We looked at the tiniest fragment of XPath that is mathematically very elegant to define. That fragment only has "*" (wildcards), "//" (slash slash) and predicates (the "[]" in XPath). What was funny was that for any combination of two of these features, it was known before that checking equivalence could be done efficiently in polynomial time. What we showed is that when you throw in all three features, then the equivalence problem becomes co-NP-complete. And that was an interesting insight because it tells you that XPath is as difficult to check for equivalence as arbitrary conjunctive queries, for example.

Interesting. So both of those papers were on XML, and after that, you moved to working on privacy and then to probabilistic data and from probabilistic data to data markets. How do you choose your next topic and the timing of the move?

This is actually quite hard, but as researchers, we need to watch technology trends and application pulls. The world changes. So in the 90s, the web was new, and for the first time, people discovered that they could share data. They could exchange data. XML was actually designed by people working as a document community. So, they designed XML thinking of it as a document. I think the database community should be credited for showing how XML should be thought of as data and as a data exchange format. The research problems that emerged were fun, but they were not particularly deep. I would say that they are largely solved by now.

But in the meantime, we got new challenges. People started to realize that by exchanging data and having access to data, you need to worry about data privacy. Also, much of the data is uncertain, so data privacy and probabilistic databases emerged later as new challenges for data management. The technical questions underlying these challenges turn out to be much harder. None of them is solved. I actually have my doubts whether privacy can ever be solved in the way in which the academic papers present it. I am a little bit more hopeful that adding a price to the data might lead to a practical solution. For probabilistic databases, they are equally technically challenging, but at least here we're not the only ones looking for solutions. The knowledge representation community and the machine learning community are very hard at work at trying to solve the same challenges we face in probabilistic databases, which is probabilistic inference.

There aren't any commercial probabilistic database systems yet. Do you see a killer app that will bring them into practice?

This is a good question. The most interesting application that I see is that of information extraction. There is a lot of data out there, but it is not in a query-able format. Information extraction is by now a mature field that takes unstructured data and converts it into

[2] Tova Milo, Dan Suciu, Victor Vianu. Typechecking for XML Transformers. PODS 2000, pp. 11-22.

[3] Gerome Miklau, Dan Suciu. Containment and Equivalence for an XPath Fragment. PODS 2002, pp. 65-76.

some structured format, but the output is often probabilistic. The information extraction tools are never 100% accurate, and they produce relational data that is probabilistic. A large example that I know of is Google's Knowledge Vault, which is separate from Google's Knowledge Graph. So, Knowledge Vault has about two billion tuples (two billion triples), and they're all probabilistic. The probabilities of these triples are pretty much uniformly distributed in 0-1. So you can find almost certain triples, and you can find almost junk triples inside, but there's a lot of rich information in that data that, hopefully, with techniques from probabilistic databases, we can query. There are other applications beyond that like record linkage, de-duplication, querying anonymized data that can be viewed as a probabilistic database. Actually, the other day, I was listening to a SIGMOD talk[4] that described a very unexpected application of probabilistic databases. They wanted to do bootstrapping, and they realized that in order to do bootstrapping for some SQL queries you don't need to resample hundreds or thousands of times, and re-run the same query over and over again. Instead, you can view the original data as a probabilistic database and then simply run the query as if you were running it over a probabilistic database. It's a very interesting and unexpected application of probabilistic databases, and I think we'll see more of these in the future.

Someone commented to me that in each of your projects you combine theory and practice perfectly. Is this always one of your goals?

Yes, it is. I believe that in academia this should be a major goal. We need to train the future scientists, and we need to train them in a way that will help them develop their careers for many years. For that, we need a good combination of both theory and practice. I also find that the most difficult theory questions are those that are grounded in practice and that the most interesting systems are those that have a strong theoretical component. I can justify research that is purely theoretical or purely systems oriented, but the most interesting ones that I saw are those types of research that combine both.

Does Big Data have any formal foundation?

Ah, Big Data does need a formal foundation, but I don't think that the jury has settled yet. The industry describes Big Data through the three V's: high volume, high variety, high velocity, but this does not set the research agenda for our community. We have dealt with these qualities of the data since the beginning of databases. We have dealt with volume. Variety means essentially semi-structured data and other data formats, and velocity, well, data streaming has addressed velocity for more than ten years now. So, we need to look for different attributes of Big Data that justify and inform our research. In my view, these attributes would be: (i) massive parallelism -- how do we do query computation over hundreds or thousands of servers; (ii) extending query languages, adding recursion, or adding some capabilities to deal with linear algebra in addition to standard relational query processing over large data; and (iii) the third attribute would be less well defined, but people should look for some kind of techniques or tools that would allow users to explore the data, maybe to understand it, maybe to find explanations, maybe to approximate query processing over very large data, or maybe produce graphs quickly and analyze and interact with those graphs. So these are the three attributes that I think should inform researchers on Big Data in our community.

[…] the most difficult theory questions are those that are grounded in practice and […] the most interesting systems are those that have a strong theoretical component.

For that second one: adding linear algebra. It sounds like you're moving a little in the direction of SAS or other statistical packages. Are you talking about turning it into a data mining type of thing? And you also said on top of SQL. You didn't say on top of MapReduce.

No, I really meant on top of SQL, on top of relational languages. I think they are here to stay. People have tried to replace them for many years. They are founded in something very principled, which is first-order logic and they're just irreplaceable. So, to this, we need to add the capabilities that people need for large-scale analysis today, which is dealing with linear algebra, doing some machine learning, or some statistical processing. It's interesting, some of the research papers, in fact, the best paper in SIGMOD this year (2014), specifically addresses data management issues

[4] Kai Zeng, Shi Gao, Barzan Mozafari, Carlo Zaniolo: The analytical bootstrap: a new method for fast error estimation in approximate query processing. SIGMOD 2014, pp. 277-288.

in System R[5]. So here we go. We feel the need to integrate linear algebra operations in data management.

What kind of data exploration tools do you have in mind?

We put our money on causality and explanation. We want to allow users to ask questions like "why does this graph have a dip here?" Or "why does this graph increase when I was expecting it to decrease?" The system should hopefully either find some records in the database that are most likely to explain that behavior or find some predicates, some groups of records (some classes of records) that explain that behavior.

Can we differentiate between correlation and causality?

There is a huge debate about this, and essentially the consensus is that you cannot deduce causality from correlations, but people have moved past this general principle. Judea Pearl wrote a very influential book[6] about 15 years ago on causality, and essentially the approach is that you would write down the causal path in your problem. In Judea Pearl's book, causal paths are like graphical models, but in databases, it is even easier to see what they are. They are like foreign keys. Every foreign key can be thought of as a causal path. If you remove the record to which this foreign key points, then implicitly you have said that you have removed the record that has a foreign key to that record. So, keys and foreign keys already give us some causal paths for free. Can we use these and maybe some other forms of causal paths to find explanations to observations on the data? That's where we are heading our research.

How does query optimization change when you're running over thousands or hundreds of thousands of servers?

That's a very interesting theoretical problem because traditional query processing has defined query complexity (the cost of running a query) in terms of disk IOs. A query that uses fewer disk IOs is better than one that uses more, or a query plan that is cheaper in terms of disk IOs is better than another query plan. But now, when we use hundreds or thousands of servers in order to do a complex data analytics computation, we often do this because we want the entire data to fit in main memory. So, disk IO is no longer the main bottleneck, but the new bottleneck is the communication. How much data do we need to communicate in order to compute that query? It's a completely new metric, and this requires some interesting research to understand the inherent lower bounds on the communication complexity for computing the various queries. This is where we have done our research, and some of that research is complemented by algorithms that other people have found. For example, there is a class of algorithms that was described by Afrati and Ullman in a paper[7] about four years ago, and they are good matches for the lower bounds that we find using our theoretical analysis.

So, the query optimization crowd worried about communication back in the old days of distributed databases, when that was the highest cost for answering a query. So, for the second time around, is it fundamentally different?

I think it is fundamentally different if you run your query on 10 or 20 nodes or whether you scale it up to 500 or 10,000 nodes. You would ask the question: If I increase my number of servers from 500 to say 10,000, will I see a benefit? Well, we need to understand how the amount of communication depends on the number of servers involved in answering that query. We don't have these answers yet. Actually, I would claim we start to have these answers as a result of this theoretical work. These are the kinds of questions that are new on this framework: the dependence on a large number of servers that we did not have before in the 80s when the first parallel databases were developed.

Although the HPC community has always thought of the cost of computation as whatever you need to do in memory, and the communication, and if you have to read stuff from disk which they have to avoid like the plague, there's nothing new under the sun. But it's a new problem. They don't run parallel database queries -- different types of analysis.

So, the relationship is interesting because linear algebra is mostly about dense matrices or dense vectors. Databases are all about sparse matrices. Every binary relation can be thought of as a very sparse matrix. It's interesting to see that the techniques developed by the HPC community, they are absolutely the best when they deal with dense matrices. If you try

[5] Ce Zhang, Arun Kumar, Christopher Ré. Materialization Optimizations for Feature Selection Workloads. SIGMOD 2014, pp. 265-276.

[6] Judea Pearl. Causality: Models, Reasoning and Inference. Cambridge University Press, 2nd edition, 2009.

[7] Foto N. Afrati, Jeffrey D. Ullman: Optimizing joins in a map-reduce environment. EDBT 2010, pp. 99-110.

to blindly apply joins to do a matrix multiplication, the performance will be way beyond that developed by people in the HPC community. Conversely, when you're dealing with sparse data, the join processing techniques that we know in our community are just unbeatable. They are the best. It's interesting to combine these two, and probably we just need to add these two different solutions together into a unified system.

> ***Rejections from SIGMOD and PODS don't mean that your work is lousy, it perhaps means that your work is ahead of its time. Eventually, good work will be published.***

Sounds interesting. Three of your Ph.D. students have been runners-up or recipients of the ACM SIGMOD Dissertation Award. I want to know how you make that happen!

I want to know too! I was just lucky to work with very talented people. Some of my current students are equally talented, so I just feel lucky to have them and look forward to working with more people like them. I really don't have a recipe. I think it's just a matter of finding the right people.

So how do you find those people that you hire?

Um, to my shame, I should acknowledge that they found me. I did not find them. So, if I look back, I'm not good at recruiting people. I think I try to get people enthusiastic about the research that I'm doing and that is where my role stops, and it's very difficult. I really don't have a recipe. I also have very limited data points. I have only had maybe 15 Ph.D. students in my entire career. I don't know if this is an accurate number, but that would be the ballpark, and every case is different.

So, you're not involved in the admission space. They find you after they've been admitted.

I am involved in the admission space, but I'm a very lousy predictor of the performance of a student just based on their application.

It can be very hard for a shy young person to get out there and talk to others, especially at a big event full of strangers, like SIGMOD where we are today. But networking is important for almost any kind of career. You just mentioned a minute ago that you've been so busy meeting with people you haven't had time to hike in our beautiful surroundings here. So how have you dealt with this issue of overcoming shyness with strangers in your career?

I'm personally very shy, and I had difficulties at the beginning of my career. My advisor helped me a lot, and I think it's the duty of any advisor to help their students get over their shyness. Also, it gets better once you have research results. When you have a paper that you really care about, then you are really eager to tell the world about it, and then the shyness is less of an impediment. You just go out there and tell people about your great results. It's just things that people have to cope with. With help from the advisor and with…

How did your advisor help you?

I'm trying to remember… I think he just introduced me to people. It wasn't terribly effective, but maybe in 1 out of 10 cases the person he introduced me to would be actually interested in my research, and that is when I would take the opportunity to explain my research. I should also mention that explaining one's research is really important. I see this with my students all the time. They don't learn from me how to best explain their research because I know what their research is and I value it. They need to go out and try to explain it to other people, and then they discover that the language that they use and the way they explain their research is often the wrong way to do it. So then they need to adjust it and try again with the next person to explain their research. So, it's a very important experience to try to advertise your own work to other people and find the best way to describe it.

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

I think the question that anyone should ask is why am I in the game? Do I do science because I want to be a scientist? Or do I do science because I like to do science? In the first case, if you do science just because you enjoy the status of a scientist then maybe this is the wrong thing to do. Maybe you should search a different career and become a much more famous person in industry or in some other kind of career where you can make much more money than you can do in science. If you really care about the scientific questions, then this is the right place to be.

Also, ignore all those rejections. Rejections from SIGMOD and PODS don't mean that your work is lousy, it perhaps means that your work is ahead of its time. Eventually, good work will be published, and then it will be much better recognized if you stuck with your ideas despite the fact that the community had a difficult time accepting those ideas.

Did you ever find yourself have to wait a long time before something finally got accepted?

Oh yes, it was very difficult to publish papers on probabilistic databases. The community was just not ready to accept this model. So, I had a low point especially when I was advertising and when I was working on the early papers on probabilistic databases.

Is there any reason that the community turned a corner or it just happened? Did something happen to make it move in that case?

I think two things happened. One is that our community got more used to accepting the need to cope with probabilistic data. And the second is that we found a better way to explain the connection between the probabilistic data model and other probabilistic models that had been around for a longer time, like graphical models and more recently, statistical relational models.

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

Yes, I do. That is still within the area of probabilistic data. It is a dichotomy theorem that tells you that every union of conjunctive queries can either be computed in polynomial time over a probabilistic database or it is #P hard to compute it over that probabilistic database. The reason why I'm really fond of this result is because it is inevitable. You need to know this if you want to do any kind of probabilistic inference. I mean this not only for database researchers but also for researchers in the knowledge representation as they become aware now of this result. It is simple, but not obvious at all. The criterion that makes a query to be computable in polynomial time not obvious, it's difficult to guess it, but once you see it, it's actually quite simple and straightforward. Moreover, the result itself was very hard to prove. It took us four years to prove the theorem, and I'm not proud that it took us so long, but I'm really happy because of the result in the end.

And where did that appear?

It appeared in the Journal of the ACM in 2012[8].

Was there a conference version before that?

There was a conference version in PODS 2010[9], but the JACM version is much more complete.

[…] the question that anyone should ask is why am I in the game? Do I do science because I want to be a scientist? Or do I do science because I like to do science?

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

Ah, I would hack. I enjoyed hacking in the past, but I didn't have any time to do hacking in the last several years. I would play more with these new fancy tools: IPython, Python, R, or Matlab. I would try to extend my knowledge of data management in the new direction.

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

I would learn more linear algebra. When I learned linear algebra in high school and in college, I was already passionate about programming, and I saw no connection between linear algebra and the cool things you could do with programs. But now linear algebra is a centerpiece of computer science and modern research, and I think the database research community needs to adapt and integrate linear algebra in its research.

Thank you very much for talking with me today.

Thank you, Marianne.

[8] Nilesh N. Dalvi, Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. J. ACM 59(6), pp. 30:1-30:87 (2012).

[9] Nilesh N. Dalvi, Karl Schnaitter, Dan Suciu. Computing query probability with incidence algebras. PODS 2010, pp. 203-214.

Data Quality – The Role of Empiricism

Shazia Sadiq
The University of Queensland, Australia
shazia@itee.uq.edu.au

Tamraparni Dasu
AT&T Labs-Research, USA
tamr@research.att.com

Xin Luna Dong
Amazon, USA
lunadong@amazon.com

Juliana Freire
New York University, USA
juliana.freire@nyu.edu

Ihab F. Ilyas
University of Waterloo, Canada
ilyas@uwaterloo.ca

Sebastian Link
The University of Auckland, New Zealand
s.link@auckland.ac.nz

Renée J. Miller
University of Toronto, Canada
miller@cs.toronto.edu

Felix Naumann
Hasso Plattner Institute, University of Potsdam, Germany
felix.naumann@hpi.de

Xiaofang Zhou
The University of Queensland, Australia
zfx@itee.uq.edu.au

Divesh Srivastava
AT&T Labs-Research, USA
divesh@research.att.com

ABSTRACT

We outline a call to action for promoting empiricism in data quality research. The action points result from an analysis of the landscape of data quality research. The landscape exhibits two dimensions of empiricism in data quality research relating to type of metrics and scope of method. Our study indicates the presence of a data continuum ranging from real to synthetic data, which has implications for how data quality methods are evaluated. The dimensions of empiricism and their inter-relationships provide a means of positioning data quality research, and help expose limitations, gaps and opportunities.

1. INTRODUCTION

Effectiveness and efficiency have been critical to the success of data management, data integration and data analytics technologies over the years. Effectiveness ensures that the result serves the purpose for which it was obtained, while efficiency ensures that the process of obtaining the result does not waste critical resources. Obviously, one without the other, while possible, is not desirable, especially in this age of Big Data, where critical decisions need to be made correctly and quickly.

Empiricism postulates the fundamental role of experiments and measurements in the advancement of science [33]. Historically, it has been very important to improving the efficiency of data management technology. The TPC family of benchmarks (tpc.org) has contributed to measuring and continually improving the efficiency of data management technology over several decades. For example, the TPC-C and TPC-E benchmarks measure the performance of on-line transaction processing applications, and the TPC-H and TPC-DS benchmarks measure the performance of decision support systems. Although the focus has been mostly on relational or structured data, efficiency-oriented benchmarks exist for non-relational data models too. Recently, the TPC benchmarks have been expanded to consider the efficiency of data integration (TPC-DI) [39] and big data processing [9].

As long as one assumes that the input data are trustworthy and of high quality, and the transformations performed on the input to produce the result are well understood and match expectations, one can happily regard the result obtained as being of high quality as well. In the big data world, however, data sources are not always trustworthy, and complex, ill-understood pipelines are used to transform the data. Consequently, the quality of the results should be

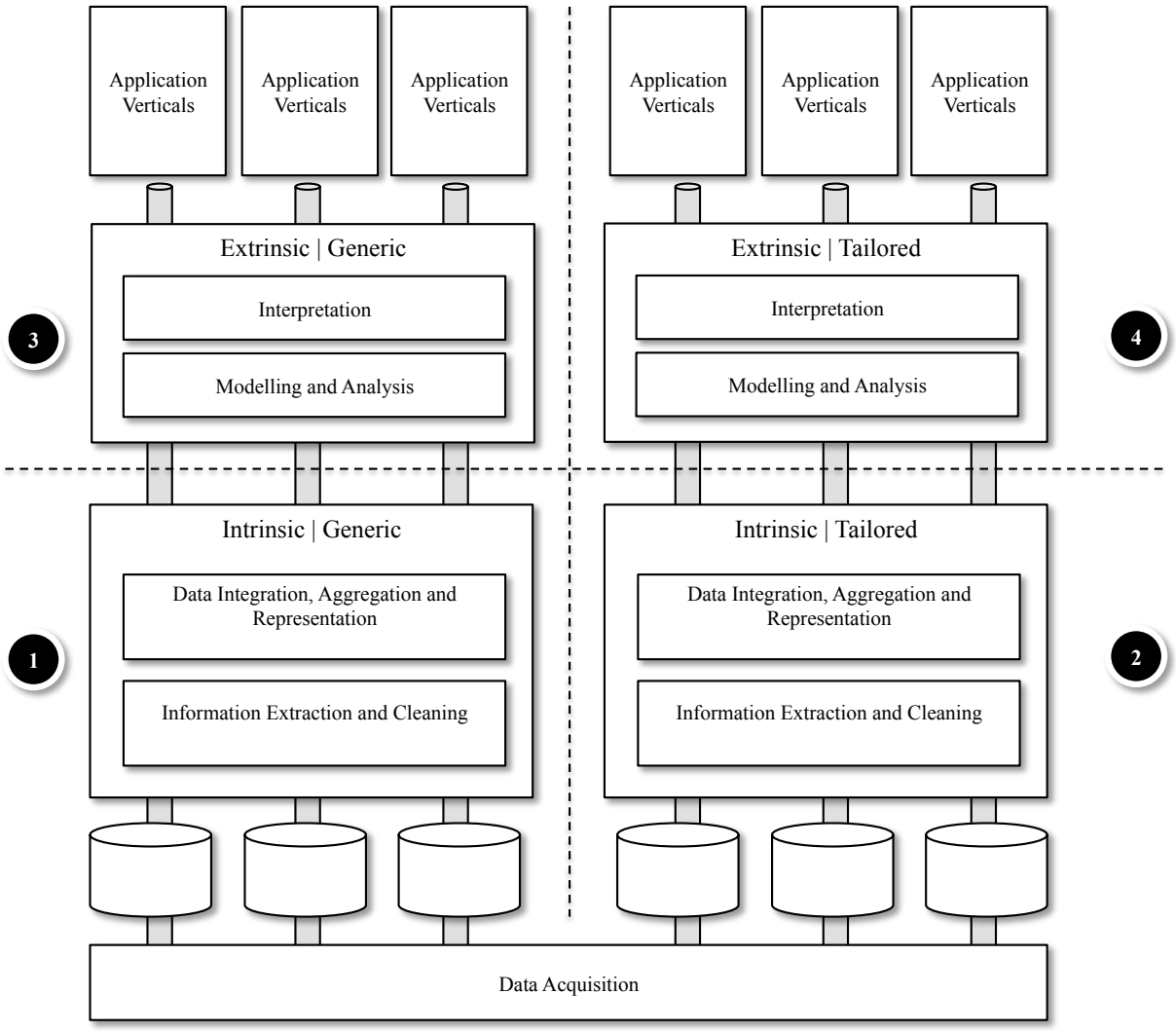


Figure 1. Typical Data Processing Pipeline

viewed with the appropriate level of skepticism. Being able to empirically evaluate the trustworthiness of the data sources and effectiveness of the data processing pipelines, and hence the quality of the obtained results, would go a long way towards ameliorating this undesirable situation. However, it is not immediately evident which aspects of the pipeline contribute more significantly to an authentic empirical evaluation.

In 2015, a group of global thought leaders from the database research community outlined several grand challenges in getting value from big data [3]. A key message was the need to develop the capacity to "understand how the quality of data affects the quality of the insight we derive from it". The role of data quality is recognized as pivotal to the effectiveness of data pipelines.

The notion of quality is highly contextual and tied deeply to fitness for use [25]. In determining the effectiveness of these pipelines, it therefore becomes critical to evaluate the fitness of the data for its intended use. Similar to how TPC benchmarks help measure efficiency in data management; empirical evaluations of data quality will help measure the effectiveness of data pipelines. However, balancing the purposefulness (depth) of data quality detection and cleaning methods with their capacity for wider applicability (scope) [17] remains a challenge.

In this paper, we identify two inter-related dimensions of empiricism that help locate the sweet-spot for empiricism in advancing data quality research and practice. These are the *type of metric*, and the *scope of method*. We explain these dimensions of empiricism in the next section.

While type of metric and scope of method have direct implications for the technology stack that implements a data processing pipeline (see Figure 1), a third aspect,

namely, the *nature of the data*, exposes a data continuum that defines the setting in which the data quality metrics and methods can be evaluated. In Section 3 we outline the data continuum and discuss the properties of real data, synthetic data and everything in between.

In Section 4, we present the various ways in which the dimensions of empiricism can be positioned, thus providing a lens through which the role of empiricism in data quality research can be studied. In order to gain a deeper insight into each of these positions, we reached out to thought leaders in data quality research [44, 45] to help elaborate on the motivation and rationale, key approaches, and possible challenges against each position. The viewpoints presented are extracted from a series of interviews conducted with the experts and are supplemented with a review of relevant literature.

Finally, in Section 5, we present a set of recommendations on promoting empiricism in data quality research and practice. These recommendations have been synthesized from the findings reported in this paper.

2. DIMENSIONS OF EMPIRICISM

Figure 1 presents a typical data processing pipeline from acquisition to analytics. The components of the pipeline have been extracted from [23], and include five steps of the data processing pipeline, namely, (i) Data Acquisition, (ii) Information Extraction and Cleaning, (iii) Data Integration, Aggregation and Representation, (iv) Modelling and Analysis, and (v) Interpretation.

Data quality considerations are rooted throughout the pipeline, from the time data are acquired, through various transformations within the pipeline, to their eventual interpretation for a given application.

Figure 1 presents four quadrants that represent four distinct positions where type of metric and scope of method influences the way in which the data processing tasks are handled. The background to these positions is introduced below and the positions are further detailed in Section 4.

2.1 Type of metric

Prior works have identified many metrics to measure specific data quality characteristics [54], such as completeness, timeliness, consistency, etc. Essentially, metrics can be intrinsic or extrinsic to the characteristics of the data [24].

- **Intrinsic metrics** are application-independent, and can be declaratively defined and measured, such as the format-consistency of a date/time attribute. Intrinsic metrics are expected to be handled in Quadrants 1 and 2 in Figure 1.
- **Extrinsic metrics**, on the other hand, are application-dependent, such as the fidelity of a specific analytical report. Thus, extrinsic metrics are exposed and managed in Quadrants 3 and 4 of Figure 1.

Even though intrinsic metrics can typically be implemented without reliance on external reality, there are some caveats to the assumption. For example, timeliness of event data can be measured from the update logs, however the notion of timeliness (comparing to the time at which the event occurred in reality) relies on external reality. Similarly, completeness can have multiple interpretations. For example, null values for mandatory attributes can be counted without reference to an external source, but missing records require reference to a trusted external source, such as master data. Missing records can also be application-dependent, for example a public transport dataset may be complete for city planning but incomplete for scheduling [41, 42].

Regardless of whether a metric is intrinsic or extrinsic, based on rules or statistical notions, it has to be tied to the underlying data to be relevant to measure data quality.

Thus, whereas intrinsic metrics are focused on the properties of the data only and not on how applications use it, the aim of intrinsic metrics is indeed to eventually contribute to achieving an extrinsic metric. An extrinsic metric also has to be tied to the underlying intrinsic metrics of data quality, even if its measurement is application-dependent. The **part-of** (or **aggregation**) relationship between an intrinsic and extrinsic metric is thus rather nuanced. The value of the data (an extrinsic metric) may be composed of not only multiple intrinsic metrics, such as completeness, lack of duplicates, format consistency etc., but also the relative importance of each metric for the task at hand.

For example, consider postal delivery services, wherein the completeness of customer address can be considered an intrinsic metric of customer data quality. At the same time, there can be an extrinsic metric on how well the data supports the business objective of priority deliveries to be completed within 72 hours. The two are clearly inter-related (i.e., incomplete customer addresses can result in delivery delays), but the relationship needs to be defined and measured in a precise way to demonstrate how the investment on making customer address data complete, impacts on the reduction in delayed or failed delivery attempts.

A data quality (detection or cleaning) method may be designed to optimize metrics of either intrinsic or extrinsic type. However, as shown in the examples above, in empirical evaluations of data quality systems, both intrinsic and extrinsic metrics need to be considered together, along with their possible dependencies.

2.2 Scope of method

There have been significant contributions from research and practice towards developing methods that assist in various phases of data quality management, including methods for detection, assessment, and repair of data quality problems [46]. Further, a variety of approaches have been proposed towards the design of these methods, for example interactive [29], exploratory [14], and autonomous [1, 4, 10, 22, 43] approaches. The scope of these methods can range from being tailored for specific use-cases, to being generically applicable.

There are two aspects that significantly influence the scope of the methods: (1) the type of data, for example structured, text, graph, etc., and (2) the application domain for which the data quality methods are being designed and developed.

- **Generic methods** can be reused in a variety of application contexts or applied to a number of data types, for example detecting similarity through tokenization and set similarity measures can be applied to strings [51], records [16], and videos [32]. As such, generic methods target problems that emerge in Quadrants 1 and 3 of Figure 1.
- **Tailored methods**, on the other hand, are specific for a particular data type or application domain, for example improving the usability of RDF data [2]. Tailored methods are developed to handle problems relating to Quadrants 2 and 4 of Figure 1.

The scope of a method will influence the design of evaluations and consequently the way in which the results of the method can be utilized. The scope of the method is independent of the type of metric. For example, a duplication detection method for relational data can be measured from both intrinsic and extrinsic perspectives.

We observe further that tailored methods can be considered as a **specialization** of a respective generic method, that is, specialized for a certain application domain or data type. Whereas traditional methods have relied on well-defined constraints and design principles, e.g., functional dependencies and normalization process for relational data, in the big data scenario these constraints are largely unknown, and data of different types can be integrated and re-purposed for different applications. This makes it difficult to navigate the spectrum of methods from applicable (generic) to purposeful (tailored), indicating a need to better understand how large collections of tailored methods (e.g., duplicate detection for specific data types) can be generalized for wider applicability.

3. The Data Continuum

Data quality research and practice have been empirically evaluated with both real and synthetic data. Synthetic data can be created through a perturbation of real data that represents ground truth [36]. Alternatively, synthetic data can be entirely created through a data generator that mimics real data properties and/or through the design of a generative model by learning parameters from real data [11, 47].

Both real and synthetic data can be of a variety of data types, such as structured or unstructured, streaming or historical etc. However, there are some key features that distinguish the two:

- **Real data** are data created in the 'wild', where there is little or no influence on its generative process from the data quality method being studied. Real data provide both meaning and impact to data quality research. However, the overheads, technical and legal, in the acquisition of real data can sometimes be prohibitive. Further, in using real data to test the efficacy of a system, ground truth is not always available or may require considerable time investment to create.
- **Synthetic data**, on the other hand, are created with specific schema and data characteristics in mind. More importantly, ground truth can be easily manufactured for synthetic data, whereas it is not readily available for real data.

In general, the absence of ground truth for real data is considered an impediment in measuring the effectiveness of data quality methods. Thus, differentiating between the discovery of actual and spurious data characteristics [27] becomes difficult. Recent work on using crowd sourcing to establish ground truth for real data has helped alleviate this problem to some extent [53].

Synthetic data can provide fertile ground to study specific problems of data quality relating to accuracy (availability of ground truth) as well as performance (large volume). However, synthetic data may not adequately capture the characteristics of the problem domain the data are supposed to represent.

Table 1. Contrasting Positions

	Generic	Tailored
Intrinsic	IG	IT
Extrinsic	EG	ET

Hence, the authenticity of the results obtained on synthetic data may be questioned. Note that one may argue that real data can also suffer from similar shortcomings attributed to design decisions made at the time of real data collection.

We note that synthetic data are an **abstraction** of real data with certain well-defined properties that help to remove unnecessary complexities and provide a controlled environment for the study of specific data quality problems. Whereas real or wild data start off as being rather opaque, through various transformations, annotations, and/or creation of ground truth, they start to become more transparent.

The process of acquiring metadata whether by profiling, or talking to experts, or augmenting with other data sources, also enhances the understanding of the data and moves the data toward the transparent end of the scale.

The process of tackling the problem of opaqueness of data, especially in the absence of ground truth, is challenging and currently under-studied. There is a need to understand the implications of these abstractions of real or wild data towards the creation of curated real data or generated synthetic data, and how these abstractions impact the overall authenticity of the data pipeline.

4. CONTRASTING POSITIONS

A number of contrasting positions emerge from the dimensions discussed above. Table 1 presents a summary of these positions relating to type of metric (I: Intrinsic and E: Extrinsic) and scope of method (G: Generic, T: Tailored), corresponding to the respective quadrant of the data processing pipeline shown in Figure 1.

Note that the four positions are designed as an aid for discussing properties of data quality methods and their evaluations. It is possible, and indeed desirable, to conduct an empirical evaluation that considers both intrinsic and extrinsic metrics, considering the use of the method in both a tailored and in a more generic setting or scope, and using the data continuum from real to synthetic data.

Table 2. Relevant Papers

Position	Papers
IG	Discovering Meaningful Certain Keys from Incomplete and Inconsistent Relations [28] Data Anamnesis: Admitting Raw Data into an Organization [30] Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources [15]
IT	Quality-Aware Entity-Level Semantic Representations for Short Texts [20] Data Quality for Temporal Streams [13]
EG	Effective Data Cleaning with Continuous Evaluation [21] Benchmarking Data Curation Systems [6]
ET	Exploring What not to Clean in Urban Data: A Study Using New York City Taxi Trips [18]

Nonetheless, these positions and their inter-relationships present a means of interrogating the body of knowledge on data quality and allow us to expose the role of empiricism in data quality research and practice.

In Table 2, we present a list of papers published in a recent special issue of the Data Engineering Bulletin [45] that focused on empirical research in data quality management. The list is not meant to be exhaustive, but an exemplification of the positions discussed below.

4.1 IG: Intrinsic and Generic

Generic methods for intrinsic metrics are positioned within Quadrant 1 of Figure 1. Several intrinsic characteristics of data, such as duplicates [37] and anomalies [7, 12], are generalizable across many uses of the data. In fact, any numerically representable profile of the data [8] can be generically understood and reasoned with. Thus, generic methods can handle data quality management for various data processing tasks such as extraction, integration, and aggregation (see Figure 1).

Often more sophisticated or tailored metrics use these generic methods as building blocks [48]. Re-use of such generically applicable methods prevents re-invention and more importantly provides a uniform way to compare extensions in tailored methods. Promoting the re-use of generic methods based on intrinsic metrics leads to knowledge sharing, limits unnecessary re-invention, and should be encouraged.

The exposition of intrinsic generic metrics when applied to real data can get buried in the contextual details of real data, making the results difficult to share and re-use. Whereas there are no obvious distinguishing aspects of using synthetic data for evaluating intrinsic generic metrics, the separation of the scope of the method from the nature of data may indeed present improved mechanisms for comparative studies, knowledge sharing and real progress rather than re-invention. It can be argued that evaluations on both real and synthetic data should be equally valued.

Example: Generic methods for intrinsic metrics include the discovery of certain keys, functional dependencies and other meta-data from real data, e.g., Gene Ontology [27], and/or from synthetic data, e.g., fd-reduced-30 [38].

4.2 IT: Intrinsic and Tailored

Even when intrinsic characteristics of data are generalizable across many data types, they often need to be refined in a type-specific way to get the most out of the data [47]. Tailored methods for intrinsic metrics are positioned within Quadrant 2 of Figure 1. For example, although the notion of near-duplicates is meaningful across many data types (strings, images, videos), the specifics of the data type would dictate what is considered a near-duplicate; while an edit-distance based distance measure is meaningful for strings and short text [20], allowing for differences in resolution is important for videos. Care should be taken to avoid unnecessary proliferation of metrics that are data type specific, although sometimes they may be unavoidable or even desirable.

While many generic and tailored methods can be applied on synthetic data, such data sets are often generated for specific purposes, and may not share all the characteristics of real data, so only a subset of the methods may be meaningful for a specific synthetic data set. Indeed, use of synthetic data for any purposes other than evaluation needs to be carefully considered. The specificity of the synthetic data also raises the need for a clear separation between the generative models, data quality methods, and persons responsible for the injection and the subsequent detection of errors.

Example: Tailored methods for intrinsic metrics include the method of [13] for anomaly detection that was tailored for temporal streams and applied to real NYSE data. It is worth noting that conducting robust scalability evaluations of such methods will need carefully crafted synthetic data with tunable parameters.

4.3 EG: Extrinsic and Generic

Data acquires value only when it is successfully used, whether by applications or by humans. Hence it is essential to provide extrinsic metrics that quantify the impact of poor data quality on the tasks that make use of the data, such as modeling, analysis and interpretation as depicted in Quadrant 3 of Figure 1. Quantification of how well a data quality method succeeds in delivering data of value is an important externally focused metric, but is, in general, difficult to study for real data due to its contextual distinctions and complexity. Externally focused metrics, such as putting a monetary cost on the process of data cleaning that makes the data "fit for use" for the given tasks, can play a unifying role towards making it easier to understand and measure the impact of poor data quality.

Synthetic data, in combination with generic methods, presents a controlled and simplified setting through which both internally and externally focused metrics can be studied. Further synthetic data can facilitate comparative studies in terms of the quality of the output from the method as well as its performance on larger scale [48]. It can also assist in the development of community agreed benchmarks for extrinsic metrics.

For example, data curation tools [35] can assist in automating the curation process and reducing the human effort cost (an extrinsic metric). However, quantifying the cost of automating curation against the reduction of human effort for real data can carry an un-manageable complexity [34], and thus may need robust evaluations based on synthetic data.

Example: A method for evaluating the quality of a data exchange system against required user-effort using synthetic data with community agreed parameters such as schema and relation size [6].

4.4 ET: Extrinsic and Tailored

Since the value of data is in its successful use by external tasks, one might argue that it is the task that should drive the metrics [18]. Thus, task-specific, externally focused metrics demand evaluation experiments to be based on real data. This would naturally lead to tailored methods that depend on the type of data and its use. Tailored methods for extrinsic metrics are positioned within Quadrant 4 of Figure 1.

For example, the needs of a network engineer may be significantly different from that of a market analyst; the same data set may meet the needs of one but not the other, necessitating tailored methods and user involvement. A new wave of methods and tools are emerging that endorse human-in-the-loop thinking. The ability to maintain provenance in such iterative

and interactive data curation methods thus becomes particularly important [19], as it improves both the explainability as well as refinement of the method. The resulting Assess-Clean-Evaluate cycle [21] has been adopted by current commercial data cleaning and curation platforms such as [49, 50].

Although real data are necessary to study task specific, externally focused metrics, there is little evidence of large scale sharing of real scenarios (applications, metrics and data) due to the proprietary nature and privacy concerns. The specificity of the real scenarios can also be a prohibiting factor in engaging researchers due to the infeasibility of investing significant effort to evaluate just one use-case [40]. Community approved synthetic but realistic data, meta-data and quality metrics can help overcome this problem and facilitate the development of publicly available benchmarks (like the open source iBench [5, 31]). However, the transparency of the data and meta-data generators is imperative to ensure the integrity and repeatability of the evaluation processes.

Example: Exploratory methods to guide data cleaning in spatio-temporal urban data, e.g., NYC taxi data, towards meeting analytical needs of end users [18].

5. WHAT NEXT?

We have presented two dimensions of empiricism that provide a lens to study data quality research, namely type of metrics, and scope of method, along with the data continuum based on the nature of data.

The dimensions and their inter-relationships provide a means of positioning data quality research, and help to expose limitations, gaps and opportunities. We assert that the classification serves both academics and practitioners in evaluating their contributions.

Academics and researchers can use this classification to reflect on the role of empiricism in their research, and identify gaps in their work towards a more comprehensive and impactful research agenda. Especially researchers who have focused on intrinsic metrics might consider expanding their evaluation to extrinsic metrics in order to study the impact on business or a respective application area. Similarly, practitioners can use the classification to identify opportunities to steer data quality practices into new directions and/or to achieve robust outcomes. In particular, practitioners, who primarily work with real data towards development of tailored solutions, may consider the role and benefits of carefully designed, community accepted synthetic data to convey the broader applicability of their data quality methods.

Based on our investigations, we propose a call to action to promote empiricism in data quality research. Below we outline three immediate and specific actions that can and should be taken:

Share. Enable empirical research by sharing data, metadata, code, application scenarios, and benchmarks. Such sharing is not absent, but can be further promoted by recognizing contributions of data products and benchmarks as high value. We note the recent addition in the PVLDB experiments and analysis paper track [52], and encourage other publication venues to also consider ways to recognize similar contributions from the research community.

Guide. Synthetic data can facilitate rigorous and reproducible evaluations. However, it is necessary to ensure the transparency of results drawn from synthetic or heavily curated real data. The data quality research community needs to develop guidelines for experimental design that stipulate clear separation between error creation and measurement. Such guidelines are well accepted in other disciplines where the stipulations on experimental design are widely accepted, e.g., [26].

Expand. In spite of several decades of data quality research and a large number of outstanding contributions, data quality remains one of the biggest challenges in data management, and has been further exaggerated in the age of big data. A shift towards embracing the spectrum of positions outlined above is needed, which presents a step change from current approaches that tend to be focused on specific extreme positions. Thus, the continuum from real to synthetic data is necessary for robust evaluations; both intrinsic and extrinsic metrics are needed to fully capture a data quality problem; and both generic and tailored methods are needed to balance purpose and applicability.

We invite the community to use, challenge, and refine the classification presented in this paper, and work with us to further promote empiricism in data quality research.

6. ACKNOWLEDGMENTS

We would like to acknowledge all the co-authors of the papers published in [45]. This work is partially supported by the Australian Government through the ARC-DP project DP140103171 (Sadiq, Zhou and Srivastava), and the NSF award OAC-1640864 (Freire).

7. REFERENCES

[1] Abedjan, Z., Chu, X., Deng, D., Fernandez, R.C., Ilyas, I.F., Ouzzani, M., Papotti, P., Stonebraker, M., and Tang, N. 2016. Detecting Data Errors: Where are we and what needs to be done? *Proceedings of the VLDB Endowment*. 9, 12, 993-1004.

[2] Abedjan, Z. and Naumann, F. 2013. Improving RDF data through association rule mining. *Datenbank-Spektrum*. 13, 2, 111-120.

[3] Abiteboul, S., Dong, L., Etzioni, O., Srivastava, D., Weikum, G., Stoyanovich, J., and Suchanek, F. 2015. The elephant in the room: getting value from Big Data. In *Proceedings of the 18th International Workshop on Web and Databases* (2015), ACM, 1-5.

[4] Ananthakrishna, R., Chaudhuri, S., and Ganti, V. 2002. Eliminating fuzzy duplicates in data warehouses. In *Proceedings of the 28th International Conference on Very Large Data Bases* (2002), VLDB Endowment, 586-597.

[5] Arocena, P., Glavic, B., Ciucanu, R., and Miller, R.J. 2015. The iBench integration metadata generator. *Proceedings of the VLDB Endowment*. 9, 3, 108-119.

[6] Arocena, P., Glavic, B., Mecca, G., Miller, R.J., Papotti, P., and Santoro, D. 2016. Benchmarking Data Curation Systems. *IEEE Data Eng. Bull.* 39, 2, 47-62.

[7] Berti-Equille, L., Dasu, T., and Srivastava, D. 2011. Discovery of complex glitch patterns: A novel approach to quantitative data cleaning. In *27th International Conference on Data Engineering (ICDE)* (2011), IEEE, 733-744.

[8] Berti-Equille, L., Loh, J.M., and Dasu, T. 2015. A Masking Index for Quantifying Hidden Glitches. *Knowledge and Information Systems*. 44, 2, 253-277.

[9] Cao, P., Gowda, B., Lakshmi, S., Narasimhadevara, C., Nguyen, P., Poelman, J., Poess, M., and Rabl, T. 2016. From BigBench to TPCx-BB: Standardization of a Big Data Benchmark. In *Technology Conference on Performance Evaluation and Benchmarking* (2016), Springer, 24-44.

[10] Chalamalla, A., Ilyas, I.F., Ouzzani, M., and Papotti, P. 2014. Descriptive and prescriptive data cleaning. In *Proceedings of the International Conference on Management of Data (ACM SIGMOD 2014)* (2014), ACM, 445-456.

[11] Christen, P. 2005. Probabilistic data generation for deduplication and data linkage. In *International Conference on Intelligent Data Engineering and Automated Learning* (2005), Springer, 109-116.

[12] Chu, X., Ilyas, I.F., and Papotti, P. 2013. Holistic data cleaning: Putting violations into context. In *Proceedings of the IEEE International Conference on Data Engineering, ICDE 2013* (Australia2013), IEEE, 458-469.

[13] Dasu, T., Duan, R., and Srivastava, D. 2016. Data Quality for Temporal Streams. *IEEE Data Eng. Bull.* 39, 2, 78-92.

[14] Dasu, T. and Johnson, T. 2003. *Exploratory data mining and data cleaning*. John Wiley & Sons.

[15] Dong, X.L., Gabrilovich, E., Murphy, K., Dang, V., Horn, W., Lugaresi, C., Sun, S., and Zhang, W. 2016. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *IEEE Data Eng. Bull.* 39, 2, 106-117.

[16] Elmagarmid, A.K., Ipeirotis, P.G., and Verykios, V.S. 2007. Duplicate record detection: A survey. *IEEE Transactions on knowledge and data engineering*. 19, 1, 1-16.

[17] Floridi, L. and Illari, P. 2014. *The philosophy of information quality*. Springer.

[18] Freire, J., Bessa, A., Chirigati, F., Vo, H.T., and Zhao, K. 2016. Exploring What not to Clean in Urban Data: A Study Using New York City Taxi Trips. *IEEE Data Eng. Bull.* 39, 2, 63-77.

[19] Freire, J., Glavic, B., Kennedy, O., and Mueller, H. 2016. The exception that improves the rule. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics* (2016).

[20] Hua, W., Zheng, K., and Zhou, X. 2016. Quality-Aware Entity-Level Semantic Representations for Short Texts. *IEEE Data Eng. Bull.* 39, 2, 93-105.

[21] Ilyas, I.F. 2016. Effective Data Cleaning with Continuous Evaluation. *IEEE Data Eng. Bull.* 39, 2, 38-46.

[22] Ilyas, I.F. and Chu, X. 2015. Trends in cleaning relational data: Consistency and deduplication. *Foundations and Trends® in Databases*. 5, 4, 281-393.

[23] Jagadish, H., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., and Shahabi, C. 2014. Big data and its technical challenges. *Communications of the ACM*. 57, 7, 86-94.

[24] Jayawardene, V., Sadiq, S., and Indulska, M. 2013. The curse of dimensionality in data quality. In *24th Australasian Conference on Information Systems (ACIS)* (2013), RMIT University, 1-11.

[25] Juran, J.M. 1989. *Juran on leadership for quality*. New York: The Free Press.

[26] Kirk, R.E. 2003. *Experimental Design*.

[27] Köhler, H., Leck, U., Link, S., and Zhou, X. 2016. Possible and certain keys for SQL. *The VLDB Journal*. 25, 4, 571-596.

[28] Köhler, H., Link, S., and Zhou, X. 2016. Discovering Meaningful Certain Keys from Incomplete and Inconsistent Relations. *IEEE Data Eng. Bull.* 39, 2, 21-37.

[29] Krishnan, S., Wang, J., Wu, E., Franklin, M.J., and Goldberg, K. 2016. Activeclean: Interactive data cleaning for statistical modeling. *Proceedings of the VLDB Endowment*. 9, 12, 948-959.

[30] Kruse, S., Papenbrock, T., Harmouch, H., and Naumann, F. 2016. Data Anamnesis: Admitting Raw Data into an Organization. *IEEE Data Eng. Bull.* 39, 2, 8-20.

[31] The iBench Project: http://dblab.cs.toronto.edu/project/iBench/

[32] Liu, J., Huang, Z., Cai, H., Shen, H.T., Ngo, C.W., and Wang, W. 2013. Near-duplicate video retrieval: Current research and future trends. *ACM Computing Surveys (CSUR)*. 45, 4, 44.

[33] Markie, P. 2004. *Rationalism vs. empiricism*.

[34] Mecca, G., Papotti, P., and Santoro, D. 2014. IQ-METER-An evaluation tool for data-transformation systems. In *30th International Conference on Data Engineering (ICDE)* (2014), IEEE, 1218-1221.

[35] Miller, R.J. 2014. Big Data Curation. In *COMAD* (2014), 4.

[36] Naumann, F. CORA Dataset: https://hpi.de/naumann/projects/repeatability/datasets/cora-dataset.html

[37] Naumann, F. and Herschel, M. 2010. An introduction to duplicate detection. *Synthesis Lectures on Data Management*. 2, 1, 1-87.

[38] Papenbrock, T., Ehrlich, J., Marten, J., Neubert, T., Rudolph, J.-P., Schönberg, M., Zwiener, J., and

Naumann, F. 2015. Functional dependency discovery: An experimental evaluation of seven algorithms. *Proceedings of the VLDB Endowment*. 8, 10, 1082-1093.

[39] Poess, M., Rabl, T., Jacobsen, H.-A., and Caufield, B. 2014. TPC-DI: the first industry benchmark for data integration. *Proceedings of the VLDB Endowment*. 7, 13, 1367-1378.

[40] Popivanov, I. and Miller, R.J. 2002. Similarity search over time-series data using wavelets. In *18th International Conference on Data Engineering (ICDE)* (2002), IEEE, 212-221.

[41] Razniewski, S., Korn, F., Nutt, W., and Srivastava, D. 2015. Identifying the extent of completeness of query answers over partially complete databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2015), ACM, 561-576.

[42] Razniewski, S., Sadiq, S., and Zhou, X. 2016. Exploiting Hierarchies for Efficient Detection of Completeness in Stream Data. In *Australasian Database Conference* (2016), Springer, 419-431.

[43] Rekatsinas, T., Chu, X., Ilyas, I.F., and Ré, C. 2017. HoloClean: Holistic Data Repairs with Probabilistic Inference. *Proceedings of the VLDB Endowment*. 10, 11, 1190-1201.

[44] Sadiq, S. and Papotti, P. 2016. Big data quality-whose problem is it? In *32nd International Conference on Data Engineering (ICDE)* (2016), IEEE, 1446-1447.

[45] Sadiq, S. and Srivastava, D. 2016. Special Issue on Data Quality *Bulletin of the Technical Committee on Data Engineering*. 39 2.

[46] Sadiq, S., Yeganeh, N.K., and Indulska, M. 2011. 20 years of data quality research: themes, trends and synergies. In *Proceedings of the Twenty-Second Australasian Database Conference-Volume 115* (2011), Australian Computer Society, Inc., 153-162.

[47] Santoro, D., Arocena, P., Glavic, B., Mecca, G., Miller, R.J., and Papotti, P. 2016. BART in Action: Error Generation and Empirical Evaluations of Data-Cleaning Systems. In *Proceedings of the 2016 International Conference on Management of Data* (2016), ACM, 2161-2164.

[48] Soliman, M.A., Ilyas, I.F., and Chang, K. 2007. Top-k query processing in uncertain databases. In *23rd International Conference on Data Engineering (ICDE)* (2007), IEEE, 896-905.

[49] Tamr: https://www.tamr.com/

[50] Trifacta: https://www.trifacta.com/

[51] Ukkonen, E. 1992. Approximate string-matching with q-grams and maximal matches. *Theoretical computer science*. 92, 1, 191-211.

[52] VLDB. 44th International Conference on Very Large Data Bases 2018: http://vldb2018.lncc.br/call-for-research-track.html

[53] Wang, J., Kraska, T., Franklin, M.J., and Feng, J. 2012. Crowder: Crowdsourcing entity resolution. *Proceedings of the VLDB Endowment*. 5, 11, 1483-1494.

[54] Wang, R.Y. and Strong, D.M. 1996. Beyond accuracy: What data quality means to data consumers. *Journal of management information systems*. 12, 4, 5-33.

Report from the Fourth Workshop on Algorithms and Systems for MapReduce and Beyond (BeyondMR'17)

Foto N. Afrati
National Technical University of Athens, Greece
afrati@softlab.ntua.gr

Jan Hidders
Vrije Universiteit Brussel, Belgium
jan.hidders@vub.be

Paraschos Koutris
University of Wisconsin-Madison, USA
paris@cs.wisc.edu

Jacek Sroka
University of Warsaw, Poland
j.sroka@mimuw.edu.pl

Jeffrey Ullman
Stanford University, USA
ullman@cs.stanford.edu

ABSTRACT

This report summarizes the presentations and discussions of the fourth workshop on Algorithms and Systems for MapReduce and Beyond (BeyondMR'17). The BeyondMR workshop was held in conjunction with the 2017 SIGMOD/PODS conference in Chicago, Illinois, USA on Friday May 19, 2017. The goal of the workshop was to bring together researchers and practitioners to explore algorithms, computational models, languages and interfaces for systems that provide large-scale parallelization and fault tolerance. These include specialized programming and data-management systems based on MapReduce and extensions thereof, graph processing systems and data-intensive workflow systems.

The program featured two well-attended invited talks, the first on current and future development in big data processing by Matei Zaharia from Databricks and the University of Stanford, and the second on computational models for the analysis and development of big data processing algorithms by Ke Yi from the Hong Kong University of Science and Technology.

1. INTRODUCTION

In this edition of BeyondMR four main themes emerged: (*i*) *languages and interfaces*, which investigates new interfaces and languages for specifying data processing workflows to increase usability, which includes both high-level and declarative languages, as well as more low-level workflow evaluation plans, (*ii*) *algorithms*, which involves the design, evaluation and analysis of algorithms for large-scale data processing and (*iii*) *integration*, which concerns the integration of different types of analytics frameworks, e.g., for database-oriented analytics and for linear algebra.

Both keynotes covered the aforementioned themes. The keynote by Matei Zaharia addressed *usability*, *integration* and *hardware*, by describing recent and future developments of the *Spark* framework, and the keynote by Ke Yi gave an overview of computational models for large-scale parallel *algorithms*.

The presented papers addressed the themes as follows. Paper [13] covered the themes *languages and interfaces* and *algorithms* by presenting an approach where spreadsheets are used as the programming interface for a large-scale data-processing framework with special algorithms for executing typical spreadsheet data-manipulation operations. Paper [7] tackles the themes *languages and interfaces* and *integration* by presenting an integrated algebra for implementing and optimizing data-processing workflows with both relational algebra and linear algebra operators. In paper [10] the themes *languages and interfaces* and *algorithms* are both addressed by presenting distributed algorithms and implementations for computing the closure of certain OWL ontologies. The same themes also return in paper [12] which discusses distributed algorithms and implementations for computing navigation queries over RDF graphs formulated in a high-level query language. The theme *algorithms* was again covered in paper [2] which discussed benchmarking dataflow systems for the purpose of machine learning, by emphasizing scalability for high-dimensional but sparse data. In paper [8] the themes *languages and interfaces*, *algorithms* and *integration* were addressed by presenting a framework translating high-level specifications of data processing pipelines to different evaluation plans, making different choices concerning parallelization frameworks, acceleration

frameworks and data-processing platforms. Furthermore, the theme *algorithms* was covered by paper [4] which discusses algorithms for matrix multiplication on MapReduce-like data processing platforms. The theme was also addressed by paper [6], presenting streaming algorithms for multi-way theta-joins, and by paper [3], presenting distributed algorithms for binning in big genomic datasets.

We will give a summary of the two keynotes in Section 2 followed by a description of the contributions of the presented papers further in Section 3.

2. SUMMARY OF KEYNOTES

We give a summary of the two keynotes, which contributed to visibility of the workshop.

What's Changing in Big Data?

The first keynote was presented by Matei Zaharia, Stanford University, USA, and Databricks. His presentation discussed the main changes to big-data processing in the last 10 years, as he has experienced as codeveloper of Spark and chief technologist of Databricks. It focused on three developments, namely (1) the user base moving from software engineers to data analysts, (2) the changing hardware and the computational bottleneck moving from I/O to the computation and (3) the delivery of big-data processing services through cloud computing.

The discussion of the changing user-base started with the observation that usability had been, and remains, a crucial factor in the success of systems such as Spark. The changing user base shows in the selection of programming languages for Spark, where a shift is seen from *Scala* and *Java* to *Python* and *R*. Three areas where there are currently usability problems are (a) the understandability of the API, (b) the complexity of performance predictability, and (c) the difficulty in accessing the system from smaller front-end tools such as Tableau and Excel. These problems are addressed by Spark projects such as *DataFrames* and *Spark SQL*. The first explicitly deals with structured data as it appears in *Python* and *R*, and the second introduces a high-level API for SQL-like processing with database-like query optimization. Next to that, there are also projects such as *ML Pipelines* for machine-learning pipelines, *graphFrames* for graph processing, and the introduction of streaming over DataFrames.

The changes in hardware in the last 10 years were summarized by the observations that although the size of storage and the speed of networks has increased by an order of magnitude, the speed of CPUs has not. Moreover, *GPU*s and *FPGA*s have become more ubiquitous and powerful, and so need to be better utilized. These concerns are addressed in the projects *Tungsten* [16] and *Weld* [11]. The first tries to make better use of the hardware by run-time code generation that exploits cache locality and uses off-heap memory management. The second allows the easy integration of different analytics libraries by offering a common algebra for relational algebra operations, linear algebra operations, and graph operations.

The final change discussed was that big-data processing services are increasingly provided through cloud computing. This provides new challenges as well as opportunities. New challenges are scaling up state management and stress testing, but opportunities are created by fast roll-out of new functionality and consequently rapid comprehensive feedback.

At the end of the presentation research challenges were discussed. These included new types of interfaces for new types of users, new optimization techniques to deal with new types and configurations of hardware, the integration of different processing platforms and the support of interaction between users of such platforms.

Sequential vs Parallel, Fine-grained vs Coarse-grained, Models and Algorithms

Ke Yi, Hong Kong University of Science and Technology, China, was the second keynote speaker. His talk focused on computational models for approximating the running time of distributed algorithms. He warned at the start, by quoting Gorge Box, *All models are wrong, but some are useful.* After this, the talk started with a quick introduction to *RAM* and *PRAM models*, emphasizing the *Concurrent Read / Exclusive Read* (CREW) and *Concurrent Write/ Exclusive Write* (EREW) variants. He also presented the *External Memory Model* (EM) where internal memory is limited to size M and the data is transferred between internal and external memory in blocks of size B. This part of the presentation was concluded by discussing the known relationships between those classes and with their classification according to two independent criteria, namely if the model is fine-grained or coarse-grained and if it is parallel or sequential.

Next, the *Parallel External-Memory* (PEM) model for multicore GPUs [1] was presented together followed by the *Bulk-Synchronous Parallel* (BSP) model. The latter is a shared-nothing, coarse-grained parallel model [15] that is well suited for models used in practice like MapReduce and Pregel. Simplifications and variants were presented and the models were compared. Also, methods to simulate one model by the other were discussed, as well as the re-

lations between BSP, PRAM and EM. It was concluded that it is unlikely that optimal BSP algorithms are produced by simulating the fine-grained PRAM or the sequential EM model.

The presentation concluded with a discussion of how to design algorithms for those models. *Work-depth models* and *Brent's theorem* were recalled. Finally, the multithreaded and external memory versions of work-depth models were presented, and several problems were discussed for join algorithms, such as the nested-loop and hypercube algorithms.

3. REVIEW OF PRESENTED WORK

(short paper) Towards minimal algorithms for big data analytics with spreadsheets

The paper [13] presents research done by Jacek Sroka, Artur Leśniewski, Mirosław Kowaluk, Krzysztof Stencel and Jerzy Tyszkiewicz, from the Institute of Informatics at the University of Warsaw. The contribution of the paper is an algorithm to answer bulk range queries. The idea is based on a range tree, but adapted to work on datasets that are too big for processing on a single machine. The algorithm is organized so that the computation is perfectly balanced between the nodes in the cluster. It also allows answering queries in bulk, i.e., a large number of queries at the same time.

The motivation for the algorithm is to significantly lower the technological barrier, which currently prevents average users from analysing big datasets available as CSV files. The authors propose creation of a tool, which translates spreadsheets of regular structure into semantically equivalent MapReduce or Spark computations. Such a tool would enable users to prepare spreadsheets on a small sample of the data and then automatically translate them into MapReduce or Spark to be executed on the full dataset. As it is estimated that spreadsheet programmers outnumber the programmers of all other languages together, achieving that goal would allow a huge number of new users to benefit from big-data processing.

(full paper) LaraDB: A Minimalist Kernel for Linear and Relational Algebra Computation

The paper [7] was produced by Dylan Hutchison, Bill Howe and Dan Suciu, from the Department of Computer Science and Engineering at the University of Washington. It presents *Lara*, an algebra consisting of three operators that expresses *Relational Algebra* (RA) and *Linear Algebra* (LA), as well as optimization rules for a certain programming model. A proof is provided that the algebra is more explicit then MapReduce, but also more general then RA and LA. The practicality and efficiency of the algebra is demonstrated by implementing its operators using range scans over partitioned, sorted maps, a primitive that is available in a wide range of back-end engines. Concretely, the operators were implemented as range iterators in *Apache Accumulo*, an implementation of Google's *BigTable*. This implementation is shown to outperform the *Accumulo*'s native MapReduce integration for mixed-abstraction workflows involving joins and aggregations in the form of matrix multiplication.

(full paper) SPOWL: Spark-based OWL 2 Reasoning Materialisation

The paper [10] was the result of work by Yu Liu and Peter Mcbrien, from the Department of Computing at Imperial College London. It presents *SPOWL*, which uses *Spark* to implement OWL inferencing over large ontologies. This system compiles T-Box axioms to *Spark* programs, which are iteratively executed to compute and materialize the closure of the ontology. The result can, for example, be used to compute SPARQL queries. The system leverages the advantages of *Spark* by caching results in distributed memory and parallelizing jobs in a more flexible manner. It also optimises the number of necessary iterations by analysing the dependencies in the T-Box hierarchy and compiling the axioms correspondingly. The performance of *SPOWL* is evaluated against *WepPIE* on *LUBM* datasets, and is shown to be generally comparable or better.

(full paper) Querying Semantic Knowledge Bases with SQL-on-Hadoop

The paper [12] describes research done by Martin Przyjaciel-Zablocki, Alexander Schätzle and Georg Lausen, in the Databases and Information Systems group, at the Institut für Informatik, in the Albert-Ludwigs-Universität Freiburg. The paper presents a new processor for *Trial-QL*, an SQL-like query language for RDF that is based on the *Triple Algebra with Recursion* (TriAL*) [9] and therefore especially suited to express navigational queries. The presented *Trial-QL* processor is built on top of *Impala* (a massive parallel SQL Engine on *Hadoop*) and *Spark*, using one unified data storage system based on *Parquet* and *HDFS*. The paper studies and compares different evaluation algorithms and storage strategies, and compares the performance to other RDF management systems such as *Sempala*, *Neo4j*, *PigSPARQL*, *H2RDF+* and *SHARD*. Interactive and competitive response times are shown on datasets of more than a billion triples, and even

results with more than 25 billion triples could be produced in minutes.

(full paper) Benchmarking Data Flow Systems for Scalable Machine Learning

The paper [2] presents the contribution by Christoph Boden, Andrea Spina, Tilmann Rabl and Volker Markl, in the Database Systems and Information Management Group at the TU Berlin. The paper investigates the scalability of data-flow systems such as *Apache Flink* and *Apache Spark* for typical machine-learning workflows. The workloads are assumed to differ from conventional workloads as are found in existing benchmarks, which are often based on tasks such as *WordCount*, *Grep* or *Sort*. The main differences are that the data tends to be more sparse but at the same high dimensional, so the scalability in the number of dimensions becomes more important. The paper therefore presents a representative set of distributed machine-learning algorithms suitable for large-scale distributed settings that have close resemblance to industry-relevant applications and provide generalizable insights into system performance. These algorithms were implemented in *Apache Spark* and *Apache Flink*, and after tuning, the two systems were compared in scalability in both the size of the data and the dimensionality of the data. Measurements were done with datasets containing up to 4 billion data points and 100 million dimensions. The main conclusion is that current systems are surprisingly inefficient in dealing with such high-dimensional data.

(full paper) A containerized analytics framework for data- and compute-intensive pipeline applications

The paper [8] describes the result obtained by Yuriy Kaniovskyi, Martin Köhler and Siegfried Benkner, in the Research Group Scientific Computing at the University of Vienna and at the School of Computer Science in the University of Manchester. It presents a framework for executing high-level specifications of data-processing pipelines on heterogeneous architectures using a variety of programming paradigms. The presented framework allows several ways of executing each step in the pipeline, using different parallellization libraries (such as *OpenMP*, *TBB*, *MPI* and *Cilk*), different acceleration frameworks (such as *OpenACC*, *CUDA* and *OpenCL*) or different data-processing platforms (e.g., *MapReduce*, *Spark*, *Storm* and *Flink*). Moreover, the intermediate results might be passed in different ways, e.g., by storing it in *HDFS*, in a local file system, in memory, or by streaming it.

The approach starts with a high-level language for describing the global data-processing pipeline. It also allows the separate specification of the different implementation variants for each step in the pipeline, together with their computational requirements. These are self-contained execution units that can encapsulate legacy, native or accelerator-based code. This enables the framework to fuse data-intensive programming paradigms (via native *YARN* applications) with computation-intensive programming paradigms (via containerization). The framework then optimizes the executing plan by taking into account the specified *performance models* and the available resources as specified in a specially developed *generic resource description framework*.

(full paper) MapReduce Implementation of Strassen's Algorithm for Matrix Multiplication

The paper [4] presents the work by Minhao Deng and Prakash Ramanan, in the Electrical Engineering and Computer Science department at Whichita State University. It studies MapReduce implementations of Strassen's algorithm for multiplying two $n \times n$ matrices, introduced by Volker Strassen in [14]. The Strassen algorithm has time complexity $\Theta(n^{\log_2(7)})$, which is an improvement over the standard algorithm since the latter has time complexity $\Theta(n^3)$ and $\log_2(7) \approx 2.81$.

When considering implementations on MapReduce of the straightforward algorithm, it has been shown that this can be done in typically one or two passes. The paper investigates in how many passes the Strassen algorithm can be implemented. This algorithm consists of three parts: *Part I* where sums and differences of quadrants of the input matrices are computed, *Part II* where several products are computed of the matrices from Part I and *Part III* where each quadrant of the final matrix is computed by taking sums and differences of the matrices from Part II.

Direct implementation of these parts in MapReduce is shown to be possible in $\log_2(n)$, 1 and $\log_2(n)$ passes, respectively. Moreover, it is shown that *Part I* can be implemented in 2 passes, with total work $\Theta(n \log_2(7))$, and reducer size and reducer workload $o(n)$. It is conjectured that *Part III* can also be implemented in a constant number of passes and with small reducer size and workload.

(short paper) Scaling Out Continuous Multi-Way Theta Joins

The paper [6] presents work by Manuel Hoffmann and Sebastian Michel, performed at the Databases and Information Systems Group at the department

of computer science at the University of Kaiserslautern. The paper presents generic tuple-routing schemes for computing distributed multiway theta-joins over streaming data. These schemes are implemented in an architecture where query plans consist of logical operators to *Apache Storm* topologies. The paper reports the first results for the *TPC-H* benchmark where the topologies are executed on *Amazon EC2* instances.

(short paper) Bi-Dimensional Binning for Big Genomic Datasets

The paper [3] describes results obtained by Pietro Pinoli, Simone Cattani, Stefano Ceri and Abdulrahman Kaitoua, in the Department of Electronics, Information, and Bioengineering at the Politecnico di Milano. The main result is a bi-dimensional binning strategy for the *SciDB* array database, motivated by implementation of the *GenoMetric Query Language* (GMQL) — a high-level query language for genomics — that the authors previously developed. Due to the array database particularities, it is not possible to dynamically split a region and distribute its replicas to an arbitrary number of adjacent cells. Yet in order to apply a binning strategy, all the regions need to be replicated an identical number of times. The authors propose to organize the bins in a two dimensional grid. In such a grid the values are located above the diagonal and it is possible to define spaces that would contain values for one bin without replicating the values. Such bins (spaces) can share values because they share some grid cells.

4. CONCLUSION

The presentations and keynotes at BeyondMR'17 provided an overview of current developments and emerging issues in the area of algorithms, computational models, architectures and interfaces for systems that provide large-scale parallelization. The workshop attracted 17 submissions from which the program committee led by Paris Koutris from the University of Wisconsin-Madison accepted 6 full papers and 3 short papers.

Keynotes and papers covered topics such as user interfaces, integration of programming paradigms, algorithmics and leveraging new hardware features. The contributions suggest that while MapReduce has been extended and replaced by newer models, there is an active area of research centred around data-management systems based on MapReduce and extensions, providing ever more insight for developing more effective graph processing systems and data-intensive workflow systems.

The sessions were well attended with an average of 35 participants, and the proceedings are published in the ACM Digital Library [5].

Acknowledgements: We would like to thank PC members, keynote speakers, authors, local workshop organizers and attendees for making BeyondMR 2017 a success. We also express our appreciation for the support from Google Inc. Finally we would like to thank the anonymous reviewers for their constructive remarks on how to improve this report.

5. REFERENCES

[1] Lars Arge, Michael T. Goodrich, Michael Nelson, and Nodari Sitchinava. Fundamental parallel algorithms for private-cache chip multiprocessors. In *Proc. of SPAA '08*, pages 197–206, New York, NY, USA, 2008. ACM.

[2] Christoph Boden, Andrea Spina, Tilmann Rabl, and Volker Markl. Benchmarking data flow systems for scalable machine learning. In Hidders [5], pages 5:1–5:10.

[3] Simone Cattani, Stefano Ceri, Abdulrahman Kaitoua, and Pietro Pinoli. Bi-dimensional binning for big genomic datasets. In Hidders [5], pages 9:1–9:4.

[4] Minhao Deng and Prakash Ramanan. Mapreduce implementation of Strassen's algorithm for matrix multiplication. In Hidders [5], pages 7:1–7:10.

[5] Jan Hidders, editor. *Proc. of BeyondMR'17*, New York, NY, USA, 2017. ACM.

[6] Manuel Hoffmann and Sebastian Michel. Scaling out continuous multi-way theta-joins. In Hidders [5], pages 8:1–8:4.

[7] Dylan Hutchison, Bill Howe, and Dan Suciu. LaraDB: A minimalist kernel for linear and relational algebra computation. In Hidders [5], pages 2:1–2:10.

[8] Yuriy Kaniovskyi, Martin Koehler, and Siegfried Benkner. A containerized analytics framework for data and compute-intensive pipeline applications. In Hidders [5], pages 6:1–6:10.

[9] Leonid Libkin, Juan Reutter, and Domagoj Vrgoč. Trial for RDF: Adapting graph query languages for RDF data. In *Proc. of PODS '13*, pages 201–212, New York, NY, USA, 2013. ACM.

[10] Yu Liu and Peter McBrien. SPOWL: Spark-based OWL 2 reasoning materialisation. In Hidders [5], pages 3:1–3:10.

[11] Shoumik Palkar, James J. Thomas, Anil Shanbhag, Malte Schwarzkopt, Saman P. Amarasinghe, and Matei Zaharia. Weld: A common runtime for high performance data analytics. In *Proc. of CIDR 2017*. www.cidrdb.org, January 2017.

[12] Martin Przyjaciel-Zablocki, Alexander Schätzle, and Georg Lausen. Querying semantic knowledge bases with SQL-on-Hadoop. In Hidders [5], pages 4:1–4:10.

[13] Jacek Sroka, Artur Leśniewski, Mirosław Kowaluk, Krzysztof Stencel, and Jerzy Tyszkiewicz. Towards minimal algorithms for big data analytics with spreadsheets. In Hidders [5], pages 1:1–1:4.

[14] Volker Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13(4):354–356, August 1969.

[15] Leslie G. Valiant. A bridging model for parallel computation. *Commun. ACM*, 33(8):103–111, August 1990.

[16] Reynold Xin and Josh Rosen. Project Tungsten: Bringing Apache Spark closer to bare metal. `https://databricks.com/blog/2015/04/28/project-tungsten-bringing-spark-closer-to-bare-metal.html`, April 2015.

Commonsense Knowledge in Machine Intelligence

Niket Tandon
Allen Institute for Artificial Intelligence
Seattle, WA
nikett@allenai.org

Aparna S. Varde
Department of CS
Montclair State University
Montclair, NJ
vardea@montclair.edu

Gerard de Melo
Department of CS
Rutgers University
Piscataway, NJ
gdm@demelo.org

ABSTRACT

There is growing conviction that the future of computing depends on our ability to exploit big data on the Web to enhance intelligent systems. This includes encyclopedic knowledge for factual details, common sense for human-like reasoning and natural language generation for smarter communication. With recent chatbots conceivably at the verge of passing the Turing Test, there are calls for more common sense oriented alternatives, e.g., the Winograd Schema Challenge. The Aristo QA system demonstrates the lack of common sense in current systems in answering fourth-grade science exam questions. On the language generation front, despite the progress in deep learning, current models are easily confused by subtle distinctions that may require linguistic common sense, e.g.*quick food* vs. *fast food*. These issues bear on tasks such as machine translation and should be addressed using common sense acquired from text. Mining common sense from massive amounts of data and applying it in intelligent systems, in several respects, appears to be the next frontier in computing. Our brief overview of the state of Commonsense Knowledge (CSK) in Machine Intelligence provides insights into CSK acquisition, CSK in natural language, applications of CSK and discussion of open issues. This paper provides a report of a tutorial at a recent conference with a brief survey of topics.

1. INTRODUCTION

Commonsense knowledge (CSK) is inherent in human cognition and behavior, yet is often too subtle for machines to acquire and use. It differs from encyclopedic knowledge, which is more factual and explicit. Clearly, modern intelligent systems can far surpass humans with respect to encyclopedic knowledge such as knowledge of named entities. For example, if we query the name of a sufficiently prominent person on a modern search engine, it will return specific details about this person, including their date and place of birth, occupation, education, significant achievements, awards, controversies, and so forth. A regular human being would find it hard to memorize such trivia about millions of people. Still, intelligent machines lag behind humans in performing simple tasks such as distinguishing between a truck and an overpass, as observed in an incident with a semi-automated automobile in 2016. The Tesla vehicle confused the truck with an overpass due to the truck's height, leading to the loss of a human life. While human drivers may suffer from fatigue and other challenges, a responsible human driver is easily able to draw on common sense to differentiate between the two. Thus, it is important to endow machines with commonsense knowledge.

Our recent tutorial on this topic has centered on precisely this challenge. It has been presented at the ACM Conference on Information and Knowledge Management (CIKM) in Singapore in November 2017.

We start from text often found in sources such as the Web. We survey the literature on extracting commonsense knowledge from text and using the acquired knowledge to provide textual outputs useful in intelligent machines. Hence we go from text to knowledge and knowledge to text. A related issue is common sense in natural language processing. For instance, a machine translation engine should not emit *quick food* as the translation when the input was in fact referring to *fast food*. We survey such high-potential areas: CSK mining methods; CSK for smarter natural language processing; and applications towards smart computing.

The remainder of this paper is laid out as follows. Section 2 provides an overview of commonsense knowledge bases and CSK acquisition. Section 3 focuses on CSK for natural language processing. In Section 4, we discuss CSK applications and open issues in various domains, including smart cities.

2. CSK ACQUISITION

2.1 Introduction to Common Sense

Commonsense knowledge differs from encyclopedic knowledge in that it deals with general knowledge rather than the details of specific entities. Most regular knowl-

edge bases (KBs) contribute millions of facts about entities such as people or geopolitical entities, but fail to provide fundamental knowledge such as the notion that a toddler is likely too young to have a doctoral degree in physics. The challenges in acquiring CSK include its elusiveness and context-dependence. Common sense is elusive because it is scarcely and often only implicitly expressed, it is affected by reporting bias [13], and it may require considering multiple modalities. Context plays an important role for common sense in defining its correctness, and this must be accounted for while acquiring it. We partition common sense into three dimensions [21],

(i) Common sense of objects in the environment, including properties, theories (such as physics), and associated emotions;

(ii) Common sense of object relationships, including taxonomic, spatial and structural relationships among the objects;

(iii) Common sense of object interactions, including actions, processes, and procedural knowledge.

Well-known projects in the commonsense KB space include hand-crafted resources such as WordNet [9] and Cyc [16], ConceptNet [14], WebChild [22], and visual KBs such as Visual Genome [8].

2.2 CSK Representation

To represent and reason over such commonsense knowledge, there are a wide range of representations that we partition across two dimensions,

(i) **Discrete or continuous**: Discrete representation of common sense in the form of structured frames, micro-theories [16, 20] and unstructured natural language representation [1, 14] have been very popular. Recently, continuous representation based on factorization and other deep learning methods [2, 26] learns representations from large amounts of Web data [12] and generalizes better than discrete representation.

(ii) **Multimodal**: Embedding based representation that account for textual and visual knowledge [10] can combine words and images in the same space and enable similarity computations as well as analogical reasoning.

Note that some assumptions typical in continuous representations for encyclopedic KBs may not hold for commonsense KBs. For instance, typical methods of generating negative training data for continuous representation learning do not apply equally well to CSK.

2.3 Acquiring CSK

The next level is to obtain more advanced commonsense facts, both from text and from video and other multimodal Web data. We characterize CSK acquisition across the following dimensions:

(i) **Level of supervision:** High quality manual, text-based commonsense KBs (CKBs) include Cyc and WordNet. Such KBs have been used extensively in various applications due to their high accuracy, but they remain costly to create and extend. Common sense acquisition through crowd-sourcing has been a well-motivated technique because commonsense games are easy for humans. ConceptNet is among the well-known crowd-sourced acquisitions, while Verbosity [24] uses visual data to drive the game experience. The main challenge facing these approaches is user engagement, because humans do not find much challenge in simple common sense based games. Automated systems have attempted to mine big data on the Web to overcome the limitations of manual and semi-automated systems. However, noisy data is a central challenge here and thus, robustness is an important dimension in the machinery. WebChild [22] is a semantically refined commonsense knowledge base mined from Web-scale textual data.

(ii) **Modality:** NEIL [3] generates a small-scale commonsense knowledge base exclusively from visual orientation and visual features from images, by starting with seed images for a phrase and refining the senses and classifiers by clustering images discovered for the phrase. In addition to individual facts, we can also mine entailments for commonsense reasoning [1]. This aspect has also been touched upon in the tutorial.

2.4 Evaluating CSK

Being less factual in nature, evaluation of CSK is a formidable challenge. Fortunately, many different techniques have evolved to address this challenge, which can be partitioned across two dimensions.

(i) **Intrinsic or extrinsic evaluation**: We argue that an intrinsic evaluation is most practical when judging "what usually holds" as opposed to "what can hold". While measuring recall is typically not feasible, recent efforts have designed some proxies towards this direction [7]. More recently, intrinsic evaluation of commonsense knowledge has been automated by visual verification and detecting inconsistencies. Extrinsic evaluation indirectly measures the correctness through performance gains on an external tasks [22].

(ii) **Manual or automated evaluation**: A number of disparate large-scale annotated challenge sets exist for measuring the impact of commonsense knowledge. These challenge datasets are either text based or visuals based, and are inference easy or hard. This includes Winograd stories [15], Aristo QA [4], and VQA [25].

Finally, we consider physical and social common sense as interesting future directions. Multimodal mining to acquire commonsense knowledge is a scalable method that overcomes the limitations of the elusiveness of CSK and visual verification and jointly leveraging disparate information sources can help overcome reporting bias.

Salient and concise KBs are helpful for quality control in KBs. In this regard, the evaluation of commonsense knowledge needs to be standardized with extrinsic datasets, to continuously track progress.

3. CSK FOR NATURAL LANGUAGE

As an example for the use of CSK in natural language, we consider the task of detecting and avoiding inappropriate collocations. A correctly collocated expression is one that a native speaker of a language such as English would typically use in good communication, e.g., *strong tea* or *red tape*. Conversely, erroneous or odd collocations include expressions that are not typically used in correct communication, e.g., *mighty tea* or *powerful tea* (instead of *strong tea*), and *crimson tape* or *scarlet tape* (instead of *red tape*). These are referred to in the literature as *collocation errors*.

Incorrectly collocated expressions can be encountered when a literal translation is conducted from a source to a target expression. If an expression does not get adequately translated, this can adversely affect communication in intelligent systems. It is thus important to fix odd collocations based on common sense. For instance, consider the expression *powerful tea* entered as a Web query. One finds that search engine results for this query contain the words *powerful* or *tea* or both. However, the user probably means to search for the availability of *strong tea*, which could further be used in an online shopping context. Upon entering the correct collocation *strong tea*, it is observed that we obtain significantly better results, including appropriate images and websites.

Linguistic classification of collocation errors: Previous work [11] has proposed a method of identifying collocation errors using association measures over syntactic patterns via a frequency based approach. CSK is captured through the writings of native speakers in KBs that serve as sources of ground truth evidence of correct collocations. Further research [6] has suggested a method of using the native or source language, i.e., the L1 language to classify collocation errors. They use annotated texts written by second language learners, incorporating corrections by professional English instructors. These serve as their sources for CSK with correctly collocated expressions. Such works address CSK-based collocations mainly from a linguistic classification perspective. As an added plus, they tangentially point towards corrective measures.

Detection and correction: Different types of collocations are addressed by Park et al. [18]. They categorize collocation errors into insertion, deletion, substitution, and transposition types. For example, substitution errors occur when a non-preferred word is used in place of a more commonly used word, e.g., *pure sky* instead of *clear sky*. Transposition errors occur when words are used in an order different from the intended meaning, e.g., *make friendships close* instead of *make close friendships*. They develop a tool called AwkChecker to detect and correct such errors in documents.

CollOrder [23] outputs ordered responses to odd collocations by relying on semantic similarity, ranking techniques and ensemble learning. This entails error detection with POS tagging and search for matches (odd collocations) followed by error correction by searching for precise collocations, ranking, filtering and frequency ordering. Large repositories such as the British National Corpus serve as sources of CSK in the form of correct collocations. Classical rule induction [5] in the context of ensemble learning is found useful to learn similarity measures for collocation error detection and correction.

Broader impacts: Incorporating CSK into natural language processing helps us develop smarter systems in machine intelligence by providing better responses and better machine translation. Open research issues such as the challenge of sparse data (as opposed to frequent data) and literary allusion are relevant to the enhancement of CSK-based approaches such as collocation error correction. Sparsity is a challenge because many approaches in the literature rely on the frequency of expressions to assess their appropriateness. As these challenges are addressed, CSK-based natural language processing will improve and second language learners and more generally users of intelligent systems will benefit.

4. FURTHER APPLICATIONS

Apart from natural language generation, further applications using CSK include sentiment analysis, set expansion and computer vision. There are challenges in reasoning with CSK that have possible solutions and present some open issues.

Many use cases for CSK will stem from the evolution towards smart cities, e.g., [17] consider smart environment, smart mobility, smart government, smart people, smart economy and smart living as key ingredients.

For each characteristic, there are important current and future applications of CSK. For example, deployment of CSK in autonomous vehicles helps them make more well-informed decisions and hence drive better [19], thus avoiding potential accidents, e.g., Tesla crashing into a truck by confusing it with an overpass. This affects the smart mobility characteristic. Developments in CSK-based natural language processing have the potential to benefit 21st century education, thus enhancing the smart people characteristic. There are significant open issues calling for further research. For instance, the use of CSK can enhance systems such as canal lights in Amsterdam that brighten and dim based on pedestrian usage [17], to promote a cleaner environment. This calls for further research on the specifics of harnessing CSK

from given repositories to improve such systems, so as to enhance the smart environment characteristic.

5. CONCLUSIONS

We have briefly surveyed CSK acquisition and the use of repositories such as WebChild, CSK in natural language for addressing collocation issues based on linguistic classification as well as detecting and correcting collocation errors, and CSK applications in domains including smart cities. We emphasize that commonsense knowledge has made people smarter, is making machines smarter and will make smart cities smarter.

The tutorial we presented at ACM CIKM on these topics was particularly well-received. The slides for this tutorial can be found at:
`http://allenai.org/tutorials/csk`.

6. REFERENCES

[1] Gabor Angeli and Christopher D. Manning. Naturalli: Natural logic inference for common sense reasoning. In *EMNLP*, 2014.

[2] Antoine Bordes and Evgeniy Gabrilovich. Constructing and Mining Web-scale Knowledge Graphs. In *KDD Tutorials*, 2014.

[3] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Neil: Extracting visual knowledge from web data. In *ICCV*, 2013.

[4] Peter Clark, Niranjan Balasubramanian, Sumithra Bhakthavatsalam, Kevin Humphreys, Jesse Kinkead, Ashish Sabharwal, and Oyvind Tafjord. Automatic construction of inference-supporting knowledge bases. In *AKBC '14*, 2014.

[5] William Cohen. Fast effective rule induction. In *ICML*, 1995.

[6] Daniel Dahlmeier and Hwee Ton Ng. Correcting semantic collocation errors with l1-induced paraphrases. In *EMNLP*, 2011.

[7] Bhavana Dalvi, Niket Tandon, and Peter Clark. Domain-targeted, high precision knowledge extraction. *TACL*, 2017.

[8] Ranjay Krishna et. al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2016.

[9] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[10] Andrea Frome, Gregory S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.

[11] Yoko Futagi, Paul Deane, Martin Chodorow, and Joel Tetreault. A computational approach to detecting collocation errors in the writing of non-native speakers of english. *Computer Assisted Language Learning*, 2008.

[12] Yoshua Bengio Geoffrey Hinton. Cifar ncap - summer school 2014.

[13] Jonathan Gordon and Benjamin Van Durme. Reporting bias and knowledge acquisition. In *AKBC*, 2013.

[14] Hugo Liu and Pushpak Singh. ConceptNet: a practical commonsense reasoning toolkit. *BT Technology Journal*, 2004.

[15] Quan Liu, Hui Jiang, Zhen-Hua Ling, Xiao-Dan Zhu, Si Wei, and Yu Hu. Combing context and commonsense knowledge through neural networks for solving winograd schema problems. *CoRR*, 2016.

[16] C. Matuszek, M. Witbrock, R.C. Kahlert, J. Cabral, D. Schneider, P. Shah, and D. Lenat. Searching for common sense: Populating Cyc from the Web. In *AAAI*, 2005.

[17] Technical University of Vienna. European smart cities: Technical report, August 2015.

[18] Taehyun Park, Edward Lank, Pascal Poupart, and Michael Terry. Is the sky pure today - awkchecker: An assistive tool for detecting and correcting collocation errors. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 2008.

[19] Priya Persaud, Aparna Varde, and Stefan Robila. Enhancing autonomous vehicles with commonsense: Smart mobility in smart cities. In *IEEE ICTAI workshop on Smart Cities*, 2017.

[20] R. Schank and R. Abelson. *Scripts, plans, goals and understanding: An inquiry into human knowledge structures*. Lawrence Erlbaum Associates, Hillsdale, NJ., 1977.

[21] Niket Tandon. Commonsense knowledge acquisition and applications. In *Doctoral dissertation*, 2016.

[22] Niket Tandon, Gerard de Melo, Fabian Suchanek, and Gerhard Weikum. Webchild: Harvesting and organizing commonsense knowledge from the web. In *WSDM*, 2014.

[23] Alan Varghese, Aparna Varde, Jing Peng, and Eileen Fitzpatrick. A framework for collocation error correction in web pages and text documents. *ACM SIGKDD Explorations*, 2015.

[24] Luis von Ahn, Mihir Kedia, and Manuel Blum. Verbosity: a game for collecting common-sense facts. In *CHI*, 2006.

[25] Peng Wang, Qi Wu, Chunhua Shen, Anton van den Hengel, and Anthony R. Dick. Fvqa: Fact-based visual question answering. *TPAMI '17*.

[26] Bishan Yang and Tom Mitchell. Leveraging Knowledge Bases in LSTMs for Improving Machine Reading. In *ACL*, 2017.