SIGMOD Officers, Committees, and Awardees

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

SIGMOD Executive Committee:
Donald Kossmann (Chair), Anastasia Ailamaki (Vice-Chair), Magdalena Balazinska, K. Selçuk Candan, Yanlei Diao, Curtis Dyreson, Yannis Ioannidis, Christian Jensen, and Jan Van den Bussche.

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

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

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

SIGMOD Record Editor-in-Chief:
Yanlei Diao, University of Massachusetts Amherst <yanlei AT cs.umass.edu>

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, Nesime Tatbul, 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), Juliana Freire (co-Chair), Kian-Lee Tan, Andy Pavlo, Aditya Parameswaran, Ioana Manolescu, Lucian Popa, Chris Jermaine, Renée Miller

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

SIGMOD Edgar F. Codd Innovations Award

For innovative and highly significant contributions of enduring value to the development, understanding, or use of database systems and databases. Formerly known as the "SIGMOD Innovations Award", it now honors Dr. E. F. (Ted) Codd (1923 - 2003) who invented the relational data model and was responsible for the significant development of the database field as a scientific discipline. Recipients of the award are the following:

Michael Stonebraker (1992)
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)

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 and Lawrence Rowe (2015)

Martin Kersten (2016)

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)

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

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

[Last updated : December 30, 2016]

Editor's Notes

Welcome to the December 2016 issue of the ACM SIGMOD Record!

This issue opens with the 2016 Dagstuhl Report that outlines some of the most important research directions in the area of Principles of Data Management (PDM). In April 2016, a group of researchers in the PDM area joined in a workshop at the Dagstuhl Castle in Germany, which was organized jointly by the Executive Committee of the ACM Symposium on Principles of Database Systems (PODS) and the Council of the International Conference on Database Theory (ICDT). As summarized in this report, the workshop identified research challenges for PDM around seven core themes, namely, *Managing Data at Scale*, *Multi-model Data*, *Uncertain Information*, *Knowledge-enriched Data*, *Data Management and Machine Learning*, *Process and Data*, and *Ethics and Data Management*. In addition, the workshop highlighted two accelerating trends: the increasing embrace of neighboring disciplines, including especially Machine Learning, Statistics, Probability, and Verification; and the increased focus on obtaining positive results to enable the use of mathematically-based insights in practical settings. These trends are expected to continue in the years to come.

The Database Principles column continues with an article by Koutris and Suciu on a formal analysis of multiway join processing in massively parallel systems. The article introduces a theoretical model, called the MPC (Massively Parallel Computation) model, which characterizes the number of synchronization points and the maximum load per machine as observed in popular Hadoop and Spark systems. Using the MPC model the article shows the design of novel algorithms for multiway join and theoretical results that prove their optimality through tight lower bounds.

The Vision column features two articles. The first article, by Kamat and Nandi, studies variance implementations in real-world database systems given their increased importance in big data analytics. The article reports that some major database systems use a representation that suffers from floating point precision loss, then reviews literature on variance calculation in both the statistics and database communities, and finally gives recommendations on implementing variance functions in various query processing settings. The second article, by Vartak et al., studies visualization recommendation systems. With the advent of large, high-dimensional datasets and significant interest in data science, there is a growing need for tools that can support rapid visual analysis. This article presents the vision of a new class of visualization systems that can automatically identify and interactively recommend visualizations relevant to an analytical task. The article describes the key requirements for such a visualization recommendation system as well as the challenges in realizing this vision, and finally presents several approaches to address the challenges.

The Distinguished Profiles column features Rick Hull, a distinguished researcher, and Stratos Idreos, a recent recipient of the SIGMOD Jim Gray Dissertation Award. Rick Hull is currently a researcher at IBM. Before joining IBM, he was a professor at the University of Southern California for many years and managed a research group at Bell Labs. Rick is an ACM Fellow and a coauthor of the classic database theory book, "Foundations of Databases." In this interview, Rick talks about his work experiences in different institutes and highlights a few research topics such as artifact-centric business process models and cognitive computing. The second interview features Stratos Idreos, the 2011 recipient of the SIGMOD Jim Gray Dissertation Award. Stratos' thesis addresses Database Cracking, that is, auto-tuning of database kernels. Stratos is currently an Assistant Professor at the Harvard University.

The Reports column features a report on the First International Workshop on Reproducible Open Science (RepScience2016), which was organized in conjunction with the 20th edition of the International Conference on Theory and Practice of Digital Libraries. The goal of the workshop was to provide a forum for constructively exploring foundational, organizational, and systemic challenges towards the implementation of Open Science publishing principles. The workshop brought together skills and experiences in the form of invited talks and paper presentations, focusing on the definition and establishment of the next generation scientific communication ecosystem.

On behalf of the SIGMOD Record Editorial board, I hope that you enjoy reading the December 2016 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 2016

Past SIGMOD Record Editors:

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

Research Directions for Principles of Data Management (Abridged)

Serge Abiteboul Marcelo Arenas Pablo Barceló Meghyn Bienvenu
Diego Calvanese Claire David Richard Hull Eyke Hüllermeier
Benny Kimelfeld Leonid Libkin Wim Martens Tova Milo
Filip Murlak Frank Neven Magdalena Ortiz Thomas Schwentick
Julia Stoyanovich Jianwen Su Dan Suciu
Victor Vianu Ke Yi

In April 2016, a community of researchers working in the area of Principles of Data Management (PDM) joined in a workshop at the Dagstuhl Castle in Germany. The workshop was organized jointly by the Executive Committee of the ACM Symposium on Principles of Database Systems (PODS) and the Council of the International Conference on Database Theory (ICDT). The mission of the workshop was to identify and explore some of the most important research directions that have high relevance to society and to Computer Science today, and where the PDM community has the potential to make significant contributions. This article presents a summary of the report created by the workshop [4]. That report describes the family of research directions that the workshop focused on from three perspectives: potential practical relevance, results already obtained, and research questions that appear surmountable in the short and medium term. The report organizes the identified research challenges for PDM around seven core themes, namely *Managing Data at Scale*, *Multi-model Data*, *Uncertain Information*, *Knowledge-enriched Data*, *Data Management and Machine Learning*, *Process and Data*, and *Ethics and Data Management*. Since new challenges in PDM arise all the time, we note that this list of themes is not intended to be exclusive.

The Dagstuhl report is intended for a diverse audience, ranging from funding agencies, to universities and industrial research labs, to researchers and scientists who are exploring the many issues that arise in modern data management. The report is also intended for policy makers, sociologists, and philosophers, because it re-iterates the importance of considering ethics in many aspects of data creation, access, and usage, and suggests how research can help to find new ways for maximizing the benefits of massive data while nevertheless safeguarding the privacy and integrity of citizens and societies.

The field of PDM is broad. It has ranged from the development of formal frameworks for understanding and managing data and knowledge (including data models, query languages, ontologies, and transaction models) to data structures and algorithms (including query optimizations, data exchange mechanisms, and privacy-preserving manipulations). Data management is at the heart of most IT applications today, and will be a driving force in personal life, social life, industry, and research for the foreseeable future. We anticipate on-going expansion of PDM research as the technology and applications involving data management continue to grow and evolve.

PDM played a foundational role in the relational database model, with the robust connection between algebraic and calculus-based query languages, the connection between integrity constraints and database design, key insights for the field of query optimization, and the fundamentals of consistent concurrent transactions. This early work included rich cross-fertilization between PDM and other disciplines in mathematics and computer science, including logic, complexity theory, and knowledge representation. Since the 1990s we have seen an overwhelming increase in both the production of data and the ability to store and access such data. This has led to a phenomenal metamorphosis in the ways that we manage and use data. During this time, we have gone (1) from stand-alone disk-based databases to data that is spread across and linked by the Web, (2) from rigidly structured towards loosely structured data, and (3) from relational data to many different data models (hierarchical, graph-structured, data points, NoSQL, text data, image data, etc.). Research on PDM has developed during that time, too, following, accompanying and influencing this process. It has intensified research on extensions of the relational model (data exchange, incomplete data, probabilistic data, ...), on other

data models (hierachical, semi-structured, graph, text, ...), and on a variety of further data management areas, including knowledge representation and the semantic web, data privacy and security, and data-aware (business) processes. Along the way, the PDM community expanded its cross-fertilization with related areas, to include automata theory, web services, parallel computation, document processing, data structures, scientific workflow, business process management, data-centered dynamic systems, data mining, machine learning, information extraction, etc.

Looking forward, three broad themes in data management stand out where principled, mathematical thinking can bring new approaches and much-needed clarity. The first relates to the overall lifecycle of so-called "Big Data Analytics", that is, the application of statistical and machine learning techniques to make sense out of, and derive value from, massive volumes of data. As documented in numerous sources, so-called "data wrangling" can form 50% to 80% of the labor costs in an analytics investigation. As discussed in the Dagstuhl report, the PDM research areas of *Managing Data at Scale*, *Knowledge-enriched Data*, *Multi-model Data*, *Uncertain Information*, and *Data Management and Machine Learning* are all relevant to supporting Big Data Analytics. The second broad theme of data management where principled thinking can help stems from new forms of data creation and processing, especially as it arises in applications such as web-based commerce, social media applications, and data-aware workflow and business process management. The PDM research areas of *Multi-model Data*, *Knowledge-enriched Data*, *Uncertain Information*, and *Process and Data* are all relevant to this theme. These are providing approaches that make it easier to understand and process the myriad kinds of data and updates involved, and to enable higher degrees of confidence in transactional software that is used to process the data. The third broad theme, which is just beginning to emerge, is the development of new principles and approaches in support of ethical data management. Emerging research suggests that the use of mathematical principles in research on *Ethics and Data Management* can lead to new approaches to ensure data privacy for individuals, a broader perspective on notions of "fair" data dissemination and analysis, and compliance with government and societal regulations at the corporate level.

The findings of the Dagstuhl report differ from, and complement, the findings of the 2016 Beckman Report [1] in two main aspects. Both reports stress the importance of "Big Data" as the single largest driving force in data management usage and research in the current era. The current report focuses primarily on research challenges where a mathematically based perspective has had and will continue to have substantial impact. This includes for example new algorithms for large-scale parallelized query processing and Machine Learning, and models and languages for heterogeneous and uncertain information. The current report also considers additional areas where research into the principles of data management can make growing contributions in the coming years, including for example approaches for combining data structured according to different models, process taken together with data, and ethics in data management.

The remainder of this article includes overviews of the seven PDM research areas mentioned above, and a concluding section with comments about the road ahead for PDM research. The interested reader is referred to the full Dagstuhl Report [4] for more detail and references.

1. MANAGING DATA AT SCALE

Volume is still the most prominent feature of *Big Data*. The PDM community, as well as the general theoretical computer science community, has made significant contributions to efficient data processing at scale. Still, however, we face important practical challenges such as the following:

New Paradigms for Multi-Way Join Processing.
A celebrated result by Atserias, Grohe, and Marx [17] has sparked a flurry of research efforts in reexamining how multi-way joins should be computed. In all current relational database systems, a multi-way join is processed in a pairwise framework using a binary tree (plan), which is chosen by the query optimizer. However, the recent theoretical studies have discovered that for many queries and data instances, even the best binary plan is suboptimal by a large polynomial factor. Several worst-case algorithms have been designed in different computation models [70, 51, 20, 6], all of which have abandoned the binary tree paradigm, while adopting a more *holistic* approach. In particular, *leapfrog join* [87] has been implemented inside a full-fledged database system. We believe that these newly developed algorithms have a potential to change how multi-way join processing is currently done in database systems. Of course, this can only be achieved with significant efforts for designing and implementing new query optimizers and cost estimation under the new paradigm.

Approximate Query Processing.

The area of *online aggregation* [49] studies new algorithms that allow to return approximate results (with statistical guarantees) for analytical queries at early stages of the processing, so that the user can terminate it as soon as the accuracy is acceptable. Recent studies have shown encouraging results [48, 61], but there is still much room for improvement: (1) The existing algorithms have only used simple random sampling or sample random walks to sample from query results. More sophisticated techniques based on Markov Chain Monte Carlo might be more effective. (2) The streaming algorithms community has developed many techniques to summarize large data sets into compact data structures while preserving properties of the data. These summarization techniques can also be useful in approximate query processing. (3) Integrating these techniques into data processing engines is still a significant challenge.

These practical challenges raise the following theoretical challenges:

The Relationship Among Various Big Data Computation Models.

The theoretical computer science community has developed many beautiful models of computation aimed at handling data sets that are too large for the traditional random access machine (RAM) model, the most prominent ones including parallel RAM (PRAM), external memory (EM) model, streaming model, the BSP model and its recent refinements to model modern distributed architectures. Several studies seem to suggest that there are deep connections between seemingly unrelated Big Data computation models for streaming computation, parallel processing, and external memory, especially for the class of problems interesting to the PDM community (e.g., relational algebra) [80, 45, 58]. Investigating this relationship would reveal the inherent nature of these problems with respect to scalable computation, and would also allow us to leverage the rich set of ideas and tools that the theory community has developed over the decades.

The Communication Complexity of Parallel Query Processing.

New large-scale data analytics systems use massive parallelism to support complex queries on datasets. These systems use clusters of servers and proceed in multiple communication rounds. In these systems, the communication cost is usually the bottleneck, and therefore has become the main measure of complexity for algorithms designed for these models. Recent studies (e.g., [20]) have established tight bounds on the communication cost for computing join queries, but many questions remain open: (1) The existing bounds are tight only for one-round algorithms. However, new large-scale systems like Spark have greatly improved the efficiency of multi-round iterative computation, thus the one-round limit seems unnecessary. The communication complexity of multi-round computation remains largely open. (2) The existing work has only focused on a small set of queries (full conjunctive queries), while many other types of queries remain unaddressed. Broadly, there is great interest in large-scale machine learning using these systems, thus it is important to study the communication complexity of classical machine learning tasks under these models. This is developed in more detail in Section 5, which summarizes research opportunites at the crossroads of data management and machine learning.

We think that the following techniques will be useful in handling these challenges: statistics, sampling and approximation theory, communication complexity, information theory, and convex optimization.

2. MULTI-MODEL DATA: AN OPEN ECOSYSTEM OF DATA MODELS

Over the past 20 years, the landscape of available data has dramatically changed. While the huge amount of available data is perceived as a clear asset, exploiting this data meets the challenges of the "4 V's" mentioned in the Introduction.

One particular aspect of the *variety* of data is the existence and coexistence of different models for semi-structured and unstructured data, in addition to the widely used relational data model. Examples include tree-structured data (XML, JSON), graph data (RDF, property graphs, networks), tabular data (CSV), temporal and spatial data, text, and multimedia. We can expect that in the near future, new data models will arise in order to cover particular needs. Importantly, data models include not only a data structuring paradigm, but also approaches for queries, updates, integrity constraints, views, integration, and transformation, among others.

Following the success of the relational data model, originating from the close interaction between theory and practice, the PDM community has been working for many years towards understanding each one of the aforementioned models formally. Classical DB topics—schema and query languages, query evaluation and optimization, incremental processing of evolving data, dealing with inconsistency and incompleteness, data integration and exchange, etc.—have been revisited. This line of work has been successful

from both the theoretical and practical points of view. As these questions are not yet fully answered for the existing data models and will be asked again whenever new models arise, it will continue to offer practically relevant theoretical challenges. But what we view as a new grand challenge is the coexistence and interconnection of all these models, complicated further by the need to be prepared to embrace new models at any time.

The coexistence of different data models resembles the fundamental problem of data heterogeneity within the relational model, which arises when semantically related data is organized under different schemas. This problem has been tackled by data integration and data exchange, but since these classical solutions have been proposed, the nature of available data has changed dramatically, making the questions open again. This is particularly evident in the Web scenario, where not only the data comes in huge amounts, in different formats, is distributed, and changes constantly, but also it comes with very little information about its structure and almost no control of the sources. Thus, while the existence and coexistence of various data models is not new, the recent changes in the nature of available data raise a strong need for a new principled approach for dealing with different data models: an approach flexible enough to allow keeping the data in their original format (and be open for new formats), while still providing a convenient unique interface to handle data from different sources. It faces the following four specific practical challenges.

Modelling Data.
How does one turn raw data into a database? Could we create methodologies allowing engineers to design a new data model?

Understanding Data.
How does one make sense of the data? Could we help the user and systems to understand the data without first discovering its structure in full?

Accessing Data.
How does one extract information? How can we help users formulate queries in a more uniform way?

Processing Data.
How does one evaluate queries efficiently?

These practical challenges raise concrete theoretical problems, some of which go beyond the traditional scope of PDM. Within PDM, the key theoretical challenges are the following.

Schema Languages.
Design flexible and robust multi-model schema languages. Multi-model schema languages should offer a uniform treatment of different models, the ability to describe mappings between models (implementing different views on the same data, in the spirit of data integration), and the flexibility to seamlessly incorporate new models as they emerge.

Schema Extraction.
Provide efficient algorithms to extract schemas from the data, or at least discover partial structural information (cf. [23, 26]). The long-standing challenge of entity resolution is exacerbated in the context of finding correspondences between data sets structured according to different models [85].

Visualization of Data and Metadata.
Develop user-friendly paradigms for presenting the metadata information and statistical properties of the data in a way that helps in formulating queries. This requires understanding and defining what the relevant information in a given context is, and representing it in a way allowing efficient updates as the context changes (cf. [30, 15]).

Query Languages.
Go beyond bespoke query languages for the specific data models [13] and design a query language suitable for multi-model data, either incorporating the specialized query languages as sub-languages or offering a uniform approach to querying, possibly at the cost of reduced expressive power or higher complexity.

Evaluation and Optimization.
Provide efficient algorithms for computing meaningful answers to a query, based on structural information about data, both inter-model and intra-model; this can be tackled either directly [56, 46] or via static optimization [21, 33].

All these problems require strong tools from PDM and theoretical computer science in general (complexity, logic, automata, etc.). But solving them will also involve knowledge and techniques from neighboring communities. For example, the second, third and fifth challenges naturally involve data mining and machine learning aspects (see Section 5). The first, second, and third raise knowledge representation issues (see Section 4). The first and fourth will require expertise in programming languages. The fifth is at the interface between PDM and algorithms, but also between PDM and systems. The third raises human-computer interaction issues.

3. UNCERTAIN INFORMATION

Incomplete, uncertain, and inconsistent information is ubiquitous in data management applications. This was recognized already in the 1970s [32], and since then the significance of the issues related to incompleteness and uncertainty has been steadily growing: it is a fact of life that data we need to handle on an everyday basis is rarely complete. However, while the data management field developed techniques specifically for handling incomplete data, their current state leaves much to be desired, both theoretically and practically. Even after 40+ years of relational technology, when evaluating SQL queries over incomplete databases one gets results that make people say *"you can never trust the answers you get from [an incomplete] database"* [34]. In fact we know that SQL can produce every type of error imaginable when nulls are present [63].

On the theory side, we appear to have a good understanding of what is needed in order to produce correct results: computing *certain answers* to queries. These are answers that are true in all complete databases that are compatible with the given incomplete database. This idea, that dates back to the late 1970s as well, has become *the* way of providing query answers in all applications, from classical databases with incomplete information [53] to new applications such as data integration and exchange [59, 14], consistent query answering [22], ontology-based data access [29], and others. The reason these ideas have found limited application in mainstream database systems is their complexity. Typically, answering queries over incomplete databases with certainty can be done efficiently for conjunctive queries or some closely related classes, but beyond the complexity quickly grows to intractable – and sometimes even undecidable, see [62]. Since this cannot be tolerated by real life systems, they resort to ad hoc solutions, which go for efficiency and sacrifice correctness; thus bizarre and unexpected behavior occurs.

While even basic problems related to incompleteness in relational databases remain unsolved, we now constantly deal with more varied types of incomplete and inconsistent data. A prominent example is that of probabilistic databases [81], where the confidence in a query answer is the total weight of the worlds that support the answer. Just like certain answers, computing exact answer probabilities is usually intractable, and yet it has been the focus of theoretical research.

The key challenge in addressing the problem of handling incomplete and uncertain data is to provide theoretical solutions that are *usable in practice*. Instead of proving more impossibility results, the field should urgently address what can actually be done efficiently.

Making theoretical results applicable in practice is the biggest practical challenge for incomplete and uncertain data. To move away from the focus on intractability and to produce results of practical relevance, the PDM community needs to address several challenges.

RDBMS Technology in the Presence of Incomplete Data.

It must be capable of finding query answers one can trust, and do so efficiently. But how do we find good quality query answers with correctness guarantees when we have theoretical intractability? For this we need new approximation schemes, quite different from those that have traditionally been used in the database field. Such schemes should provide guarantees that answers can be trusted, and should also be implementable using existing RDBMS technology.

Models of Uncertainty.

What is provided by current practical solutions is rather limited. Looking at relational databases, we know that they try to model everything with primitive null values, but this is clearly insufficient. We need to understand types of uncertainty that need to be modeled and introduce appropriate representation mechanisms.

Benchmarks for Uncertain Data.

What should we use as benchmarks when working with incomplete/uncertain data? Quite amazingly, this has not been addressed; in fact standard benchmarks tend to just ignore incomplete data, making it hard to test efficiency of solutions in practice.

Handling Inconsistent Data.

How do we make handling inconsistency (in particular, consistent query answering) work in practice? How do we use it in data cleaning? Again, there are many strong theoretical results here, but they concentrate primarily on tractability boundaries and various complexity dichotomies for subclasses of conjunctive queries, rather than practicality of query answering techniques. There are promising works on enriching theoretical *repairs* with user preferences [78], or ontologies [44], along the lines of approaches described in Section 4, but much more foundational work needs to be done before they can get to the level of practical tools.

Handling Probabilistic Data.

The common models of probabilistic databases are arguably simpler and more restricted than the models studied by the Statistics and Machine Learning communities. Yet common complex models can be simulated by probabilistic databases if one can support expressive query languages [55]; hence, model complexity can be exchanged for query complexity. Therefore, it is of great importance to develop techniques for approximate query answering, on expressive query languages, over large volumes of data, with practical execution costs.

The theoretical challenges can be split into three groups.

Modeling.

We need to provide a solid theoretic basis for the practical modeling challenge above; this means understanding different types of uncertainty and their representations. As with any type of information stored in databases, there are lots of questions for the PDM community to work on, related to data structures, indexing techniques, and so on.

Reasoning.

There is much work on this subject; see Section 4 concerning the need to develop next-generation reasoning tools for data management tasks. When it comes to using such tools with incomplete and uncertain data, the key challenges are: How do we do inference with incomplete data? How do we integrate different types of uncertainty? How do we learn queries on uncertain data? What do query answers actually tell us if we run queries on data that is uncertain? That is, how results can be generalized from a concrete incomplete data set.

Algorithms.

To overcome high complexity, we often need to resort to approximate algorithms, but approximation techniques are different from the standard ones used in databases, as they do not just speed up evaluation but rather ensure correctness. The need for such approximations leads to a host of theoretical challenges. How do we devise such algorithms? How do we express correctness in relational data and beyond? How do we measure the quality of query answers? How do we take user preferences into account?

4. KNOWLEDGE-ENRICHED DATA MANAGEMENT

Over the past two decades we have witnessed a gradual shift from a world where most data used by companies and organizations was regularly structured, neatly organized in relational databases, and treated as complete, to a world where data is heterogenous and distributed, and can no longer be treated as complete. Moreover, not only do we have massive amounts of data; we also have very large amounts of rich knowledge about the application domain of the data, in the form of taxonomies or full-fledged ontologies, and rules about how the data should be interpreted, among other things. Techniques and tools for managing such complex information have been studied extensively in Knowledge Representation, a subarea of Artificial Intelligence. In particular logic-based formalisms, such as description logics and different rule-based languages, have been proposed and associated reasoning mechanisms have been developed. However, work in this area did not put a strong emphasis on the traditional challenges of data management, namely huge volumes of data, and the need to specify and perform complex operations on the data efficiently, including both queries and updates.

Both practical and theoretical challenges arise when rich domain-specific knowledge is combined with large amounts of data and the traditional data management requirements, and the techniques and approaches coming from the PDM community will provide important tools to address them. We discuss first the practical challenges.

Providing End Users with Flexible and Integrated Access to Data.

A key requirement in dealing with complex, distributed, and heterogeneous data is to give end users the ability to directly manage such data. This is a challenge since end users might have deep expertise about a specific domain of interest, but in general are not data management experts. Ontology-based data management has been proposed recently as a general paradigm to address this challenge.

Ensuring Interoperability at the Level of Systems Exchanging Data.

Enriching data with knowledge is not only relevant for providing end-user access, but also enables direct inter-operation between systems, based on the exchange of data and knowledge at the system level. A specific area where this is starting to play an important role is e-commerce, where standard ontologies are already available [50].

Personalized and Context-Aware Data Access and Management.

Information is increasingly individualized and only fragments of the available data and knowledge might

be relevant in specific situations or for specific users. Heterogeneity needs to be dealt with, both with respect to the modeling formalism and with respect to the modeling structures chosen to capture a specific real-world phenomenon.

Bringing Knowledge to Data Analytics and Data Extraction.

Increasing amounts of data are being collected to perform complex analysis and predictions. Currently, such operations are mostly based on data in "raw" form, but there is a huge potential for increasing their effectiveness by enriching and complementing such data with domain knowledge, and leveraging this knowledge during the data analytics and extraction process.

Making the Management User Friendly.

Systems combining large amounts of data with complex knowledge are themselves very complex, and thus difficult to design and maintain. Appropriate tools that support all phases of the life-cycle of such systems need to be designed and developed, based on novel user interfaces for the various components.

To provide adequate solutions to the above practical challenges, several key theoretical challenges need to be addressed, requiring a blend of formal techniques and tools traditionally studied in data management, with those typically adopted in knowledge representation in AI.

Development of Reasoning-Tuned DB Systems.

Such systems will require new/improved database engines optimized for reasoning over large amounts of data and knowledge, able to compute both crisp and approximate answers, and to perform distributed reasoning and query evaluation.

Choosing/Designing the Right Languages.

The languages and formalisms adopted in the various components of knowledge-enriched data management systems have to support different types of knowledge and data, e.g., mixing open and closed world assumption, and allowing for representing temporal, spatial, and other modalities of information [27, 18, 25, 16, 69].

New Measures of Complexity.

To appropriately assess the performance of such systems and be able to distinguish easy cases that seem to work well in practice from difficult ones, alternative complexity measures are required that go beyond the traditional worst-case complexity. These might include suitable forms of average case or parameterized complexity, complexity taking into account data distribution (on the Web), and forms of smoothed analysis.

Next-Generation Reasoning Services.

The kinds of reasoning services that become necessary in the context of knowledge-enriched data management applications go well beyond traditional reasoning studied in knowledge representation, which typically consists of consistency checking, classification, and retrieval of class instances. The forms of reasoning that are required include processing of complex forms of queries in the presence of knowledge, explanation (which can be considered as a generalization of provenance), abductive reasoning, hypothetical reasoning, inconsistency-tolerant reasoning, and defeasible reasoning to deal with exceptions.

Incorporating Temporal and Dynamic Aspects.

A key challenge is represented by the fact that data and knowledge is not static, and changes over time, e.g., due to updates on the data while taking into account knowledge, forms of streaming data, and more in general data manipulated by processes. Dealing with dynamicity and providing forms of inference (e.g., formal verification) in the presence of both data and knowledge is extremely challenging and will require the development of novel techniques and tools [28, 16].

In summary, incorporating domain-specific knowledge to data management is both a great opportunity and a major challenge. It opens up huge possibilities for making data-centric systems more intelligent, flexible, and reliable, but entails computational and technical challenges that need to be overcome. We believe that much can be achieved in the coming years. Indeed, the increasing interaction of the PDM and the Knowledge Representation communities has been very fruitful, particularly by attempting to understand the similarities and differences between the formalisms and techniques used in both areas, and obtaining new results building on mutual insights. Further bridging this gap by the close collaboration of both areas appears as the most promising way of fulfilling the promises of Knowledge-enriched Data Management.

5. DATA MANAGEMENT AND MACHINE LEARNING

We believe that Data Management (DM) and Machine Learning (ML) can mutually benefit from each other. Nowadays, systems that emerge from the

ML community are strong in their capabilities of statistical reasoning, and systems that emerge from the DM community are strong in their support for semantics, maintenance and scale. This complementarity in assets is accompanied by a difference in the core mechanisms: the PDM community has largely adopted methodologies driven by logic, while the ML community centralized around probability and statistics. Yet, modern applications require systems that are strong in *both* aspects, providing a thorough and sophisticated management of data while incorporating its inherent statistical nature.

We envision a plethora of research opportunities in the intersection of PDM and ML. We outline several directions, which we classify into two categories: *DM for ML* and *ML for DM*. The required methodologies and formal foundations span a variety of related fields such as logic, formal languages, computational complexity, statistical analysis, and distributed computing.

Category DM for ML

Key challenges in this area are as follows.

Feature Generation and Engineering.

Feature engineering refers to the challenge of designing and extracting signals to provide to the general-purpose ML algorithm at hand, in order to properly perform the desired operation. This is a critical and time-consuming task [57], and a central theme of modern ML methodologies. Unlike usual ML algorithms that view features as numerical values, the database has access to, and understanding of, the *queries* that transform raw data into these features. Thus, PDM can contribute to feature engineering in various ways, especially on a semantic level, and provide solutions to problems such as the following: How to develop effective languages for query-based feature creation? How to use such languages for designing a set of complementary features optimally suited for the ML task at hand? Is a given language suitable for a certain ML task? Important criteria for the goodness of a feature language include the risks of *under/overfitting* the training data, as well as the computational complexity of evaluation. The PDM community has already studied problems of a similar nature [47].

The promise of deep (neural network) learning brings substantial hope for reducing the effort in manual feature engineering. Is there a general way of solving ML tasks by applying deep learning directly to the database (as has already been done, for example, with *semantic hashing* [74])? Can database queries (of different languages) complement neural networks by means of expressiveness and/or efficiency?

Large-Scale Machine Learning.

Machine learning is nowadays applied to massive data sets of considerable size, including potentially unbounded streams of data. Under such conditions, an effective data management and the use of appropriate data structures that offer the learning algorithm fast access to the data are major prerequisites for realizing model induction and inference in an efficient manner [72]. Research along this direction has amplified in recent years and includes, for example, the use of hashing [88], Bloom filters [31], tree-based data structures [38] in learning algorithms. Related to this is work on distributed machine learning, where data storage and computation is accomplished in a network of distributed units [7], and the support of machine learning by data stream management systems [67].

Complexity Analysis.

The PDM community has established a strong machinery for fine-grained analysis of querying complexity; see, e.g., [10]. Complexity analysis of such granularity is highly desirable for the ML community, especially for analyzing learning algorithms that involve various parameters like I/O dimension, and number of training examples [54]. Results along this direction, connecting DM querying complexity and ML training complexity, have been recently shown [75].

Category ML for DM

Data management systems support a core set of querying operators (e.g., relational algebra, grouping and aggregate functions, recursion) that are considered as the common requirement of applications. We believe that this core set should be revisited, and specifically that it should be extended with common ML operators.

Incorporating ML features is a natural evolution for PDM. Database systems with such features have already been developed [77, 12]. Query languages have traditionally been designed with emphasis on being *declarative*: a query states how the answer should logically relate to the database, not how it is to be computed algorithmically. Incorporating ML introduces a higher level of declarativity, where one states how the end result should behave (via examples), but not necessarily which query is deployed for the task. In that spirit, we propose the following directions for PDM research.

Unified Models.

An important role of the PDM community is in establishing common formalisms and semantics for the database community. It is therefore an important opportunity to establish the "relational algebra" of data management systems with built-in ML/statistics operators.

Lossy Optimization.

From the early days, the focus of the PDM community has been on *lossless* optimization, i.e., optimization that does not modify the final result [76, 89]. As mentioned in Section 1, in some scenarios it makes sense to apply *lossy* optimization that guarantees only an approximation of the answer. Incorporating ML into the query model gives further opportunities for lossy optimization, as training paradigms are typically associated with built-in quality (or "risk") functions. Hence, we may consider reducing the execution cost if it entails a bounded impact on the quality of the end result [9].

Confidence Estimation.

Once statistical and ML components are incorporated in a data management system, it becomes crucial to properly estimate the *confidence* in query answers [77]. It is then an important direction to establish probabilistic models that capture the combined process and allow to estimate probabilities of end results. For example, by applying the notion of the VC-dimension, an important theoretical concept in generalization theory, to database queries, Riondato et al. [73] provide accurate bounds for their selectivity estimates that hold with high probability. This direction can leverage the past decade of research on probabilistic databases [82], which can be combined with theoretical frameworks of machine learning, such as PAC learning [86].

6. PROCESS AND DATA

Many forms of data evolve over time, and most processes access and modify data sets. Industry works with massive volumes of evolving data, primarily in the form of transactional systems and Business Process Management (BPM) systems. Over the past half century, computer science research has studied foundational issues of process and of data mainly as separated phenomena, but research into basic questions about systems that combine process and data has been growing over the past decade. Two key areas where data and process have been studied together are scientific workflows and data-aware BPM [52].

In the 1990's and 00's, foundational research in scientific workflow helped to establish the basic frameworks for supporting these workflows, to enable the systematic recording and use of provenance information, and to support systems for exploration that involve multiple runs of a workflow with varying configurations [36].

Foundational work on data-aware BPM began in the mid-00's [24, 41], enabled in part by IBM's "Business Artifacts" model for business process [71], that combines data and process in a holistic manner. Deutch and Milo [39] provide a survey and comparison of several of the most important early models and results on process and data. One variant of the business artifact model has provided the conceptual basis for the recent OMG Case Management Model and Notation (CMMN) standard [65]. Rich work on verification for data-aware processes has emerged [28, 40], and the artifact-based perspective is enabling an approach to managing the interaction of business processes and legacy data systems [83].

Foundational work in the area of process and data has the potential for continued and expanded impact in the following six practical challenge areas.

Automating Manual Processes.

While many business processes have been automated using techniques from the BPM field, there are many other processes that are still manual – often because high levels of variation make it cost prohibitive to automate using current techniques.

Evolution and Migration of Business Processes.

Managing change of business processes remains largely manual, highly expensive, time consuming, and risk-prone.

Business Process Compliance and Correctness.

Compliance with government regulations and corporate policies is a rapidly growing challenge, e.g., as governments attempt to enforce policies around financial stability and data privacy. Ensuring compliance is largely manual today, and involves understanding how regulations can impact or define portions of business processes, and then verifying that process executions will comply.

Business Process Interaction and Interoperation.

Managing business processes that flow across enterprise boundaries has become increasingly important with globalization of business and the splintering of business activities across numerous companies. The recent industrial interest in shared ledger technologies, e.g., Blockchain, highlights the importance of this area.

Business Process Discovery and Understanding.
The field of Business Intelligence, which provides techniques for mining and analyzing information about business operations, is essential to business success, but is today based on a broad variety of largely *ad hoc* and manual techniques [37].

Workflow and Business Process Usability.
Enabling people to understand and work effectively to manage large numbers of process definitions and process instances remains elusive, especially when considering the interactions between process, data (both newly created and legacy), resources, the workforce, and business partners.

The above practical BPM challenges raise key research challenges that need to be addressed using approaches that include mathematical and algorithmic frameworks and tools.

Verification and Static Analysis.
Because of the infinite state space inherent in data-aware processes [28, 40], verification currently relies on faithful abstractions reducing the problem to classical finite-state model checking. Further work is needed to develop more powerful abstractions, address new application areas, enable incremental verification techniques, and enable modular styles of verification that support "plug and play" approaches.

Tools for Design and Synthesis.
Although compilers and relational database design have both benefited from solid mathematical foundations (context free grammars and dependency theory, respectively), there is still no robust framework that supports principled design of business processes in the larger context of data, resources, and workforce.

Models and Semantics for Views, Interaction, and Interoperation.
A robust theory of views for data-aware business processes has the potential to enable substantial advances in the simplification of process comparison, process composition, process interoperation, process out-sourcing, and process privacy (e.g., see [3]).

Analytics for Business Processes.
The new, more holistic perspective of data-aware processes can help to provide a new foundation for the field of business intelligence, including new approaches for instrumenting processes to simplify data discovery [64], and new styles of modularity and hierarchy in both the processes and the analytics on them.

Research in process and data will require on-going extensions of the traditional approaches, on both the database and process-centric sides, and also extensions along the lines just mentioned. A new foundational model for modern BPM may emerge, which builds on the artifact and shared-ledger approaches but facilitates a multi-perspective understanding, analogous to the way relational algebra and calculus provide two perspectives on data querying.

One cautionary note is that research in the area of process and data today is hampered by a lack of large sets of examples, e.g., sets of process schemas that include explicit specifications concerning data, and process histories that include how data sets were used and affected. More broadly, increased collaboration between PDM researchers, applied BPM researchers, and businesses would enable more rapid progress towards resolving the concrete problems in BPM faced by industry today.

7. HUMAN-RELATED DATA & ETHICS

More and more "human-related" data is massively generated, in particular on the Web and in phone apps. Massive data analysis, using data parallelism and machine learning techniques, is applied to this data to generate more data. We, individually and collectively, are losing control over this data. We do not know the answers to questions as important as: Is my medical data really available so that I get proper treatment? Is it properly protected? Can a private company like Google or Facebook influence the outcome of national elections? Should I trust the statistics I find on the Web about the crime rate in my neighborhood?

Although we keep eagerly consuming and enjoying more new Web services and phone apps, we have growing concerns about criminal behavior on the Web, including racist, terrorist, and pedophile sites; identity theft; cyber-bullying; and cyber crime. We are also feeling growing resentment against intrusive government practices such as massive e-surveillance even in democratic countries, and against aggressive company behaviors such as invasive marketing, unexpected personalization, and cryptic or discriminatory business decisions.

Societal impact of big data technologies is receiving significant attention in the popular press [11], and is under active investigation by policy makers [68] and legal scholars [19]. It is broadly recognized that this technology has the potential to improve people's lives, accelerate scientific discovery and innovation, and bring about positive societal change. It is also clear that the same technology

can in effect limit business faithfulness to legal and ethical norms.And while many of the issues are political and economical, technology solutions must play an important role in enabling our society to reap ever-greater benefits from big data, while keeping it safe from the risks.

We believe that the main inspiration for the data management field in the 21st century comes from the management of human-related data, with an emphasis on solutions that satisfy ethical requirements.

In the remainder of this section, we will present several facets of ethical data management.

Responsible Data Analysis.

Human-related data analysis needs to be "responsible" — to be guided by humanistic considerations and not simply by performance or by the quest for profit. The notion of responsible data analysis is considered generally in [79] and was the subject of a recent Dagstuhl seminar [5]. We now outline several important aspects of the problem, especially those where we see opportunities for involvement by PDM.
Fairness. Responsible data analysis requires that both the raw data and the computation be "fair", i.e. not biased [43]. There is currently no consensus as to which classes of fairness measures, and which specific formulations, are appropriate for various data analysis tasks. Work is needed to formalize the measures and understand the relationships between them.
Transparency and accountability. Responsible data analysis practices must be transparent [35, 84], allowing a variety of stakeholders, such as end-users, commercial competitors, policy makers, and the public, to scrutinize the data collection and analysis processes, and to interpret the outcomes. Interesting research challenges that can be tackled by PDM include using provenance to shed light on data collection and analysis practices, supporting semantic interrogation of data analysis methods and pipelines, and providing explanations in various contexts, including knowledge-based systems and deep learning.
Diversity. Big data technology poses significant risks to those it overlooks [60]. Diversity [8, 42] requires that not all attention be devoted to a limited set of objects, actors or needs. The PDM community can contribute, for instance, to understanding the connections between diversity and fairness, and to develop methods to manage trade-offs between diversity and conventional measures of accuracy.

Verifying Data Responsibility.

A grand challenge for the community is to develop verification technology to enable a new era of responsible data. One can envision research towards developing tools to help users understand data analysis results (e.g., on the Web), and to verify them. One can also envision tools that help analysts, who are typically not computer scientists nor experts in statistics, to realize responsible data analysis "by design".

Data Quality and Access Control on the Web.

The evaluation of data quality on the Web is an issue of paramount importance when our lives are increasingly guided and determined by data found on the Web. We would like to know whether we can trust particular data we found. Research is needed towards supporting access control on the Web. It may build for instance on cryptography, blockchain technology, or distributed access control [66].

Personal Information Management Systems.

A Personal Information Management System is a (cloud) system that manages all the information of a person. By returning part of the data control to the person, these systems tend to better protect privacy, re-balance the relationship between a person and the major internet companies in favor of the person, and in general facilitate the protection of ethical values [2].

Ethical data management raises new issues for computer science in general and for data management in particular. Because the data of interest is typically human-related, the research also includes aspects from other sciences, notably, cognitive science, psychology, neuroscience, linguistics, sociology, and political sciences. The ethics component also leads to philosophical considerations. In this setting, researchers have a chance for major societal impact, and so they need to interact with policy makers and regulators, as well as with the media and user organizations.

8. LOOKING FORWARD

As illustrated in the preceding sections, the principled, mathematically-based approach to the study of data management problems is providing conceptual foundations, deep insights, and much-needed clarity. This report describes a representative, but by no means exhaustive, family of areas where research on the Principles of Data Management (PDM) can help to shape our overall approach to working with data as it arises across an increasingly broad array of application areas.

The Dagstuhl workshop highlighted two important trends that have been accelerating in the PDM community over the past several years. The first is the

increasing embrace of neighboring disciplines, including especially Machine Learning, Statistics, Probability, and Verification, both to help resolve new challenges, and to bring new perspectives to them. The second is the increased focus on obtaining positive results, that enable the use of mathematically-based insights in practical settings. We expect and encourage these trends to continue in the coming years.

The need for precise and robust approaches for increasingly varied forms of data management continues to intensify, given the fundamental and transformational role of data in our modern society, and given the continued expansion of technical, conceptual, and ethical data management challenges. There is an associated and on-going expansion in the family of approaches and techniques that will be relevant to PDM research. The centrality of data management across numerous application areas is an opportunity both for PDM researchers to embrace techniques and perspectives from adjoining research areas, and for researchers from other areas to incorporate techniques and perspectives from PDM. Indeed, we hope that this report can substantially strengthen cross-disciplinary research between the PDM and neighboring theoretical communities and, moreover, the applied and systems research communities across the many application areas that rely on data in one form or another.

9. REFERENCES

[1] D. Abadi et al. The Beckman report on database research. *Commun. ACM*, 59(2):92–99, 2016.

[2] S. Abiteboul, B. André, and D. Kaplan. Managing your digital life. *Commun. ACM*, 58(5):32–35, 2015.

[3] S. Abiteboul, P. Bourhis, and V. Vianu. Comparing workflow specification languages: A matter of views. *ACM Trans. Database Syst.*, 37(2):10, 2012.

[4] S. Abiteboul et al. Research directions for Principles of Data Management (Dagstuhl perspectives workshop 16151). `https://arxiv.org/abs/1701.09007`.

[5] S. Abiteboul et al., editor. *Data, Responsibly*, volume 16291 of *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl – LZI, 2016, forthcoming.

[6] F. N. Afrati and J. D. Ullman. Optimizing multiway joins in a map-reduce environment. *IEEE Trans. Knowl. Data Eng.*, 23(9):1282–1298, 2011.

[7] A. Agarwal et al. A reliable effective terascale linear learning system. *Journal of Machine Learning Research*, 15:1111–1133, 2014.

[8] R. Agrawal et al. Diversifying search results. In *WSDM*, pages 5–14. ACM, 2009.

[9] M. Akdere et al. The case for predictive database systems: Opportunities and challenges. In *Conference on Innovative Data Systems Research (CIDR)*, pages 167–174. www.cidrdb.org, 2011.

[10] A. Amarilli, P. Bourhis, and P. Senellart. Provenance circuits for trees and treelike instances. In *ICALP*, pages 56–68. Springer, 2015.

[11] J. Angwin et al. Machine bias. ProPublica, May 2016.

[12] M. Aref et al. Design and implementation of the LogicBlox system. In *SIGMOD*, pages 1371–1382. ACM, 2015.

[13] M. Arenas, G. Gottlob, and A. Pieris. Expressive languages for querying the semantic web. In *PODS*, pages 14–26. ACM, 2014.

[14] M. Arenas et al. *Foundations of Data Exchange*. Cambridge University Press, 2014.

[15] M. Arenas et al. Faceted search over RDF-based knowledge graphs. *J. Web Sem.*, 37:55–74, 2016.

[16] A. Artale et al. A cookbook for temporal conceptual data modelling with description logics. *ACM Trans. on Computational Logic*, 15(3):25:1–25:50, 2014.

[17] A. Atserias, M. Grohe, and D. Marx. Size bounds and query plans for relational joins. *SIAM J. Comput.*, 42(4):1737–1767, 2013.

[18] J.-F. Baget et al. On rules with existential variables: Walking the decidability line. *Artificial Intelligence*, 175(9–10):1620–1654, 2011.

[19] S. Barocas and A. D. Selbst. Big data's disparate impact. *California Law Review*, 104, 2016.

[20] P. Beame, P. Koutris, and D. Suciu. Communication steps for parallel query processing. In *PODS*, pages 273–284. ACM, 2013.

[21] M. Benedikt, W. Fan, and F. Geerts. XPath satisfiability in the presence of DTDs. *J. ACM*, 55(2), 2008.

[22] L. Bertossi. *Database Repairing and Consistent Query Answering*. Morgan&Claypool Publishers, 2011.

[23] G. J. Bex et al. Inference of concise regular expressions and DTDs. *ACM Trans. Database Syst.*, 35(2), 2010.

[24] K. Bhattacharya et al. Towards formal analysis of artifact-centric business process models. In *BPM*, pages 288–304. Springer, 2007.

[25] M. Bienvenu et al. Ontology-based data access: A study through Disjunctive Datalog, CSP, and MMSNP. *ACM Trans. Database Syst.*, 39(4):33:1–33:44, 2014.

[26] M. J. Cafarella, D. Suciu, and O. Etzioni. Navigating extracted data with schema discovery. In *WebDB*, 2007.

[27] D. Calvanese, G. De Giacomo, and M. Lenzerini. Conjunctive query containment and answering under description logics constraints. *ACM Trans. on Computational Logic*, 9(3):22.1–22.31, 2008.

[28] D. Calvanese, G. De Giacomo, and M. Montali. Foundations of data-aware process analysis: a database theory perspective. In *PODS*, pages 1–12. ACM, 2013.

[29] D. Calvanese et al. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[30] S. Cebiric, F. Goasdoué, and I. Manolescu. Query-oriented summarization of RDF graphs. *Proc. VLDB Endowment*, 8(12):2012–2015, 2015.

[31] M. Cissé, N. Usunier, T. Artieres, and P. Gallinari. Robust Bloom filters for large multilabel classification tasks. In *NIPS*, 2013.

[32] E. F. Codd. Understanding relations (installment #7). *FDT - Bulletin of ACM SIGMOD*, 7(3):23–28, 1975.

[33] W. Czerwinski et al. The (almost) complete guide to tree pattern containment. In *PODS*, pages 117–130. ACM, 2015.

[34] C. J. Date. *Database in Depth – Relational Theory for Practitioners*. O'Reilly, 2005.

[35] A. Datta, M. C. Tschantz, and A. Datta. Automated experiments on ad privacy settings. *PoPETs*, 2015(1):92–112, 2015.

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

[37] U. Dayal et al. Data integration flows for business intelligence. In *EDBT*, pages 1–11. ACM, 2009.

[38] K. Dembczynski, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic

classifier chains. In *ICML*, pages 279–286. Omnipress, 2010.
[39] D. Deutch and T. Milo. A quest for beauty and wealth (or, business processes for database researchers). In *PODS*, pages 1–12. ACM, 2011.
[40] A. Deutsch, R. Hull, and V. Vianu. Automatic verification of database-centric systems. *SIGMOD Record*, 43(3):5–17, 2014.
[41] A. Deutsch et al. Automatic verification of data-centric business processes. In *ICDT*. ACM, 2009.
[42] M. Drosou and E. Pitoura. DisC diversity: result diversification based on dissimilarity and coverage. *Proc. VLDB Endowment*, 6(1):13–24, 2012.
[43] C. Dwork et al. Fairness through awareness. In *ITCS*, pages 214–226. ACM, 2012.
[44] T. Eiter, T. Lukasiewicz, and L. Predoiu. Generalized consistent query answering under existential rules. In *KR*, pages 359–368. AAAI Press, 2016.
[45] J. Feldman et al. On distributing symmetric streaming computations. In *SODA*, pages 710–719. SIAM, 2008.
[46] G. Gottlob, C. Koch, and R. Pichler. Efficient algorithms for processing XPath queries. *ACM Trans. Database Syst.*, 30(2):444–491, 2005.
[47] G. Gottlob and P. Senellart. Schema mapping discovery from data instances. *J. ACM*, 57(2), 2010.
[48] P. J. Haas and J. M. Hellerstein. Ripple joins for online aggregation. In *SIGMOD*, pages 287–298. ACM, 1999.
[49] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *SIGMOD*, pages 171–182. ACM, 1997.
[50] M. Hepp. The web of data for e-commerce: Schema.org and GoodRelations for researchers and practitioners. In *ICWE*, pages 723–727. Springer, 2015.
[51] X. Hu and K. Yi. Towards a worst-case i/o-optimal algorithm for acyclic joins. In *PODS*. ACM, 2016.
[52] R. Hull and J. Su. NSF Workshop on Data-Centric Workflows, May, 2009. `http://dcw2009.cs.ucsb.edu/report.pdf`.
[53] T. Imielinski and W. Lipski. Incomplete information in relational databases. *J. ACM*, 31(4):761–791, 1984.
[54] K. Jasinska et al. Extreme F-measure maximization using sparse probability estimates. In *ICML*. JMLR.org, 2016.
[55] A. K. Jha and D. Suciu. Probabilistic databases with MarkoViews. *Proc. VLDB Endowment*, 5(11):1160–1171, 2012.
[56] M. Kaminski and E. V. Kostylev. Beyond well-designed SPARQL. In *ICDT*, pages 5:1–5:18. Schloss Dagstuhl – LZI, 2016.
[57] S. Kandel et al. Enterprise data analysis and visualization: An interview study. *IEEE Trans. Vis. Comput. Graph.*, 18(12):2917–2926, 2012.
[58] P. Koutris, P. Beame, and D. Suciu. Worst-case optimal algorithms for parallel query processing. In *ICDT*, pages 8:1–8:18. Schloss Dagstuhl – LZI, 2016.
[59] M. Lenzerini. Data integration: a theoretical perspective. In *PODS*, pages 233–246. ACM, 2002.
[60] J. Lerman. Big data and its exclusions. *Stanford Law Review Online*, 66, 2013.
[61] F. Li, B. Wu, K. Yi, and Z. Zhao. Wander join: Online aggregation via random walks. In *International Conference on Management of Data (SIGMOD)*, pages 615–629. ACM, 2016.
[62] L. Libkin. Incomplete information: what went wrong and how to fix it. In *PODS*, pages 1–13. ACM, 2014.
[63] L. Libkin. SQL's three-valued logic and certain answers. *ACM Trans. Database Syst.*, 41(1):1, 2016.
[64] R. Liu et al. Business artifact-centric modeling for real-time performance monitoring. In *BPM*, pages 265–280, 2011.
[65] M. Marin, R. Hull, and R. Vaculín. Data-centric BPM and the emerging Case Management standard: A short survey. In *BPM Workshops*, pages 24–30, 2012.
[66] V. Z. Moffitt et al. Collaborative access control in WebdamLog. In *SIGMOD*, pages 197–211. ACM, 2015.
[67] G. D. F. Morales and A. Bifet. SAMOA: Scalable advanced massive online analysis. *Journal of Machine Learning Research*, 16:149–153, 2015.
[68] C. Muñoz, M. Smith, and D. Patil. Big data: A report on algorithmic systems, opportunity, and civil rights. *Executive Office of the President, The White House*, May 2016.
[69] N. Ngo, M. Ortiz, and M. Simkus. Closed predicates in description logics: Results on combined complexity. In *KR*, pages 237–246. AAAI Press, 2016.
[70] H. Q. Ngo et al. Worst-case optimal join algorithms: [extended abstract]. In *PODS*, pages 37–48. ACM, 2012.
[71] A. Nigam and N. Caswell. Business Artifacts: An Approach to Operational Specification. *IBM Systems Journal*, 42(3), 2003.
[72] Y. Prabhu and M. Varma. FastXML: a fast, accurate and stable tree-classifier for extreme multi-label learning. In *KDD*, pages 263–272. ACM, 2014.
[73] M. Riondato et al. The vc-dimension of SQL queries and selectivity estimation through sampling. In *ECML/PKDD*, pages 661–676. Springer, 2011.
[74] R. Salakhutdinov and G. E. Hinton. Semantic hashing. *Int. Journal of Approximate Reasoning*, 50(7):969–978, 2009.
[75] M. Schleich, D. Olteanu, and R. Ciucanu. Learning linear regression models over factorized joins. In *SIGMOD*, pages 3–18. ACM, 2016.
[76] P. G. Selinger et al. Access path selection in a relational database management system. In *SIGMOD*, pages 23–34. ACM, 1979.
[77] J. Shin et al. Incremental knowledge base construction using DeepDive. *Proc. VLDB Endowment*, 8(11):1310–1321, 2015.
[78] S. Staworko, J. Chomicki, and J. Marcinkowski. Prioritized repairing and consistent query answering in relational databases. *Ann. Math. Artif. Intell.*, 64(2-3):209–246, 2012.
[79] J. Stoyanovich, S. Abiteboul, and G. Miklau. Data responsibly: Fairness, neutrality and transparency in data analysis. In *EDBT*, pages 718–719. OpenProceedings.org, 2016.
[80] D. Suciu and V. Tannen. A query language for NC. In *PODS*, pages 167–178. ACM, 1994.
[81] D. Suciu et al. *Probabilistic Databases*. Morgan&Claypool Publishers, 2011.
[82] D. Suciu et al. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.
[83] Y. Sun, J. Su, and J. Yang. Universal artifacts. *ACM Trans. on Management Information Systems*, 7(1), 2016.
[84] L. Sweeney. Discrimination in online ad delivery. *Commun. ACM*, 56(5):44–54, 2013.
[85] B. ten Cate, V. Dalmau, and P. G. Kolaitis. Learning schema mappings. *ACM Trans. Database Syst.*, 38(4):28, 2013.
[86] L. Valiant. A theory of the learnable. *Commun. ACM*, 17(11):1134–1142, 1984.
[87] T. L. Veldhuizen. Triejoin: A simple, worst-case optimal join algorithm. In *ICDT*, pages 96–106. OpenProceedings.org, 2014.
[88] K. Weinberger et al. Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120. ACM, 2009.
[89] M. Yannakakis. Algorithms for acyclic database schemes. In *VLDB*, pages 82–94. IEEE, 1981.

A Guide to Formal Analysis of Join Processing in Massively Parallel Systems

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

Dan Suciu
University of Washington
suciu@cs.washington.edu

1. INTRODUCTION

Over the last decade, there has been an enormous increase in the volume of data that is being stored, processed and analyzed. In order to improve the performance of query processing on such amounts of data, many modern data management systems (*e.g.* Spark [23, 28], Hadoop [13, 9, 24], and others [19, 14]) have resorted to the power of parallelism to speed up computation. Parallelism enables the distribution of computation for data-intensive tasks into hundreds, or even thousands of machines, and thus significantly reduces the completion time for several crucial data processing tasks.

In this paper, we present a survey on recent results [18, 4, 5, 17] that study the computational complexity of mulitway join processing in such massively parallel systems. Our goal is twofold. First, we introduce a simple theoretical model, called the MPC (Massively Parallel Computation) model, that allows us to rigorously analyze the computational complexity of various parallel algorithms for query processing. Second, using the MPC model as a theoretical tool, we show how we can design novel algorithms and techniques for multiway join processing, and how we can prove their optimality through tight lower bounds. Our analysis provides a deeper understanding of how much synchronization, communication and data load is required when we compute a multiway join query, and informs of what is possible to achieve under specific system constraints.

Organization. We first present the MPC model in Section 2. Equipped with the model, we describe and rigorously analyze the behavior of several algorithms for the natural join query (Section 3) and the triangle query (Section 4). These two queries cover most of the techniques and algorithms we use for general multiway join queries, which we subsequently describe in Sections 5 and 6. We conclude in Section 7 by discussing some key takeaways of our results.

2. THE MPC PARALLEL MODEL

We introduce here the *Massively Parallel Computation* model, or MPC. We will use MPC to analyze the parallel complexity of various multiway join algorithms, as well as prove lower bounds on the amount of communication and synchronization.

In the MPC model, computation is performed by a cluster of p machines using a shared-nothing architecture. The *shared-nothing* paradigm is widely applied in modern big data management systems [25]. The computation proceeds in *rounds*: each round consists of some local computation followed by global exchange of data between the machines. At the end of each round, the machines have to *synchronize*, *i.e.* wait for all machines to finish before proceeding to the next round.

The input data of size m (in tuples) is initially evenly distributed among the p machines, *i.e.* each machine stores m/p data (this captures how input relations are typically distributed in a distributed file system like HDFS [22]). After the computation is complete, the output result is present in the union of the output of the p machines.

The complexity of an algorithm in the MPC model is characterized by two parameters:

number of rounds (r) This parameter captures the number of synchronization points that an algorithm requires during execution. A smaller number of rounds means that the algorithm can run with less synchronization.

maximum load (L) This parameter is defined as the maximum amount of data that a machine can *receive* at any round during computation. A smaller load means that data is more evenly distributed, and the amount of data communicated is smaller.

An ideal algorithm uses a single round ($r = 1$) and distributes the data evenly without any replication, hence achieving maximum load $L = m/p$. Since this is rarely possible, algorithms for query

processing need to use more rounds, have an increased maximum load, or both. An algorithm with load $L = m$ does not exploit data parallelism at all, since we can send all input data to a single machine and do processing locally. In general, for the problems we will discuss in this paper, the load will be of the form $L = m/p^{1-\varepsilon}$, for some $0 \leq \varepsilon < 1$; we call the parameter ε the *space exponent*. The challenge is to identify the optimal tradeoff between the number of rounds and maximum load for various computational tasks.

Other Parallel Models. There has been a plethora of parallel computation models proposed over the years [1, 15, 11]. The MPC model is closer to the BSP model [26], which also describes synchronous computation; the main difference is that MPC ignores the computation cost as a parameter, and focuses instead on the amount of data communicated at each machine.

3. JOINS IN PARALLEL

We first present how we can compute a natural join between two binary relations,

$$J(x, y, z) = R(x, y), S(y, z).$$

Let m_R, m_S be the sizes of R, S respectively, and assume that initially both R and S are uniformly distributed on the p machines in the cluster; thus, the input load per machine is m/p, with $m = m_R + m_S$. We discuss several algorithms that all work using a single round, and analyze in detail the load L they can achieve.

3.1 Hash Join

Let h be a random hash function that maps attribute values (U) to the domain $[p] = \{1, \cdots, p\}$. To partition the tuples, every machine iterates over its local tuples and sends every tuple $R(a, b)$ to machine $h(b)$, and every tuple $S(c, d)$ to machine $h(c)$. After receiving the tuples, each machine computes the join locally. This basic one-round algorithm was pioneered since the earliest parallel database systems [10] and can be found today in virtually all parallel join implementations [9, 19, 7, 29, 27].

Load Analysis. Since we are distributing m tuples over p machines, one may hope the load to be m/p; unfortunately, this is not always the case. We rule out bad hash functions with many collisions: there is a lot of work on designing good hash functions and we assume to have one. In particular, we will assume that h is a *perfectly random hash function*: this means that h is drawn uniformly at random from the set of hash functions that map the universe U to $[p]$. Such a hash function guarantees that (i) for any value $v \in U$ and every hash bucket $s \in [p]$, $P(h(x) = s) = 1/p$, and (ii) for any distinct values $v_1, v_2, \ldots$, the hash buckets $h(v_1), h(v_2), \ldots$ are independent [20, pp.107].[1]

Observation 1. *The expected load for any machine s is m/p, since each input tuple in R or S is sent to the machine with probability $1/p$ over the random choices of the hash function.*

Define the *degree* d_i of a value $v_i \in U$ as the number of tuples $R(x, y)$ or $S(y, z)$ with $y = v$. We say that the value is *skewed* if its degree is $> m/p$.

Observation 2. *If the input has a skewed value, then for any choice of hash function h, there exists an* overloaded *machine with load $> m/p$. Indeed, all input tuples having the skewed value must be hashed to the same machine, which becomes overloaded.*

Thus, the ideal load is m/p, but if the input is skewed then the load exceeds m/p. It turns out that the converse also holds: if the database is skew-free, then the maximum load is $O(m/p)$ with high probability. However, in the rigorous analysis we must pay an additional $\ln(p)$ factor, either by allowing the load to grow to $\ln(p) \cdot m/p$, or by requiring the maximum degree to be $< m/(p \cdot \ln(p))$. The rest of this subsection provides the full formal analysis, for readers interested in the detailed argument.

Suppose $U = \{v_1, \ldots, v_N\}$ is the universe of possible values; let $d_i = 0$ if v_i does not occur in the input. The input size is $m = \sum_i d_i$.

Theorem 3.1. *[6, 12] Let $X_1, \ldots, X_N \in \{0, 1\}$ be independent and identically distributed (i.i.d.) random variables such that $P(X_i = 1) = 1/p$. Let $d = \max_{i=1}^N d_i$. Then, for any $\delta \geq 0$:*

$$P\left(\sum_{i=1}^{N} d_i X_i \geq (1+\delta)m/p\right) \leq \exp\left(-\frac{m}{pd}h(\delta)\right)$$

where $h(\delta) = (1+\delta)\ln(1+\delta) - \delta$.

We use Theorem 3.1 to analyze the load of the Hash Join algorithm. Fix some machine, and denote X_i the random variable that is 1 if the value v_i is hashed to that server, and 0 otherwise. Then $P(X_i = 1) = 1/p$, and the load at that machine is $\sum_i d_i X_i$. The probability that the maximum load L, over all machines, exceeds the expected load m/p

[1] In practice, we can choose a good enough hash function, for example by having a fixed function h_0, choosing a random seed r, and define $h(v) = h_0(v \text{ xor } r)$.

by a factor $1+\delta$ follows from Theorem 3.1 and the union bound:

$$P(L > (1+\delta)m/p) \leq p \exp\left(-\frac{m}{pd}h(\delta)\right)$$

The main result for Hash Join follows:

PROPOSITION 3.2. *The maximum load L for the Hash Join algorithm is bounded as follows:*

(1) If $d = 1$ (the degree of every value is 1), then for any $\delta < 1$, $P(L > (1+\delta)m/p) \leq p\exp(-\frac{m\delta^2}{3p})$; this probability decreases exponentially fast in the input size m.

(2) If $d \leq m/(3p\ln(p))$, then for any $\delta > 1$, $P(L > (1+\delta)m/p) \leq p^{1-\delta}$; this probability decreases polynomially in the number of machines p.

(3) If $d \leq m/p$, $P(L > \ln(p)m/p) \leq p^{-1}$; this probability also decreases polynomially in the number of machines p.

PROOF. Item (1) follows from the fact that for $\delta < 1$, we have that $h(\delta) \geq \delta^2/3$. Item (2) follows from $h(\delta) \geq \delta/3$ and $p\exp(-\ln(p)\delta) = p^{1-\delta}$. Item (3) follows from setting $1+\delta = \ln p$. Then, since $(1+\delta)\ln(1+\delta) - \delta > 2(1+\delta)$ whenever $\ln(1+\delta) > 3$, we have $p\exp(-2\ln(p)) = p^{-1}$. □

Even though the above analysis is for the case of a simple join, all algorithms in this paper that use one or more hash functions to partition data values into buckets can also be analyzed in a similar way, using a generalization of Theorem 3.1. The main takeaway is that skew-free input means that the maximum load is close to the expected load; on the other hand, skewed input can cause the maximum load to be much larger than the expected load. The particular threshold that determines when a value becomes skewed depends on (i) the query, and (ii) the sizes of the relations.

3.2 Broadcast Join

A simple join algorithm that is immune to skew is the Broadcast Join, where each machine *broadcasts* all its local S-tuples to all other machines. There is no need to reshuffle R, and after the communication step all machines can compute the join locally. This algorithm is also implemented in several systems [21], and performs best when the size of S is much smaller than that of R. The load is $m_R/p + m_S$, regardless of whether the data is skewed or not; this becomes $O(m/p)$ when $m_S = O(m_R/p)$.

3.3 Cartesian Join

The worst behavior for the Hash Join algorithm occurs when both R and S exhibit worst-case skew, *i.e.* they have a single y-value. Then the algorithm will send all tuples of R and S to the same machine, whose load will be $L = m$. In this case, the join degenerates to a *cartesian product*: $R(x) \times S(z)$ (we drop the attribute y since it is constant), and requires a different algorithm.

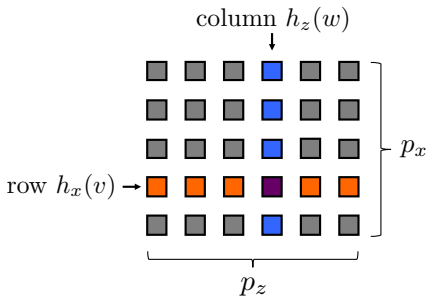


Figure 1: Depiction of the Cartesian Join algorithm. The p machines are organized into a $p_x \times p_z$ rectangle.

Let p_x, p_z be two integers such that $p_x \cdot p_z = p$; we call these numbers *shares* [2]. We organize the p machines into a $p_x \times p_z$ rectangle, where each server is uniquely identified by a pair of numbers $(i,j) \in [p_x] \times [p_z]$ (see Figure 1). Let h_x, h_z be two random hash functions with domains $[p_x]$ and $[p_z]$ respectively. The Cartesian Join works as follows. Initially, it sends every tuple $R(v)$ to all machines of the form $(h_x(v), *)$, and every tuple $S(w)$ to all machines $(*, h_z(w))$. In other words the algorithm broadcasts $R(v)$ to the entire row $h_x(v)$, and broadcasts $S(w)$ to the entire column $h_z(w)$, as seen in Figure 1. After the communication step, each machine computes the cartesian product of its local tuples. The algorithm is correct, because every answer (v,w) of the cartesian product will be in the output of some machine, namely the machine at $(h_x(v), h_z(w))$. There is no skew, since every data value has degree 1 (R and S are sets).

Load Analysis. Since R is partitioned into p_x buckets, each machine receives $O(m_R/p_x)$ tuples from R, and similarly $O(m_S/p_z)$ tuples from S. The load is then

$$L = \frac{m_R}{p_x} + \frac{m_S}{p_z} \geq 2\left(\frac{m_R m_S}{p_x p_z}\right)^{1/2} = 2\left(\frac{m_R m_S}{p}\right)^{1/2}$$

where equality holds when the two terms m_R/p_x, m_S/p_z are equal. This implies that the optimal values for p_x, p_z are:

$$p_x = \left(p\frac{m_R}{m_S}\right)^{1/2} \qquad p_z = \left(p\frac{m_S}{m_R}\right)^{1/2}$$

When $m_R = m_S$, the algorithm organizes the p machines in a $\sqrt{p} \times \sqrt{p}$ square. Otherwise, it allocates

more shares to the larger relation. Since we have to ensure that $p_x, p_z \geq 1$, if $m_R < m_S/p$, we set $p_x = 1, p_z = p$, in which case the algorithm degenerates to the Broadcast Join (R is broadcast) and the load becomes $O(m_S/p)$. Similarly, if $m_S < m_R/p$, we set $p_x = p, p_z = 1$ and the load becomes $O(m_R/p)$. The load of the Cartesian Join will be:

$$L = O\left(\max\left\{\frac{m_R}{p}, \frac{m_S}{p}, \left(\frac{m_R m_S}{p}\right)^{1/2}\right\}\right) \quad (1)$$

We should note that so far we have ignored any rounding issues for the (integer) shares; rounding is challenging problem in practice, and we refer the reader to [8] for further discussion.

Lower Bound. It is easy to see that the above allocation of shares is optimal. We will provide a brief sketch of the argument. Suppose that machine j receives $m_{R,j}, m_{S,j}$ tuples from R and S respectively. Then, it can output at most $m_{R,j} \cdot m_{S,j} \leq (m_{R,j} + m_{S,j})^2/4 = L_j^2/4$ tuples. Since we have p machines, the total output $\sum_j L_j^2/4$ must be at least $m_R m_S$. Therefore,

$$m_R m_S \leq \sum_j L_j^2/4 \leq \sum_j L^2/4 = pL^2/4$$

where the last inequality follows from $L = \max_j L_j$. Thus, $L \geq 2(m_R m_S)^{1/2}/p^{1/2}$.

3.4 Skew Join

All algorithms discussed so far achieve the optimal load only if the data has no skew. To achieve the optimal load in the case of skew, we follow a different approach: we treat the heavy (skewed) and light (non-skewed) values separately.

Heavy Hitters. Recall that the analysis of the Hash Join algorithm fails if some value $y = v$ occurs more than m/p times in R and S, in which case we say it is a *heavy hitter*. We explain briefly how to compute these values. Fix a machine, and call a local value v a *candidate* if its local degree is $> m/p^2$. Each machine computes its candidates and their local degrees. All candidates are broadcast to all machines, and then each machine computes the global degrees of all candidates by adding up the local degrees; the heavy hitters are those candidates with a global degree $> m/p$. Although we need one round to compute the heavy hitters, we will not count this step towards the total number of communication rounds, because the size of data exchange is much smaller, and in many cases the heavy hitters are known already.

The Skew Join Algorithm. Let $\mathcal{H}$ be the set of heavy hitters, and d_h^R, d_h^S be the degrees of $y = h$ in $R(x, y)$ and $S(y, z)$ respectively. Notice that value h may be skewed only in R, or only in S, or in both. Skew Join computes in parallel (1) the join on the light hitters, and (2) the join on the heavy hitters.

The light hitters are computed using the Hash Join; Proposition 3.2 tells us that the load will be $O(\ln(p)m/p) = \tilde{O}(m/p)$; we will use $\tilde{O}$ to hide any logarithmic factors. For the heavy hitters, we assign to each value $h \in \mathcal{H}$ an exclusive group of p_h machines such that $p = \sum_h p_h$, which we use to compute the residual query corresponding to the heavy hitter value $y = h$. The *residual query* $q[h/x]$ is obtained by substituting x with the constant h; in this case, it is the cartesian product $R(x, h) \times S(h, z)$. According to Eq. (1), we can compute $q[h/x]$ with load $L_h = \max\{d_h^R/p_h, d_h^S/p_h, \left(d_h^R d_h^S/p_h\right)^{1/2}\}$. Since the maximum load is $L = \max_h L_h$, to minimize L we choose the p_h to make all L_h equal:

$$p_h = p \cdot \max\left\{\frac{d_h^R}{m_R}, \frac{d_h^S}{m_S}, \frac{d_h^R d_h^S}{\sum_j d_j^R d_j^S}\right\}$$

The maximum load of Skew Join becomes:

$$L = \tilde{O}\left(\max\left\{\frac{m_R}{p}, \frac{m_S}{p}, \sqrt{\frac{\sum_h d_h^R d_h^S}{p}}\right\}\right)$$

By extending the argument from the previous section, it can be shown that this load is optimal, given the degree distribution for d_h^R, d_h^S.

Discussion. We end this section by discussing two special cases of interest. The first is when there is a single heavy hitter h which occurs in all tuples in R and in S, in other words $d_h^R = m_R$, $d_h^S = m_S$. In this case $p_h = p$, and Skew Join degenerates to a Cartesian Join. In this scenario of extreme skew, the speedup is reduced from linear ($\tilde{O}(m/p)$) to sublinear ($\tilde{O}(m/p^{1/2})$). The second special case is when the skew is one-sided, *i.e.* when only values in R are skewed. Then $d_h^S \leq m_S/p$ for all h, and the reader can verify that the load becomes $\tilde{O}(\max\{m_R/p, m_S/p\}) = \tilde{O}(m/p)$. In other words, a join with one-sided skew is no more expensive than a skew free-join, and has linear speedup.

4. TRIANGLES IN PARALLEL

Our next query is the triangle query:

$$\Delta(x, y, z) = R(x, y), S(y, z), T(z, x).$$

We denote the sizes of R, S, T by m_R, m_S, m_T respectively. A traditional parallel query execution engine typically computes the triangle query in two rounds. In the first round, it computes the natural join $U = R(x, y), S(y, z)$ using the parallel Hash

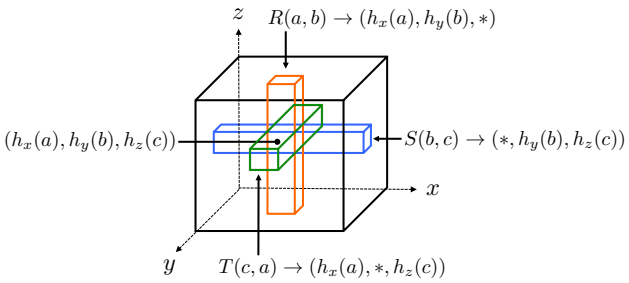


Figure 2: Depiction of the HC algorithm for the triangle query. The p machines are organized into a $p_x \times p_y \times p_z$ cube.

Join algorithm. In the second round, it joins the intermediate relation $U(x, y, z)$ with $T(z, x)$ using again a parallel Hash Join. The issue with such an execution strategy is that the intermediate relation U can be much larger than the input, and shuffling U to join with T can be a costly operation in terms of communication. Motivated by this, we present here two one-round algorithms that compute Δ: one for skew-free input, and the other for skewed data.

4.1 The HyperCube Algorithm

The HyperCube (HC) algorithm was first introduced by Afrati and Ullman [2] in the MapReduce context, with the name SHARES. It first organizes the p machines into a 3-dimensional cube (one dimension per variable). Let p_x, p_y, p_z be the sizes of the dimension for variables x, y, z respectively, called *shares*. Each machine is represented by a distinct point in $\mathcal{P} = [p_x] \times [p_y] \times [p_z]$. Since we have p available machines, we have $p_x \cdot p_y \cdot p_z \leq p$.

The algorithm makes use of three hash functions h_x, h_y, h_z, which map values from the domain U to $[p_x]$, $[p_y]$, $[p_z]$ respectively. During the communication phase of the first round, each tuple $R(a, b)$ is sent to all machines of the form $(h_x(a), h_y(b), *)$. Notice that every tuple is replicated p_z times, since it is sent to p_z machines. We distribute the tuples from S, T similarly: $S(b, c)$ is sent to $(*, h_y(b), h_z(c))$ and $T(c, a)$ to $(h_x(a), *, h_z(c))$. The communication pattern is depicted in Figure 2. During the computation phase, each machine simply performs a local computation of the query Δ on the data fragments it has received.

The correctness of the algorithm comes from the fact that any output triangle $\Delta(a, b, c)$ will be computed and output on the machine with coordinates $(h_x(a), h_y(b), h_z(c))$, since that machine will have all necessary input data sent to it.

Load Analysis. The above algorithm works in one round, and outputs the correct result. We next analyze how to choose the shares and how to compute the maximum load L.

We first focus on how R is distributed. The key observation is that R is partitioned into $p_x \cdot p_y$ buckets. Assuming that R is skew-free (here this means that the degree of each x-value is $\leq m_R/p_x$ and of each y-value $\leq m_R/p_y$), each bucket will be of approximately the same size, and thus the load sent to a machine will be $\tilde{O}(m_R/(p_x p_y))$. Similarly, the load for S will be $\tilde{O}(m_S/(p_y p_z))$, and for T it will be $\tilde{O}(m_T/(p_x p_z))$. The total load L will be at least the maximum of the three quantities. To find the optimal shares, we can construct an optimization problem, where the objective is to minimize L under the constraints that $L \geq m_R/(p_x \cdot p_y)$ (similarly for S, T), and also $p_x \cdot p_y \cdot p_z \leq p$.

In order to solve the above optimization program, we transform it into a linear program (LP) by taking logarithms with base p on both sides. Let $\lambda = \log_p L$ and $e_i = \log_p p_i$ for $i = \{x, y, z\}$. Since $p_i = p^{e_i}$, we call e_i the *share exponent*. The LP takes the following form:

$$\begin{aligned} \textbf{minimize} \quad & \lambda \\ \textbf{subject to} \quad & e_x + e_y + \lambda \geq \log_p m_R \\ & e_y + e_z + \lambda \geq \log_p m_S \\ & e_x + e_z + \lambda \geq \log_p m_T \\ & e_x + e_y + e_z \leq 1 \\ & e_x, e_y, e_z, \lambda \geq 0 \end{aligned}$$

The solution of the above LP obtains the best possible share exponents for the particular sizes of the input relations for the skew-free case. By using the principle of LP duality, the optimal load can also be expressed as the maximum value of the following (non-linear) optimization problem:

$$\begin{aligned} \textbf{maximize} \quad & \left(\frac{m_R^{u_R} m_S^{u_S} m_T^{u_T}}{p} \right)^{1/(u_R+u_S+u_T)} \\ \textbf{subject to} \quad & u_R + u_S \leq 1 \\ & u_S + u_T \leq 1 \\ & u_R + u_T \leq 1 \\ & u_R, u_S, u_T \geq 0 \end{aligned}$$

The vector (u_R, u_S, u_T) forms a *fractional edge packing* of the query Δ; we will explain in the next section how this notion is defined for any multiway join query. By examining the vertices of the polytope formed by the edge packings, we can show that

the optimal load is:

$$L = \tilde{O}\left(\max\left\{\frac{m_R}{p}, \frac{m_S}{p}, \frac{m_T}{p}, \frac{(m_R m_S m_T)^{1/3}}{p^{2/3}}\right\}\right)$$

The first three terms are obtained through the edge packings $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ respectively, while the last term through $(1/2,1/2,1/2)$.

Discussion. To keep the discussion simple, let us consider the case where $m_R < m_S = m_T = m$. There are two cases:

- $p < m/m_R$. In this case, the optimal packing is either $(0,1,0)$, or $(0,0,1)$, and the load becomes $\tilde{O}(m/p)$. To achieve this linear speedup, the HC algorithm allocates shares $p_x = p_y = 1$, $p_z = p$, *i.e.* it performs a parallel Hash Join of S, T on z, and broadcasts R.
- $p \geq m/m_R$. In this case, the optimal packing is $(1/2,1/2,1/2)$, and the load becomes $L = (m_R m_S m_T)^{1/3}/p^{2/3}$. To achieve this load, the HC algorithm allocates shares as follows: $p_x = p_y = (m_R/m)^{1/3}p^{1/3}$, $p_z = (m/m_R)^{2/3}p^{1/3}$.

In the case where all input relations have the same size, *i.e.* $m_R = m_S = m_T = m$, the shares are equal: $p_x = p_y = p_z = p^{1/3}$, and the load becomes $\tilde{O}(m/p^{2/3})$. The speedup now is not linear; however, as we will see in Section 5, this is the best we can hope for among one-round algorithms.

4.2 Triangles with Skew

The HC algorithm is optimal only for skew-free data. But how do we define skewed values (heavy hitters) for the triangle? For the sake of simplicity, assume that all relations have size equal to m. A value for x, y or z is a *heavy hitter* if it has degree $> m/p^{1/3}$ for any of the two relations it belongs; otherwise, it is light. To achieve optimal load for skewed data, we follow the same approach as the Skew Join algorithm, by treating heavy and light values separately.

The algorithm will deal with the light values by running the vanilla HC algorithm, achieving maximum load $\tilde{O}(m/p^{2/3})$. To handle the heavy hitters, we distinguish two cases.

Case 1. In this case, we handle the tuples that have values with degree $\geq m/p$ in at least two variables. Without loss of generality, suppose that both x, y are heavy in at least one of the two relations they belong to. The observation is that there are at most $2p$ such heavy values for each variable, and hence we can send all tuples in R with both x, y heavy (at most $4p^2$) to all machines. Then, it remains to compute the query $S'(y,z), T'(z,x)$, where x and y can take only p values. We can do this by running the standard Hash Join algorithm; since the degree of z-values will be at most p for each relation, there is no skew and the maximum load will be $\tilde{O}(m/p)$.

Case 2. In this case, we handle the remaining output: this includes the tuples where one value has degree $\geq m/p^{1/3}$, and the other values have degree $\leq m/p$. Without loss of generality, assume that we want to compute the query for the x-values that are heavy in either R or T. Observe that there are at most $2p^{1/3}$ such heavy hitters. If $\mathcal{H}_x$ denotes the set of heavy hitter values for variable x, the residual query $q[h/x]$ for each $h \in \mathcal{H}_x$ is:

$$q[h/x] = R(h,y), S(y,z), T(z,h)$$

which is equivalent to computing the query $q_x(y,z) = R'(y), S(y,z), T'(z)$ with input sizes d_h^R, m, d_h^T respectively. As with the Skew Join, we allocate an exclusive group of p_h servers to compute $q[h/x]$ for each $h \in \mathcal{H}_x$. Having Case 1 ensures that the input to the residual query is skew-free, hence the load L_h for a particular value h is given by:

$$L_i = O\left(\max\left\{\frac{m}{p_h}, \sqrt{\frac{d_h^R d_h^T}{p_h}}\right\}\right)$$

We can now set p_h similar to how we chose the values for the Skew Join:

$$p_h = p \cdot \max\left\{\frac{1}{p^{1/3}}, \frac{d_h^R d_h^S}{\sum_{j\in\mathcal{H}_x} d_j^R d_j^S}\right\}$$

This assignment obtains the following load:

$$L = \tilde{O}\left(\max\left\{\frac{m}{p^{2/3}}, \sqrt{\frac{\sum_h d_h^R d_h^S}{p}}\right\}\right)$$

Summing up all the cases, we obtain that the load of the 1-round algorithm for computing triangles is:

$$\tilde{O}\left(\max\left\{\frac{m}{p^{2/3}}, \sqrt{\frac{\sum_h d_h^R d_h^S}{p}}, \sqrt{\frac{\sum_h d_h^R d_h^T}{p}}, \sqrt{\frac{\sum_h d_h^S d_h^T}{p}}\right\}\right)$$

The above algorithm is optimal for computing triangles in one round when the degree distribution is given. Observe that in the case of extreme skew, the load increases from $\tilde{O}(m/p^{2/3})$ in the skew-free case to $\tilde{O}(m/p^{1/3})$.

5. GENERAL JOINS IN ONE ROUND

We have seen so far how to analyze parallel algorithms for computing the join and triangle query. In this section, we generalize our techniques to compute general multiway join queries in *one round*.

5.1 The General HyperCube Algorithm

The algorithm we present here is a generalization of the HyperCube algorithm for triangles. We will consider a multiway join query without projections, called also a *full conjunctive query*:

$$q(x_1, \ldots, x_k) = S_1(\bar{x}_1), \ldots, S_\ell(\bar{x}_\ell)$$

Throughout this section, the size of relation S_j will be m_j. The HC algorithm assigns to each variable x_i, where $i = 1, \ldots, k$, a *share* p_i, such that $\prod_{i=1}^{k} p_i = p$. Each machine is then represented by a distinct point $\mathbf{y} \in \mathcal{P}$, where $\mathcal{P} = [p_1] \times \cdots \times [p_k]$; in other words, machines are mapped into points of a k-dimensional hypercube.

During communication, we use k independently chosen hash functions $h_i : U \to [p_i]$ and send each tuple t of relation $S_j(x_{i_1}, \ldots, x_{i_a})$ to all machines in the destination subcube of t:

$$\mathcal{D}(t) = \{\mathbf{y} \in \mathcal{P} \mid \forall m \in [a] : h_{i_m}(t[i_m]) = \mathbf{y}_{i_m}\}$$

Then, each machine locally computes the query q for the subset of the input that it has received.

The correctness of the HC algorithm follows from the observation that, for every potential output tuple $(a_1, \ldots, a_k)$, machine $(h_1(a_1), \ldots, h_k(a_k))$ contains all the necessary information to decide whether it belongs in the answer or not. Observe also that the choice of $p_1, \ldots, p_k$ gives a different parametrization of the HC algorithm. To analyze the load of the HC algorithm for a particular choice of shares, we will use a generalization of Proposition 3.2.

Definition 5.1. *Let $\mathbf{p} = (p_1, \ldots, p_k)$ be a vector of shares. If for every relation S_j and every tuple of values t over $A \subseteq \bar{x}_j$ the degree of t in S_j is at most $m_j / \prod_{x_i \in A} p_i$, we say that the input is* skew-free *w.r.t.* $\mathbf{p}$.

Proposition 5.2. *Let $\mathbf{p} = (p_1, \ldots, p_k)$ be shares of the HC algorithm. If the input is skew-free w.r.t $\mathbf{p}$, the maximum load is (with high probability)*

$$\tilde{O}\left(\max_j \frac{m_j}{\prod_{i:x_i \in S_j} p_i}\right)$$

The above analysis provides us with a tool to choose the best shares for the HC algorithm. Recall from Section 4 that we can write $p_i = p^{e_i}$, where $e_i \in [0, 1]$ is called the *share exponent* for x_i, and let $\lambda = \log_p L$. Then, we compute the optimal shares by optimizing the following LP:

$$\begin{aligned} \text{minimize} \quad & \lambda \\ \text{subject to} \quad & \sum_{i \in [k]} -e_i \geq -1 \\ & \forall j \in [\ell] : \sum_{x_i \in S_j} e_i + \lambda \geq \log_p(m_j) \\ & \forall i \in [k] : e_i \geq 0, \quad \lambda \geq 0 \end{aligned} \tag{2}$$

Observe that this is the general form of the LP in Section 4. We can now describe our main result on the performance of the HC algorithm. A *fractional edge packing* of a query q is a non-negative weight assignment $\mathbf{u} = (u_1, \ldots, u_j)$ to each relation S_j such that for every variable x_i, $\sum_{j:x_i \in S_j} u_j \leq 1$. Let $\mathsf{pk}(q)$ be the set of all edge packings for q.

Theorem 5.3 (Upper Bound [4]). *Given a query q and p machines, let $\mathbf{e} = (e_1, \ldots, e_k)$ be the optimal solution to (2). Let $p_i = p^{e_i}$, and suppose that the input data is skew-free w.r.t. to $\mathbf{p} = (p_1, \ldots, p_k)$. Then the HC algorithm with shares $\mathbf{p}$ achieves w.h.p.*

$$L = \tilde{O}\left(\max_{\mathbf{u} \in \mathsf{pk}(q)} \left(\frac{\prod_{j=1}^{\ell} m_j^{u_j}}{p}\right)^{1/\sum_j u_j}\right)$$

.

A case of special interest is when all input relations have the same cardinalities, i.e. $m_1 = m_2 = \cdots = m_\ell = m$. In this case the load is $\tilde{O}(m/p^{1/\tau^*})$. Here, τ^* is the fractional edge packing number, defined as $\tau^* = \max_{u \in \mathsf{pk}(q)} \sum_j u_j$.

5.2 Optimality

The HC algorithm is optimal (up to logarithmic factors) among one-round algorithms. Indeed, we can show that there exists a family of skew-free inputs with maximum degree one (called *matching databases*) such that no one-round algorithm can achieve a bound better than the one in Theorem 5.3.

Theorem 5.4 (Lower Bound [5]). *Given a query q, any (randomized) algorithm that computes q correctly in one round with p machines must have maximum load*

$$L = \Omega\left(\max_{\mathbf{u} \in \mathsf{pk}(q)} \left(\frac{\prod_{j=1}^{\ell} m_j^{u_j}}{p}\right)^{1/\sum_j u_j}\right)$$

5.3 Dealing with Skew

If the input data is not skew-free with respect to the optimal share allocation, the HC algorithm does not achieve the optimal anymore. Instead, we

have to use techniques described in the previous two sections to deal with skew, by separating the heavy and light hitters and considering the residual queries. We refer the interested reader to [5] for further details on how to approach general join queries. It still remains an open problem to find a load-optimal one-round algorithm for any input, where optimality in this case is defined with respect to a fixed degree distribution of the input.

6. BEYOND ONE ROUND

In this section, we discuss the analysis of multi-way join algorithms for multiple rounds. Throughout this section, we assume that all input relations have the same size m.

6.1 A Worst-Case Lower Bound

We first present a lower bound for the best possible load of any multi-round algorithm. Atserias, Grohe, and Marx [3] have shown the following result, known as the *AGM bound*: if all input relations have size $\leq m$, then the size of the query is $\leq m^{\rho^*}$, where ρ^* is the *fractional edge covering number* of the query q. Moreover, the AGM bound is tight, in the sense that there exists a "worst case" input database with relation sizes $\leq m$, on which the query returns an answer of size m^{ρ^*}. The fractional edge covering number is defined as the maximum value of $\sum_j u_j$, where the numbers $u_j \geq 0$ are associated to the input relations R_j, and must satisfy the following conditions, for every variable x_i: $\sum_{j:x_i \in R_j} u_j \geq 1$.

THEOREM 6.1 ([17]). *Let q be a join query. Then, there exists a family of instances where relations have the same size m, such that every MPC algorithm that computes q with p machines using a constant number of rounds requires load $\Omega(m/p^{1/\rho^*})$.*

PROOF SKETCH. Assume that an algorithm $\mathcal{A}$ computes q with load L in r rounds. Since each machine receives at most $r \cdot L$ tuples from each relation S_j, we can use the AGM bound to argue that the total number of output tuples will be at most $p(r \cdot L)^{\rho^*}$. If the input database is the worst case input (or close to it), then $\mathcal{A}$ must output m^{ρ^*} tuples and therefore $p(r \cdot L)^{\rho^*} \geq m^{\rho^*}$, or equivalently $L \geq m/(rp^{1/\rho^*})$. □

For the triangle query, since $\rho^* = 3/2$, any algorithm with a constant number of rounds must use load $\Omega(m/p^{2/3})$. Notice that for the triangle query, $\tau^* = \rho^* = 3/2$, and thus we can use the 1-round algorithm from the previous section to compute the query on skew-free data with load $\tilde{O}(m/p^{2/3})$.

Recall that $m/p^{1/\tau^*}$ is the optimal load for one-round algorithms and skew-free data, while $m/p^{1/\rho^*}$ is a lower bound for multi-round algorithms and arbitrary data. In general, there is no relationship between τ^* and ρ^*: for example, the join query has $\tau^* = 1 < \rho^* = 2$, while the query $R(x), S(x,y), T(y)$ has $\tau^* = 2 > \rho^* = 1$.

The upper bound for multi-rounds and arbitrary data is open, except for the case when all input relations are binary: in this case $\tau^* \leq \rho^*$ and it has been shown [16] that the optimal load is precisely $\tilde{O}(m/p^{1/\rho^*})$. Intuitively, after using one round to compute the query on the skew-free data fragment with a load $m/p^{1/\tau^*}$, we can exploit the additional rounds to compute the query on the skewed data values with load $m/p^{1/\rho^*}$. We illustrate this idea next on the triangle query.

6.2 The Triangle Revisited

Recall that the HC algorithm computes Δ on skew-free data with load $\tilde{O}(m/p^{2/3})$ in one round. We will show here that we can compute the query on arbitrary data using the same load in 2 rounds. The main component is a parallel algorithm that computes a semi-join query optimally in a single round, independent of skew.

PROPOSITION 6.2 ([17]). *The semi-join query $q_1(x,y) = R(x), S(x,y)$ can be computed in one round with maximum load $\tilde{O}(m/p)$. Query $q_2(x,y) = R(x), S(x,y), T(y)$ can be computed in two rounds with load $\tilde{O}(m/p)$.*

PROOF SKETCH. To compute q_1 we use Skew Join, outlined in Section 3.4. The data is skewed only on one side, because $R(x)$ is a set; therefore, Skew Join has a load of $\tilde{O}(m/p)$. To compute q_2, in the first round we compute the semi-join $A(x,y) = R(x), S(x,y)$: importantly, the size of the result is no larger than m. In the second round, we compute the semi-join $q(x,y) = A(x,y), T(y)$. Both steps have a load of $\tilde{O}(m/p)$. □

We now describe the 2-round algorithm for computing Δ on arbitrary input data. Recall from Section 4 that a value h is a *heavy hitter* if for some relation the degree of h exceeds $m/p^{1/3}$. For the light values, we can run the HC algorithm in one round and obtain load $\tilde{O}(m/p^{2/3})$.

It remains to output the tuples for which at least one variable has a heavy value. Without loss of generality, consider the case where variable x has heavy values and observe that there are at most $2p^{1/3}$ such heavy values for x ($p^{1/3}$ for R and $p^{1/3}$ for T). For each heavy value h, we assign an *exclusive* set of $p' = p^{2/3}$ servers to compute the query $q[h/x] =$

$R(h, y), S(y, z), T(z, h)$, which is equivalent to computing the residual query $q' = R'(y), S(y, z), T'(z)$. Recall that in the analysis of Section 4, it was expensive to compute this residual query in a single round. However, by Proposition 6.2 using two rounds we can compute each q' with load $\tilde{O}(m/p') = \tilde{O}(m/p^{2/3})$. We have thus shown:

THEOREM 6.3. *The triangle query Δ on input with relation sizes equal to m can be computed by an MPC algorithm in two rounds with $\tilde{O}(m/p^{2/3})$ load, under any input data distribution.*

The 2-round algorithm achieves a better load than any 1-round algorithm in the worst-case scenario. Indeed, there exists an $\Omega(m/p^{1/2})$ lower bound for 1-round algorithms on arbitrary data. By using an additional round, we can beat this bound and achieve a lower load. This confirms our intuition that with more rounds we can reduce the maximum load, even in the case of skewed data.

6.3 General Join Queries

Recall that every algorithm, regardless of the number of rounds, must have load $\Omega(m/p^{1/\rho^*})$. It is currently not known whether this load is optimal for general join queries, but it has been proven to be optimal in [17, 16] for queries where all relations have arity 2.

Our algorithm for computing the triangle query suggests the following general strategy for computing an arbitrary query on arbitrary (possibly skewed) data: (1) compute the query on the light hitters, using one round and load $\tilde{O}(m/p^{1/\tau^*})$. (2) compute the query on the heavy hitters using multiple rounds. If $\tau^* \leq \rho^*$ *and* step (2) can be done with a load of $m/p^{1/\rho^*}$, this algorithm is optimal.

Recall that τ^* and ρ^* are the fractional edge packing number, and the fractional edge covering number of the query respectively. If all relations in the query have arity 2, then the query hypergraph is a graph, and in that case $\tau^* \leq \rho^*$. [17] described an algorithm for step (2) with load $\tilde{O}(m/p^{1/\rho^*})$ for chain queries and for cycles (including the algorithm for the triangle query that we discussed earlier). [16] described a non-trivial extension of this algorithm to arbitrary queries where all relational symbols have arity 2, still having load $\tilde{O}(m/p^{1/\rho^*})$. In all these cases the algorithm consisting of steps (1) and (2) is optimal, since $\tau^* \leq \rho^*$.

It is currently open how to compute optimally queries when $\tau^* > \rho^*$. If the optimal load is indeed $m/p^{1/\rho^*}$, then we need an entirely new approach to compute the query over the light hitters, with a better load than the HC algorithm. To see such an example, consider the query

$$q_3(x_1, x_2, x_3, y_1, y_2, y_3) = R(x_1, x_2, x_3), S_1(x_1, y_1), S_2(x_2, y_2), S_3(x_3, y_3), T(y_1, y_2, y_3).$$

where $\tau^* = 3$ and $\rho^* = 2$. When applied to the light hitters, the HC algorithm allocates equal shares of $p^{1/6}$ to all variables, thus the load is $\tilde{O}(m/p^{1/3})$; this load is also a lower bound for one round algorithms. Yet for multiple rounds the best lower bound is $\Omega(m/p^{1/\rho^*}) = \Omega(m/p^{1/2})$: it is open whether q_3 can be computed with this load in multiple rounds.

7. CONCLUSION

In this paper, we presented recent results on the communication cost of algorithms for computing multiway join queries in modern big data analytics systems. We conclude by discussing some of the key takeaways of our results.

1. The communication cost for 1-round join queries can be reduced by using multi-dimensional hash-partitioning. Each variable x is allocated a share p_x, and the values of x are hash partitioned into p_x buckets. When all relations have the same size m, then the optimal load of any 1-round algorithm is $O(m/p^{1/\tau^*})$, where τ^* is the value of the optimal fractional edge packing.

2. In any partitioning scheme of a relation R, the expected number of tuples received by a machine is the relation size divided by the number of buckets into which it is partitioned. For example, for the triangle query, $R(x, y)$ is partitioned into $p_x p_y = p^{2/3}$ buckets, hence the load is $m/p^{2/3}$.

3. Skew occurs when a data value for some variable x overflows one of its buckets. In a multi-dimensional hash-partitioning scheme, tuples with the same x-value are distributed to fewer buckets than the entire relation, which makes the algorithm quite resilient to skew. For example, in a triangle query a value for variable x is skewed if it occurs more than $m/p^{1/3}$ times. For concrete numbers, assuming $p = 1000$ machines, R is partitioned into 100 buckets, yet we can tolerate values with degree up to $m/10$, much larger than the expected bucket size of $m/100$. There are at most 10 skewed x-values (heavy hitters), regardless of the size of R.

4. Multiple communication rounds can be used effectively to compute queries over skewed data. It is currently open what the optimal load of a multi-round algorithm is. The best lower bound is $\Omega(m/p^{1/\rho^*})$, and this bound is known to be optimal for queries where all relations have arity 2 and the same size.

8. REFERENCES

[1] F. N. Afrati, A. D. Sarma, S. Salihoglu, and J. D. Ullman. Upper and lower bounds on the cost of a map-reduce computation. *PVLDB*, 6(4):277–288, 2013.

[2] F. N. Afrati and J. D. Ullman. Optimizing joins in a map-reduce environment. In *EDBT*, pages 99–110, 2010.

[3] A. Atserias, M. Grohe, and D. Marx. Size bounds and query plans for relational joins. In *FOCS*, pages 739–748, 2008.

[4] P. Beame, P. Koutris, and D. Suciu. Communication steps for parallel query processing. In *PODS*, pages 273–284, 2013.

[5] P. Beame, P. Koutris, and D. Suciu. Skew in parallel query processing. In *PODS*, pages 212–223, 2014.

[6] P. Beame, P. Koutris, and D. Suciu. Communication cost in parallel query processing. *CoRR*, abs/1602.06236, 2016.

[7] R. Chaiken, B. Jenkins, P. Larson, B. Ramsey, D. Shakib, S. Weaver, and J. Zhou. SCOPE: easy and efficient parallel processing of massive data sets. *PVLDB*, 1(2):1265–1276, 2008.

[8] S. Chu, M. Balazinska, and D. Suciu. From theory to practice: Efficient join query evaluation in a parallel database system. In *SIGMOD*, pages 63–78, 2015.

[9] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In *OSDI*, pages 137–150, 2004.

[10] D. J. DeWitt, S. Ghandeharizadeh, D. A. Schneider, A. Bricker, H. Hsiao, and R. Rasmussen. The gamma database machine project. *IEEE Trans. Knowl. Data Eng.*, 2(1):44–62, 1990.

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

[12] E. Gribkoff and D. Suciu. Slimshot: In-database probabilistic inference for knowledge bases. *PVLDB*, 9(7):552–563, 2016.

[13] Hadoop. `http://hadoop.apache.org/`.

[14] D. Halperin, V. T. de Almeida, L. L. Choo, S. Chu, P. Koutris, D. Moritz, J. Ortiz, V. Ruamviboonsuk, J. Wang, A. Whitaker, S. Xu, M. Balazinska, B. Howe, and D. Suciu. Demonstration of the myria big data management service. In *SIGMOD*, pages 881–884, 2014.

[15] H. J. Karloff, S. Suri, and S. Vassilvitskii. A model of computation for mapreduce. In *SODA*, pages 938–948, 2010.

[16] B. Ketsman and D. Suciu. A worst-case optimal multi-round algorithm for parallel computation of conjunctive queries, 2017. To appear in PODS.

[17] P. Koutris, P. Beame, and D. Suciu. Worst-case optimal algorithms for parallel query processing. In *ICDT*, pages 8:1–8:18, 2016.

[18] P. Koutris and D. Suciu. Parallel evaluation of conjunctive queries. In *PODS*, pages 223–234, 2011.

[19] S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton, and T. Vassilakis. Dremel: Interactive analysis of web-scale datasets. *PVLDB*, 3(1):330–339, 2010.

[20] M. Mitzenmacher and E. Upfal. *Probability and computing - randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

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

[22] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *MSST*, pages 1–10, 2010.

[23] Apache spark. `http://spark.apache.org/`.

[24] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy. Hive - a warehousing solution over a map-reduce framework. *PVLDB*, 2(2):1626–1629, 2009.

[25] M. S. University and M. Stonebraker. The case for shared nothing. *Database Engineering*, 9:4–9, 1986.

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

[27] J. Wang, T. Baker, M. Balazinska, D. Halperin, B. Hayes, B. Howe, D. Hutchinson, S. Jain, R. Maas, P. Mehta, D. Moritz, B. Myers, J. Ortiz, D. Suciu, A. Whittaker, and S. Xu. The Myria big data management and analytics system and cloud services, 2017. To appear in CIDR.

[28] R. S. Xin, J. Rosen, M. Zaharia, M. J. Franklin, S. Shenker, and I. Stoica. Shark: SQL and rich analytics at scale. In *SIGMOD*, pages 13–24, 2013.

[29] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. In *HotCloud*, 2010.

A Closer Look at Variance Implementations In Modern Database Systems

Niranjan Kamat Arnab Nandi
Computer Science & Engineering
The Ohio State University
{kamatn,arnab}@cse.osu.edu

ABSTRACT

Variance is a popular and often necessary component of aggregation queries. It is typically used as a secondary measure to ascertain statistical properties of the result such as its error. Yet, it is more expensive to compute than primary measures such as `SUM`, `MEAN`, and `COUNT`.

There exist numerous techniques to compute variance. While the definition of variance implies two passes over the data, other mathematical formulations lead to a single-pass computation. Some single-pass formulations, however, can suffer from severe precision loss, especially for large datasets.

In this paper, we study variance implementations in various real-world systems and find that major database systems such as PostgreSQL and most likely System X, a major commercial closed-source database, use a representation that is efficient, but suffers from floating point precision loss resulting from catastrophic cancellation. We review literature over the past five decades on variance calculation in both the statistics and database communities, and summarize recommendations on implementing variance functions in various settings, such as approximate query processing and large-scale distributed aggregation. Interestingly, we recommend using the mathematical formula for computing variance if two passes over the data are acceptable due to its precision, parallelizability, and surprisingly computation speed.

1. INTRODUCTION

New large-scale distributed data management and analytics systems are being developed at a rapid pace, with the scalability aspect of computation being their predominant development focus (excepting [10]). Comparatively lesser efforts have been expended on ensuring numerical correctness and stability of algorithms. While such an approach can result in the queries being answered more quickly, it can also cause the computation to have a higher level of numerical imprecision.

The concern of achieving numerical stability and precision is pertinent in numerous computational scenarios; it is especially important in variance calculation, which has an ubiquitous presence in large-scale analytics and is known to suffer from precision issues [4]. Variance is an important aggregate function and an essential tool in sampling-based aggregation queries. Typically used as a secondary measure, it augments measures such as `AVERAGE` and provides an insight into data distribution beyond the primary measure. Computation of variance, however, is susceptible to precision loss when the variance is much smaller than the mean [1].

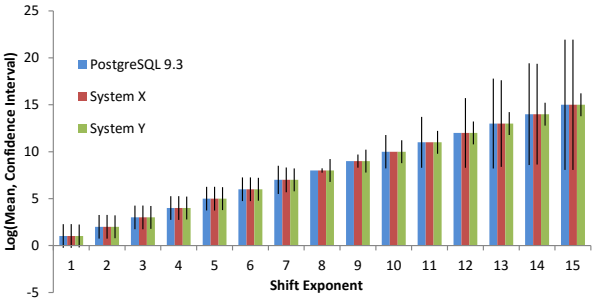


Figure 1: Effect of Variance Error on T-Test Confidence Intervals: As the magnitude of values increases (x-axis, true margin of error is kept consistent for each dataset), mean is expected to increase, and size of error bars is expected to stay the same. However, PostgreSQL and System X error bars ($\alpha = 0.05$) vary widely, while System Y has correct error bars (details in Section 1.1).

There exist several techniques to compute variance. The standard formula uses two passes and provides an accurate estimate (*Two Pass*). Due to its perception of being more expensive, other techniques have been developed that require a single pass over the data. One such formula, while fast, is known to suffer from precision loss (*Textbook One Pass*) due to *catastrophic cancellation* [4], an undesirable effect of a floating point operation that causes relative error to far exceed absolute error. Figure 1 demonstrates this problem. As a side note, this problem affects calculators as well [4].

Another formula (*Updating*), as recommended by Knuth [8], has found a strong foothold in the database community, with numerous implementations citing him. However, this formula is constrained by the fact that it can only incorporate a single value into the current running estimates. It is unable to combine the estimates from different subsets of data.

Given the rise of large-scale data processing, massive multi-core support and availability of GPUs, it is prudent to consider representations such as *Pairwise Updating*, that can combine partial results at a larger scale instead of incrementally incorporating a single data point. Further, *Pairwise Updating* is also known to provide better precision for both single ([1]) and double precision input (Section 4).

Contributions & Outline:

- We catalog usage of different variance formulas in various open source database systems (Table 2).
- We experiment with different closed source and open source databases to investigate precision loss issues. We find that precision of PostgreSQL and System X deteriorates the most. After looking at the PostgreSQL source code, we can verify that it uses *Textbook One Pass*, and hypothesize that System X does so as well.
- We empirically study the accuracy of the different representations under varying additive shifts and dataset sizes including a hitherto unstudied one, which we call *Total Variance*.
- We recommend using *Two Pass* if performing two passes over the data is acceptable (Section 5), which seems counter-intuitive, but works due to its computational simplicity.

In the next subsection, we look at the adverse effects of imprecise variance calculation. Section 2 presents different variance representations and their properties. We then note the representations used by modern databases in Section 3. Section 4 lists our analysis of the behavior of the different formulas (using double precision input compared with single precision in Chan et al. [1]). We conclude with our recommendations for choosing an appropriate variance representation in current environments.

1.1 Impact of Variance Calculations

Due to the pervasive use of variance, a loss of precision can have an impact in a variety of different domains. In the following paragraphs, we look at some use cases where the lack of precision in variance calculation can have adverse consequences.

Incorrect Output: It is possible to experimentally observe the loss of precision as incorrect output. To illustrate the pitfalls in using *Textbook One Pass*, 100 values were generated from a $Uniform(0, 1)$ distribution and shifted by $10^{Shift\ Exponent}$ for *Shift Exponent* varying from 1 to 15. The variance as a result of data being shifted should be similar to the one without any shift. We verify this by adding and subtracting the shift exponent and note that the variance of the resultant dataset was close to the true sample variance. However, Figure 1, which depicts the sample mean and confidence interval, shows that PostgreSQL and System X suffer from variance calculations being susceptible to precision loss due to the shift. We know that PostgreSQL uses *Textbook One Pass* and the pattern of the erroneous calculations displayed by both hints towards System X using it as well. In contrast, other database systems suffered minor precision loss as expected (these results are not shown since they do not add any additional information to the figure). It should be noted that System Y was found to be highly immune to precision loss.

Visualization: Erroneous variance calculation can have a notable impact on visualizations as shown by Figure 1. While error bars should be similar, they instead vary widely and inaccurately for higher shift values for PostgreSQL and System X. We also found *Datavore*, which powers the *Profiler* visualization system [7], to use *Two Pass*.

Negative Variance: It is possible for variance to be negative while using *Textbook One Pass* – a theoretically impossible result. We observed in the PostgreSQL source code that variance is set to zero, if negative. Figure 1 shows numerous values of 0 (missing error bars) for PostgreSQL (shift exponent 8, 9, and 12) and System X (shift exponents 10 and 11), providing evidence of System X employing a similar strategy for handling negative variance values and using *Textbook One Pass*.

Decision Support Systems: As a building block in popular algorithms, flaws in variance implementations can have far-reaching impacts, e.g., in hypothesis testing, which is an integral part of decision support systems. Having imprecise or incorrect variance estimates can greatly change the result of hypothesis testing.

Data Mining: Variance is an important tool in statistical analysis and machine learning algorithms such as Gaussian Naive Bayes, or Mixture of Gaussians based algorithms such as background modeling, clustering, or topic modeling. For example, we found usage of *Textbook One Pass* within a graphics library of the *R* language. Similarly, MADlib [3] was also found to have a call to the PostgreSQL variance function: thus, an erroneous calculation of variance can extend from the underlying databases to the systems built on top of them.

Name	Formula	Accuracy	Passes	Storage	Parallel
Two Pass	$S = \sum_{i=1}^{N}(x_i - \bar{x})^2$, $\bar{x} = \frac{\sum_{x=1}^{N} x_i}{N}$	✓	2	O(1)	✓
Textbook 1 Pass	$S = \sum_{i=1}^{N} x_i^2$ - $\frac{1}{N}(\sum_{i=1}^{N} x_i)^2$	✗	1	O(1)	✓
Shifted 1 Pass	$S = \sum_{i=1}^{N}(x_i - \bar{x})^2 - (\sum_{i=1}^{N}(x_i - \bar{x}))^2/N$	Varies	1	O(1)	✓
Pairwise Updating	$T_{1,m+n} = T_{1,m} + T_{m+1,m+n}$, $S_{1,m+n} = S_{1,m}$ $+S_{m+1,m+n} + \frac{m}{n(m+n)}(\frac{n}{m}T_{1,m} - T_{m+1,m+n})^2$	✓	1	O(ln(N))	✓
Updating-YC	$T_{1,j} = T_{1,j-1} + x_j$ $S_{1,j} = S_{1,j-1} + \frac{1}{j(j-1)}(jx_j - T_{1,j})^2$	✓	1	O(1)	✗
Updating-WWH (Updating)	$M_{1,j} = M_{1,j-1} + \frac{x_j - M_{1,j-1}}{j}$, $S_{1,j} = S_{1,j-1}$ $+(j-1) \times (x_j - M_{1,j-1}) \times (\frac{x_j - M_{1,j-1}}{j})$	✓	1	O(1)	✗
Total Variance	$S = \sum_{i=1}^{groups}\left(n_i(m_i - \bar{x})^2 + (n_i - 1)v_i\right)$	✓	3	Varies	Varies

Table 1: Commonly used Formulas for Variance.

2. VARIOUS VARIANCE FORMULATIONS

Table 1 presents the common variance representations [1]. We use a similar naming convention to that used by Chan et al. [1]. S represents the sum of squares. The sample variance can be given by $\frac{S}{N-1}$, where N is the sample size. x_i is the i^{th} data point. $\bar{x}$ is the sample mean. $M_{m,n}$ is the mean of the data points from indexes m to n (both inclusive). $T_{m,n}$ is the total of the data points from indexes m to n (both inclusive). We have also provided *Total Variance* (derivation in the technical report [6]). In its formula, n_i, m_i, and v_i represent the count, mean, and variance respectively, of the i^{th} group.

Textbook One Pass can be computationally dangerous as the quantities $\sum_{i=1}^{N} x_i^2$ and $\frac{1}{N}(\sum_{i=1}^{N} x_i)^2$ can nearly cancel each other out. The *Pairwise Updating* formula hierarchically combines pairs of variance values and uses $O(log(N))$ storage while reducing the relative errors from $O(N)$ to $O(log(N))$ [1]. *Updating-YC* represents Youngs and Cramer formula [13] and is essentially identical to *Updating Pairwise* when $m = 1$ or $n = 1$. The *Updating-WWH* formula refers to the nearly identical formulas used by Welford et al. [11], West et al. [12], and Hanson et al. [2] and has similar precision as *Updating-YC*. We have used the *Updating-WWH* representation for updates using a single data point, and denote it by *Updating*. Shifting the data by an exact or approximate value of $\bar{x}$ (*Shifted One Pass*) can also result in substantial accuracy gains [1].

2.1 Total Variance

Since this is the first paper to introduce the *Total Variance* representation, we explain its steps in more details below. In the first pass, which is over the tuples, the variance (using one of the other formulas), mean, and count, of individual groups are computed. The second pass, over the groups thus formed, finds the overall mean of the data. In the third pass, over the groups, the overall variance is then found. Since the second and third passes are over the groups obtained as a result of the first pass, and different formulas can be used to compute variance of individual groups in the first pass, complexity of the overall algorithm can vary widely. While second and third passes are highly parallelizable, its overall parallelizability is dependent upon the formula used to find variance of the groups. It is designed for combining variances of different groups and is agnostic to the representation used in the first pass – our implementation uses *Updating*.

Computing mean of individual groups is a well-researched subject with Tian et al. [10] providing a good overview. We use a single pass algorithm to compute mean of individual groups and to combine means of groups as well. To handle a large number of groups, one can look into using an aggregation tree to combine means. The usual technique of mean estimation can be used in case the number of groups is large, at the cost of decreased precision.

There does not appear to be a theoretically ideal group size for *Total Variance*, and we could not determine one experimentally either [6]. In distributed execution, one natural way is to consider data across different nodes as groups. Further, data within a node can be equally partitioned, so that each core works on a single subgroup.

2.2 Properties of Different Representations

While Chan et al. [1] provide an overview of the accuracy, passes, and storage required for most of the formulas given in Table 1 (other than *Total Variance*), their classification as being distributive, and thus the ability to be parallelized, has not been explicitly listed before, which we do. In Table 1, the *Storage* column depicts the extra space needed for computing variance, which is above and beyond that needed to store the data itself.

The accuracy of *Shifted One Pass* depends on that of the mean estimate. *Pairwise Updating* is the only representation providing accurate results while being highly parallelizable and requiring a single pass. Additionally, as we will see in Section 4, the precision of *Total Variance* is slightly better than that of *Updating Pairwise*, which has the best precision amongst all single pass algorithms. As a side note, *Two Pass*, *Total Variance* and *Textbook One Pass* are the only representations that can be represented using a standard SQL query. Note that Table 2.1 of [1] succinctly enumerates the error bounds of different formulations. Further, Kahan summation [5, 10] can help improve their precision.

2.3 Data Conditioning

Data shifting and scaling are immensely useful in improving accuracy of algorithms [4]. For example, shifting the data by its mean is the basis for *Shifted One Pass*. Indeed, Chan et al. [1] demonstrate the usefulness of shifting by an approximate mean computed using a sample of the data by proving that it reduces the bounds of the condition number. Further, techniques such as dividing by the mean or using the log function [4] can be helpful in improving the accuracy. However, along with requiring additional computational resources these techniques can also worsen the accuracy under malicious datasets [1], and need careful user supervision.

2.4 Hybrid Formulae

It is clear that different implementations can be used to find variance of different groups, and combine partial results. Indeed, it has been brought to our attention that a commercial system uses the *Updating-YC* formula to compute variance at individual nodes, and combines them using *Pairwise Updating* formula. *Total Variance* is a hybrid formula as well. This provokes an interesting piece of future work – choosing different representations at different computation steps, based on factors such as numerical precision, data partitioning, time for first result, number of passes permissible (Section 5). This idea is elaborated upon in Section 5.

2.5 Current Recommendation Guidelines

Chan et al. [1] provide detailed recommendation guidelines for different variance formulas. They recommend usage of *Pairwise Updating* for combining variances across multiple processors since it reduces the errors and is massively parallelizable if extra $O(log(N))$ space is available. Further, it is also the safest (least precision loss) algorithm to use within each processor, under the constraint of a single pass.

2.6 Extensibility to Other Measures

Standard deviation, standard error, and coefficient of variation are important statistical measures, and perform variance computation. Thereby, they are also affected by the properties of the underlying representation. Similarly, the properties will also extend to any user-defined measure whose variance can be expressed in a closed form as a function of the variance of one of the measure dimensions. For example, for a user-defined measure given by $a * \texttt{AVG}(agg) + b$, where a and b are constants and agg is a measure dimension, the variance of the measure can be given in closed form as $a^2 * \texttt{VARIANCE}(agg)$.

3. VARIANCE IMPLEMENTATIONS IN MODERN DATABASE SYSTEMS

We looked at the code of multiple open source databases to find their variance representations. We also conjecture about two closed source ones through our experiments.

Database	Formula
PostgreSQL 9.4.4	Textbook One Pass
MySQL 5.7	Updating
Impala 2.1.5	Updating Pairwise
Hive 1.2.1	Updating Pairwise
Spark 1.4.1	Updating Pairwise
SQLite	No Variance Support
System X	Textbook One-pass *(Conjecture)*
System Y	Cannot Conjecture

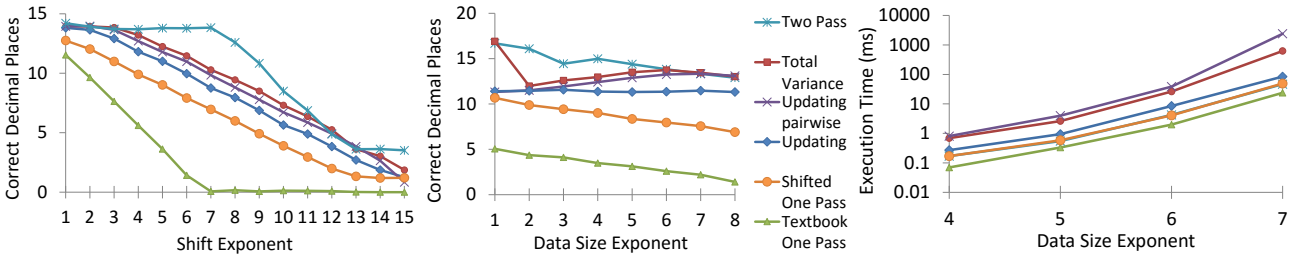
Table 2: Variance Implementations in Databases.

PostgreSQL uses *Textbook One Pass* and is thus susceptible to precision loss. MySQL uses Knuth's modification [8] of Welford's updating formula. Therefore, it can only process a single additional data point, and cannot avail of the possible parallelization. Spark 1.4.1 and Impala 2.1.5, on the other hand, use a modified version of *Updating Pairwise*.

Although the source code for System X is not available, we conjecture that it uses *Textbook One Pass* as its precision behavior was similar to that of PostgreSQL. System Y was found to have the best precision. We hypothesize that it uses higher precision variables, but cannot make any conjecture about the exact representation.

4. EXPERIMENTAL ANALYSIS

Chan et al. [1] have looked at the precision of different algorithms using single precision input. We present the precision results using double precision input. We also evaluate the precision of *Total Variance*. We look at the precision in the variance cal-



(a) With increasing shift exponent, all representations experience precision loss, though some more severely than others.

(b) Precision generally decreases with increasing dataset size.

(c) *Two Pass* provides results faster than others, excepting *Textbook One Pass*, which has the least numerical precision.

Figure 2: *Two Pass* not only has the highest precision, but also requires second lowest execution time.

culation offered by the different databases. We also present the execution times of different algorithms on data sizes up to 100 million tuples. The results are the average over 100 runs. Experiments were performed using Ubuntu 14.04.05 LTS with a 4 core, 2.4 GHz Intel CPU, with 16 GB RAM, and 256 GB SSD storage, using a single execution thread.

Dataset: Although numerous benchmarks exist to evaluate the accuracy of numerical algorithms, they are constrained by their dataset size. For example, the biggest dataset in the NIST StRD [9] benchmark consists of 5000 points. Furthermore, for this dataset, the mean is not significantly larger than the standard deviation ($\mu = 4.5348$, $\sigma = 2.8673$). Therefore, in a similar vein as Tian et al. [10], we created synthetic datasets of different sizes using $Uniform(0, 1)$ (variance being $\frac{1}{12}$). They were shifted by adding values ranging from 10^1 to 10^{15}.

4.1 Impact of Shift

Numerical precision was evaluated using varying additive shifts, over a dataset of size 10000. Group size was set at 10 for *Total Variance*. We present our findings in Figure 2a, where Y-axis represents the number of correct decimal digits (non-fractional part of the result was 0). The results were as expected [1], with *Two Pass* having the best precision, and *Textbook One Pass* the worst.

4.2 Impact of Data Size

Since precision errors typically accumulate, we tried datasets of sizes from 10 to 100 million. The shift was set at 10^5. Figure 2b shows that precision generally worsens with increasing data size. *Two Pass* again outperforms other algorithms. *Textbook One Pass* consistently exhibits the worst precision.

Counter-intuitively, the precision of *Total Variance* and *Updating Pairwise* was found to increase for the data size exponents from 2 to 6. We are unable to conjecture the reason behind this behavior. The precision error for *Updating Pairwise* increases as $O(log(n))$, while that for others (except *Total Variance*) increases as at least $O(n)$ [1], where n is the data size. Therefore, while we can expect the error in *Updating Pairwise* to not increase at the same rate as others, the error decrease is unexpected. In the absence of theoretical error bounds for *Total Variance*, we cannot hypothesize about the possible cause. To ensure there were no irregularities, the experiment was repeated multiple times with similar results.

4.3 Single-Threaded Execution Speed

We also looked at the execution time of different algorithms with increasing data size (Figure 2c). Results with lower data sizes have not been presented due to the computation taking minimal time. Surprisingly, there was no discernible difference in execution time between *Two Pass* and *Shifted One Pass*. Only *Textbook One Pass* took lesser time than *Two Pass*. We attribute the low execution time of *Two Pass* to simplicity of its computation.

4.4 Impact of Shift on Different Databases

We look at variance precision for the different databases under varying additive shifts. We took efforts to ensure different systems have similar data types. 100 points were chosen from a $Uniform(0, 1)$ distribution. Figure 3 shows that precision loss follows a similar pattern in System X and PostgreSQL. Impala and MySQL have a similar error profile as well.

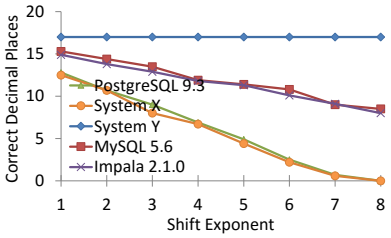


Figure 3: Databases follow expected precision patterns.

4.5 Miscellaneous Experiments

In one of the other experiments, details in [6], we noted that changing group size in *Total Variance* did not have a significant effect on the precision. In another experiment, multi-threaded execution gave

us expected speedups for the parallelizable formulations. Finally, we inspected the mantissa of the two terms that compose *Textbook One Pass* to demonstrate the cause of catastrophic cancellation.

5. CONCLUSION & RECOMMENDATIONS

Precision issues associated with *Textbook One Pass* have been well documented. However, we have seen that databases such as PostgeSQL and likely System X still use it. We recommend from the perspective of safety to discontinue its usage. Though there might be arguments for its continued usage after warning the users in certain scenarios, the arguments against it far outweigh the speedup benefit and its ease of implementation. Although error inherently exists in approximate query processing, numerical precision errors are easy to eliminate and hard to apportion and therefore should be avoided whenever possible. Hence, we recommend to the designers of databases, and statistics and analytics packages, to discontinue its usage. Further, it would be wise for users to perform a sanity check using experiments similar to those given in Section 4.1.

Previous work has recommended *Pairwise Updating* from the perspective of precision, speed, and parallelizability [1]. However, we have seen from our experiments of up to 100 million data points, that the most accurate algorithm, *Two Pass*, takes lesser time than *Updating*, *Updating Pairwise*, and *Total Variance*. Further, it takes around the same amount of time as *Shifted One Pass*, which relies on mean estimation. *Two Pass* is also easy to implement and parallelize. Therefore, in the case that **performing two passes over the data is acceptable, *Two Pass* should be the preferred algorithm**. Determining whether two passes are acceptable, however, is a nuanced decision. When the data fits in memory, performing two passes over the data is clearly acceptable as all representations will incur the identical data read I/O cost. When the data cannot fit in memory, summing up the estimated I/O and computation times can help determine whether *Two Pass* will need the least amount of time, in which case it should be chosen.

In other cases, i.e., whenever *Two Pass* is not estimated to require the least execution time, there does not exist a clear winner, due to different algorithms having different strengths and weaknesses. *Updating* provides faster results at lower precision, compared with *Updating Pairwise*, without needing additional memory. *Updating Pairwise* is parallelizable, whereas *Updating* is not. While *Shifted One Pass* provides quick results, its accuracy is dependent on correctness of the mean estimate. *Total Variance* has good accuracy, although it takes longer to execute, and is dependent on the algorithm used to compute group statistics, while also needing multiple passes. Hence, there does not exist any algorithm that dominates every other algorithm, resulting in there not being a clear choice. We can see that a query planner that devises hybrid formulas, while taking the data distribution, estimated I/O and computation costs, and the overall strengths and weaknesses of different algorithms into consideration, appears to be an important and ideal piece of future work.

6. ACKNOWLEDGMENT

We acknowledge the generous support of U.S. NSF under awards IIS-1422977 and CAREER IIS-1453582. We would also like to thank the reviewer for their highly insightful and helpful comments.

7. REFERENCES

[1] T. Chan et al. Algorithms for Computing the Sample Variance: Analysis and Recommendations. *Am. Stat.*, 1983.
[2] R. J. Hanson. Stably Updating Mean and Standard Deviation of Data. *ACM*, 1975.
[3] J. M. Hellerstein, C. Ré, F. Schoppmann, et al. The MADlib Analytics Library: or MAD Skills, the SQL. *VLDB*, 2012.
[4] N. J. Higham. Accuracy and Stability of Numerical Algorithms. *SIAM*, 2002.
[5] W. Kahan. Further Remarks on Reducing Truncation Errors. *ACM*, 8(1):40, 1965.
[6] N. Kamat and A. Nandi. A Closer Look at Variance Implementations In Modern Database Systems. *Arxiv TR*, 2016.
[7] S. Kandel et al. Profiler: Integrated Statistical Analysis and Visualization for Data Quality Assessment. *AVI*, 2012.
[8] D. E. Knuth. *Art of Computer Programming, Volume 2: Seminumerical Algorithms*. 2014.
[9] J. Rogers, J. Filliben, et al. StRD: Statistical Reference Datasets for Testing the Numerical Accuracy of Statistical Software, 1998.
[10] Y. Tian, S. Tatikonda, and B. Reinwald. Scalable and Numerically Stable Descriptive Statistics in SystemML. *ICDE*, 2012.
[11] B. Welford. Note on a Method for Calculating Corrected Sums of Squares and Products. *Technometrics*, 1962.
[12] D. West. Updating Mean and Variance Estimates: An Improved Method. 1979.
[13] E. A. Youngs et al. Some Results Relevant to Choice of Sum and Sum-of-Product Algorithms. *Technometrics*, 1971.

Towards Visualization Recommendation Systems

Manasi Vartak[1] Silu Huang[2] Tarique Siddiqui[2] Samuel Madden[1] Aditya Parameswaran[2]

[1]MIT
{mvartak, madden}@csail.mit.edu

[2]University of Illinois (UIUC)
{shuang86, tsiddiq2, adityagp}@illinois.edu

ABSTRACT

Data visualization is often used as the first step while performing a variety of analytical tasks. With the advent of large, high-dimensional datasets and significant interest in data science, there is a need for tools that can support rapid visual analysis. In this paper we describe our vision for a new class of visualization systems, namely visualization recommendation systems, that can automatically identify and interactively recommend visualizations relevant to an analytical task. We detail the key requirements and design considerations for a visualization recommendation system. We also identify a number of challenges in realizing this vision and describe some approaches to address them.

1. INTRODUCTION

Data visualization is perhaps the most widely used tool in a data analyst's toolbox, but the state of the art in data visualization still involves manual generation of visualizations through tools like Excel or Tableau. With the rise of interest in data science and the need to derive value from data, analysts increasingly want to use visualization tools to explore data, spot anomalies and correlations, and identify patterns and trends [30, 14]. For these tasks, current tools require substantial manual effort and tedious trial-and-error. In this paper, we describe our vision for a new class of *visualization recommendation* (VISREC) systems that automatically recommend visualizations that highlight patterns or trends of interest, thus enabling fast visual analysis.

Why Now? Despite the widespread use of visualization tools, we believe that we are still in the early stages of data visualization. We draw an analogy to movie recommendations: current visualization tools are akin to a movie catalog; they allow users to select and view the details of any movie in the catalog, and do so repeatedly, until a desired movie is identified. No current tools provide functionality similar to a movie recommendation system which gives users the ability to intelligently traverse the space of movies and identify interesting movies, without getting bogged down by their sheer number or unnecessary details. On similar lines, the goal of VISREC systems is to allow users to easily traverse the space of visualizations and focus only on the ones most relevant to the task. There are two reasons why such visual recommendation tools are more important now than ever before:

Size. While the size of datasets—in terms of number of records and number of attributes— has been rapidly increasing, the amount of human attention and time available to analyze datasets has stayed constant. With larger datasets, users must manually specify and examine a larger number of visualizations and must experiment with more attributes and subsets of data before arriving at visualizations showing patterns of interest.

Varying Skill Levels. Users with varying levels of skill in statistical and programming techniques are now performing data analysis. As a result, there is a need for easy-to-use analysis tools for domain-experts who have limited data analysis expertise. Such tools can perform the heavy-lifting for analyzing correlations and patterns, and surface relevant insights in the form of accessible and intuitive visualizations.

Limitations of Current Tools. Current visualization tools such as Excel and Tableau provide a powerful set of mechanisms to manually specify visualizations. However, as tools to perform sophisticated analyses of high-dimensional datasets, they lack several features:

- *Inadequate navigation to unexplored areas.* Due to the large number of attributes and values taken on by each attribute, exploring all parts of a dataset is challenging with current tools. Often some attributes of the dataset are never visualized, and visualizations on certain portions of the dataset are never generated. This focus on a tiny part of the data is especially problematic if the user is inexperienced or unfamiliar with attributes in the dataset.
- *Insufficient means to specify trends of interest.* Current tools lack the means to specify *what* the analyst is looking for, e.g., perhaps they want to find all products that took a hit in February, or they want to find all attributes on which two products differ. Us-

ing current tools, the analyst must specify each candidate visualization individually and determine if it satisfies the desired criteria.

- *Poor comprehension of context or the "big picture".* Existing tools provide users no context for the visualization they are currently viewing. For instance, for a visualization showing a dip in sales for February, current tools cannot provide information about whether this dip is an anomaly or a similar trend is seen in other products as well. Similarly, the tool cannot indicate that another attribute (e.g. inclement weather) may be correlated with the dip in sales. In current tools, users must generate related visualizations manually and check if correlations or explanations can be identified. There are also no means for users to get a high-level summary of typical trends in the visualizations of a dataset.
- *Limited understanding of user preferences.* Apart from giving users the ability to re-create past visualizations, existing tools do not take past user behavior into account while identifying relevant visualizations. For instance, if the user typically views only a handful of attributes from a dataset, maybe it is worth recommending to this user other attributes that may be correlated or similar to these attributes.

Recent work by us and others has attempted to propose systems that address various aspects of visualization recommendations, e.g. [32, 40, 46, 37]. Commercial products are also beginning to incorporate elements of VISREC into their tools [1, 2]. However, all of these tools are far from being full-featured VISREC systems. This position paper aims to detail the key requirements and design considerations for building a full-feature VISREC system. While we are inspired by traditional product recommendation systems in developing the ideas in this paper, our primary focus will be on aspects that are unique to the VISREC setting. Throughout this paper, we focus on the *systems-oriented* challenges of building a VISREC system. There are many challenging user interface and interaction problems that must be addressed to build an effective VISREC system; these are, however, outside the scope of this vision paper.

We begin by discussing *axes* or *dimensions* that are relevant in making a recommendation (Section 2), the criteria for assessing quality of recommendations (Section 3), and architectural considerations (Sections 4 and 5). We then describe our current work in this area (Section 6) and conclude with a brief discussion of related work (Section 7).

2. RECOMMENDATION AXES

Whether a visualization is *useful* for an analytical task depends on a host of factors. For instance, a visualization showing sales over time may be useful in a sales projection task, while a visualization showing toy breakdown by color may be useful in a product design task. Similarly, a visualization showing a dip in profit may be of interest for a salesperson, while a visualization explaining the upward trend in auto accidents would be of interest to auto-manufacturers. In this section, we outline five factors that we believe must be accounted for while making visualization recommendations: we call these *recommendation axes*.

I. Data Characteristics. In many ways, the goal of a visualization recommender system is to mine the data for interesting values, trends, and patterns to speed up data analysis. These patterns may be then presented to the user at different stages of analysis, e.g. when they first load the dataset, while performing some task, or viewing a particular visualization. There are a number of data characteristics that a VISREC system can consider while making recommendations, e.g.: *a*) *summaries*, e.g., histograms or summary statistics [43], providing an overview of the data distribution; *b*) *correlations*, e.g., Pearson correlation, Chi-squared test [43], providing an understanding of correlated attributes; *c*) *patterns and trends*, e.g., regression [43], association rules, or clustering, providing an understanding of what is "typical" in the dataset and enabling users to contextualize trends; *d*) *advanced statistics*, e.g., tests like ANOVA, Wilcox rank sum [43] aiding in deeper analysis.

II. Intended Task or Insight. Along with data, an important input to a VISREC system is the intent of the user performing analysis: This includes the following aspects: *a*) *style of analysis*: e.g. exploratory, comparative, predictive, or targeted; *b*) *subject of analysis*: subset of data and attributes of interest (e.g., adult males, sweater products, color); *c*) *goal of analysis*: e.g. explanations for a certain behavior (e.g., why is there a spike in february in sales), comparison between subsets of data (e.g., how are staplers doing compared to two years ago), finding unusual or outlier patterns (e.g., are there any toy colors doing "differently"), or finding specific patterns (e.g., chairs with high sales on October 15). While we may be able to obtain explicit task information from the user (e.g. via a drop-down menu or query language of sorts), we may also infer intent through user actions. Finally, if we have information about the user's assumptions or biases regarding the data or task, the VISREC system can also provide recommendations to counter these biases.

III. Semantics and Domain Knowledge. A large amount of semantic information is associated with any dataset—what data is stored, what information does each attribute provide, how are the attributes related, how does this dataset relate to others, etc. This semantic information determines, in part, whether a visualization is "interesting" or "unusual". For instance, if a user is analyz-

ing a dip in profits, semantics would indicate that visualizations showing attributes such as sales, revenue, cost of goods sold, number of items sold would be useful. An even more significant factor—and much harder to capture—is domain knowledge. The user possesses unique or domain-specific knowledge that guides the search for attributes, trends and patterns. For example, a recommendation system that only considers data and task may recommend a visualization showing that the OB-GYN hospital unit has a disproportionately high percentage of female patients. A person with minimal domain knowledge would note that the trend shown in this visualization is obvious and therefore the visualization is unhelpful. Domain knowledge can include typical behavior of specific attributes or subsets of data (e.g., sales always goes up around christmas time, or electronics sales is always greater than stapler sales), or relationships between groups of attributes, (e.g., sales and profits are always proportional). It can also include external factors not in the dataset, e.g., an earthquake may have affected hard disk drive production.

IV. Visual Ease of Understanding. A dimension that is completely unique to visualization recommendation is what we call *visual ease of understanding*. This dimension ensures that data has been displayed in the most intuitive way for easy understanding. Work such as [27, 28] proposes techniques to choose visual encodings, while related work in information visualization includes a variety of techniques to visualize data with varying dimensionality and data types [26, 15, 23, 20].

V. User Preferences and Competencies. Multiple users analyzing the same dataset may have attributes of common interest, while the same user analyzing different datasets may prefer specific visualization types. Similarly, certain views of a particular dataset may be most intuitive or most relevant during a particular phase of analysis, leading most users to prefer these visualizations. A VISREC system also needs to account for the varying levels of visual literacy and statistical ability of the user. There is a large body of work on extracting user preferences (e.g., [16, 29]) as well as cognitive modeling (e.g., [13]), techniques from which can be adapted for VISREC. Furthermore, these techniques can be combined with assessments of visual and statistical literacy (e.g. [11, 9]) to tailor recommendations for each user.

Traditional recommendation systems focus mainly on User Preference and to some extent on Intended Task; however, the other axes enumerated above are tailored to VISREC systems.

3. RECOMMENDATION CRITERIA

The previous section discussed factors that contribute to the utility of visualizations. In this section, we discuss criteria to measure quality of visualization recommendations. We find that some criteria are similar to traditional product recommendations (e.g. relevance) while others are unique to VISREC (e.g. non-obviousness) or are re-interpretations of existing criteria (e.g. surprise).

- *Relevance*: This metric measures whether the recommendation is useful to the user in performing the particular analytic task. As discussed in the previous section, many factors such as data, task, semantics etc., play a role in determining relevance.
- *Surprise*: This metric measures the novelty or unexpected-ness of a recommendation. For product recommendations, this metric prefers items the user didn't explicitly ask for but are relevant. In VISREC, this corresponds to visualizations that show something *out of the ordinary*. For example, a dip in sales of staplers may not be interesting by itself but when juxtaposed with the booming sales of other stationery items, it becomes interesting.
- *Non-obviousness*: This metric is specific to VISREC. Non-obviousness measures whether the recommendation is expected given semantics and domain knowledge (as opposed to surprise which is defined with respect to data). For instance, the OB-GYN example discussed previously was surprising from a statistical point of view, but was, in fact, obvious to a user with minimal domain knowledge.

Since we expect the recommender system to recommend multiple visualizations, the quality of the *visualization set* is as important as the quality of individual visualizations. We note that the order of recommendations is also important in this regard and we expect order to be determined by relevance, along with measures related to coherence over time and visualization set quality.

- *Diversity.* This metric measures how different are the individual visualizations in the recommended collection. Diversity may be measured with respect to attributes, visualization types, different statistics, visual encodings, etc. A more subtle notion of diversity would capture the "informativeness" of a collection of visualizations relative to each other—the *conditional utility* of a visualization given others.
- *Coverage.* This metric measures how much of the space of potential visualizations and of the dataset is covered by recommendations. While users particularly value coverage during exploration, during analysis, users seek to understand how *thorough* are the recommendations shown to them. For instance, the user would like to understand whether the system examined ten or ten thousand visualizations (and similarly whether the system examined 10% or 100% of the data) before recommending visualizations.

4. ADAPTING RECSYS TECHNIQUES

The task of building a VISREC system brings up a

natural question: recommender systems is a rich area of research; how much of existing work can we reuse? Our goal in this section is to broadly identify problems in VISREC that can be solved using existing techniques, and those that require new techniques.

Existing methods for product recommendation broadly fall into three categories [7, 4]: (i) *content-based filtering* that predicts user preferences based on item attributes; (ii) *collaborative-filtering* that uses historical ratings to determine user or item similarity; and (iii) *knowledge-based filtering* that uses explicit knowledge models to make recommendations. Collaborative filtering is probably the most popular technique currently used in recommender systems (e.g. at Amazon [24]). However, collaborative filtering (as well as content-based filtering) assumes that there is historical rating data available for a large number of items. As a result, it suffers from the traditional *cold start problems* when historical ratings are sparse. Knowledge-based filtering [39], in contrast, does not depend on history and therefore, does not suffer from cold start problems.

VISREC differs from product recommendations in a few key areas that impact the techniques that can be used for recommendation. In VISREC, *new* datasets are being analyzed by *new* users constantly. Furthermore, each new task on a dataset can produce an entirely new (and large) set of visualizations from which the system must recommend, i.e., not only is the universe of items large, it is generated on-the-fly. Consequently, VISREC systems almost never have sufficient historical ratings to inform accurate collaborative or content-based filtering. Visualization recommenders must therefore rely on on-the-fly, knowledge-based filtering. This is not to say that techniques such as collaborative filtering cannot be used to transfer learning across datasets; it means that while such techniques can aid in recommendations, the heavy lifting must be performed by knowledge-based filtering.

Applying knowledge-based techniques to VISREC brings up several challenges that have not been addressed in the recommender systems literature: (i) Models must be developed for capturing the effect of each recommendation axis (Section 2) on visualization utility; (ii) Knowledge models must be such that they can perform online processing with interactive latencies. For example, along the data axis, several of the existing data mining techniques from Section 2 are optimized for offline processing. As a result, these techniques must be adapted to work in an online setting with small latencies; (iii) Efficient ranking techniques and ensemble methods must be developed for combining large number of models along individual axes, and multiple axes.

VISREC systems also suffer from a problem not faced by product recommendations, namely one of *false discoveries*. When making automated visualization recommendations, a VISREC system evaluates many tens or hundreds of visualizations before making recommendations. Since a visualization can (roughly) be thought of as performing a hypothesis test, chances of finding spurious patterns increase with increasing number of visualizations. As a result, a VISREC system must account for potential false discoveries in recommendations using techniques such as Bonferroni [8] or FDR [6] correction.

Thus, while there is a rich body of work in recommender systems, the unique challenges of VISREC require the development of new, and in many cases, online and efficient recommendation techniques. In the next section, we discuss the implications of the unique VISREC requirements on system design and techniques that can be used to meet these requirements.

5. ARCHITECTURAL CONSIDERATIONS

Making visualization recommendations, particularly based on data, is computationally expensive. Therefore, we find that the most important consideration in making real-time recommendations is the *data processing engine*. While traditional disk-resident databases can accomodate large datasets, they cannot provide the interactive speeds necessary for visualization recommendation. As a result, a VISREC system must take advantage of main-memory using techniques such as operating on samples, pre-materializing views and using efficient indexes. We now elaborate on some of these strategies.

Pre-computation. Many real-world recommender systems perform complex and expensive computation (e.g. computations on the item-user matrix in collaborative filtering [24]) in an offline phase. The results of this computation are then used to make fast predictions during the online phase. Since VISREC systems must employ knowledge-based filtering and the set of potential visualization is not known upfront, opportunities to perform complex computations offline may be limited. However, some types of pre-computation, drawn from the database systems literature, can be employed. For example, *data cubes* can be used to precompute and store aggregate views for visualization (e.g. Nanocubes [25]). Along the lines of data cubes, a visualization recommender can also perform offline computation of various statistics and correlations that can inform subsequent explorations and construction of visualizations. Specialized indexes tailored to access patterns unique to visualization recommendations (e.g. [22]) can be used to further speed up online data access. Finally, traditional *caching* approaches that have been used with great success both on the client-side as well as the server-side can be used to further reduce recommendation latency.

Online Computation. As discussed previously, visual recommenders are in the unique position of having to produce the space of potential recommendations on-the-

fly. As a result, a significant part of the computations must happen online. To avoid latencies in the hours, online computation must perform aggressive optimizations while evaluating visualizations. Some of the techniques include: (i) *parallelism*: faced with a large space of potential visualizations that must be evaluated, we can evaluate visualizations in parallel to produce a large speedup; (ii) *multi-query optimization*: the computations used to produce candidate visualizations are often very similar; they perform similar operations on the same or closely related datasets. Consequently, multi-query optimizations techniques [34, 31] can be used to intelligently group queries and share computation; (iii) *pruning*: while the above techniques can increase the speed of execution, they do not reduce the search space of visualizations. Although hundreds of visualizations are possible for a given dataset, only a small fraction of the visualizations are actually useful. As a result, a significant fraction of computational resources are wasted on low-utility visualizations. Pruning techniques (e.g. confidence-interval pruning [43], bandit resource allocations [41]) can be used to discard low-utility views with minimal computation; (iv) *better algorithms*: finally, there are opportunities to develop better and faster algorithms to compute statistical properties.

Approximate Computation. Approximate query processing [3, 5] has been shown to have tremendous promise in reducing query latencies on large datasets. Techniques based on different sampling strategies (e.g. stratified sampling, coresets [10], importance sampling [38]) can be used to further speed up computation, especially because users may be satisfied with imperfect results: both imperfect visualizations [22] and imperfect recommendations of visualizations. Sampling brings with it a few challenges. For a given computation, we must choose the right type of sample (based on size, technique etc). Additionally, for a given sampling strategy, we must provide users with measures of confidence in the results (e.g. confidence intervals). These measures of quality are particularly important in data analysis since they inform users how much they can trust certain results. Finally, while sampling may be useful to compute many statistical properties, certain properties such as outliers cannot be answered correctly with a sample.

6. OUR PRIOR AND CURRENT WORK

We now briefly describe some of our efforts towards building VISREC systems and future work.

SEEDB. As a first attempt towards building a full-fledged VISREC system, we built SEEDB [40] (Figure 1). SEEDB is designed as a mixed-initiative [18] system that provides users the ability to both manually construct visualizations (component "B"), and receive recommendations (component "D"). In judging utility, SEEDB deems a visualization to be interesting if it displays a large deviation from a reference. For example, a visualization of sales of staplers over time may be interesting if a reference (e.g., sales of all products) is showing an opposite trend. Our user study comparing SEEDB with and without recommendations demonstrates that users are *3X more likely to find recommended visualizations useful as compared to manually generated visualizations*.

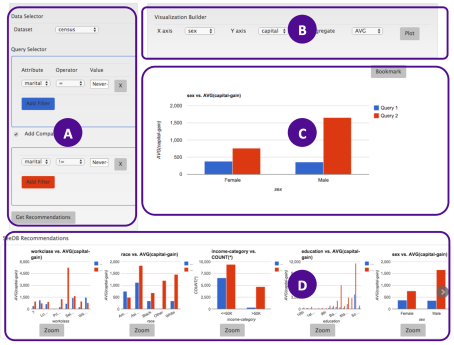


Figure 1: SEEDB *Frontend: (A) query builder, (B): visualization builder, (C): visualization pane, (D) recommendations pane*

zenvisage. Our new visualization recommendation tool is called Zenvisage—meaning to view (data) effortlessly [37, 36]. The goal of zenvisage (Figure 2) is to quickly identify interesting patterns or trends from large datasets via one of two mechanisms: a simple drag-and-drop based interactive interface with query sketching capabilities (e.g., find a visualization where there is a spike at a certain point simply by drawing the desired visualization on a canvas—Box 4 shows the result for the drawing in Box 3, while Box 2 shows other typical visualizations for context) and a visual data exploration language called ZQL for more complex requests (Box 5).

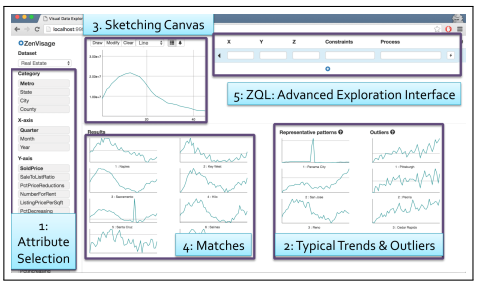


Figure 2: zenvisage *Frontend*

7. RELATED WORK

Partial Automation of Visualizations. Tools such as Spotfire and Tableau have recently started providing some features for automatically choosing visualizations for a data set [1, 2]; however these features are restricted to a set of aesthetic rules-of-thumb that guide visualization. Profiler [19] detects anomalies in data and provides some visualization recommendation functionality.

VizDeck [21] allows users to select from visualizations presented on a dashboard.

VISREC **Systems.** Work such as [28, 27] focuses on recommending visual encodings for a user-defined set of attributes, thus addressing the *visual ease of understanding* axis. Similar to SEEDB, [35, 45, 44] use different statistical properties of the data to recommend visualizations. [12] monitors user behavior to mine for intent and provides recommendations, while [42] uses task information and semantic web ontologies. Most recently, the Voyager system [46] has been proposed to provide visualization recommendations for exploratory search.

Finding patterns and trends. The data mining and machine learning community has developed a large swath of statistical analysis tools such as Knime, RapidMiner, SAS, and SPSS, and programming libraries [17, 33] for doing complex analytics tasks such as classification, clustering, and dimensionality reduction. While many of these tools and libraries can be employed in VISREC systems to mine for patterns in data, only expert users with detailed knowledge of algorithmic details and parameterizations can use these tools effectively.

8. CONCLUSION

With increasing interest in data science and large numbers of high-dimensional datasets, there is a need for easy-to-use, powerful visualization recommendation tools to support visual analysis. While we are in the early days of VISREC systems, we believe the directions outlined in this paper, as well as the analogies to and differences with traditional recommendation systems can lead to interesting, challenging, and impactful problems for the database research community.

9. REFERENCES

[1] Spotfire, http://www.tibco.com/company/news/releases/2015/tibco-announces-recommendations-for-spotfire-cloud. [Online; accessed 17-Aug-2015].

[2] Tableau showme. [Online; accessed 17-Aug-2015].

[3] S. Acharya et al. The aqua approximate query answering system. SIGMOD '99, pages 574–576, New York, NY, USA, 1999. ACM.

[4] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *TKDE*, 17(6):734–749, 2005.

[5] S. Agarwal et al. Blinkdb: Queries with bounded errors and bounded response times on very large data. EuroSys '13, 2013.

[6] Y. Benjamini and D. Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.

[7] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109–132, 2013.

[8] C. E. Bonferroni. Teoria statistica delle classi e calcolo delle probabilità. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, 8:3–62, 1936.

[9] J. Boy, R. A. Rensink, E. Bertini, and J.-D. Fekete. A principled way of assessing visualization literacy. *IEEE transactions on visualization and computer graphics*, 20(12):1963–1972, 2014.

[10] D. Feldman, M. Schmidt, and C. Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. SODA '13, pages 1434–1453, 2013.

[11] J. B. Garfield. Assessing statistical reasoning. *Statistics Education Research Journal*, 2(1):22–38, 2003.

[12] D. Gotz and Z. Wen. Behavior-driven visualization recommendation. IUI '09, pages 315–324, New York, NY, USA, 2009. ACM.

[13] T. M. Green, W. Ribarsky, and B. Fisher. Building and applying a human cognition model for visual analytics. *Information Visualization*, 8(1):1–13, Jan. 2009.

[14] P. Hanrahan. Analytic database technologies for a new kind of user: the data enthusiast. In *SIGMOD Conference*, pages 577–578, 2012.

[15] I. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *TVCG*, 6(1):24–43, 2000.

[16] S. Holland, M. Ester, and W. Kießling. Preference mining: A novel approach on mining user preferences for personalized applications. In *PKDD 2003*, pages 204–216. Springer, 2003.

[17] G. Holmes, A. Donkin, and I. H. Witten. Weka: A machine learning workbench. In *Conf. on Intelligent Information Systems '94*, pages 357–361. IEEE, 1994.

[18] E. Horvitz. Principles of mixed-initiative user interfaces. CHI'99, pages 159–166. ACM, 1999.

[19] S. Kandel et al. Profiler: integrated statistical analysis and visualization for data quality assessment. In *AVI*, pages 547–554, 2012.

[20] D. Keim et al. Information visualization and visual data mining. *TVCG*, 8(1):1–8, 2002.

[21] A. Key, B. Howe, D. Perry, and C. Aragon. Vizdeck: Self-organizing dashboards for visual analytics. SIGMOD '12, pages 681–684, 2012.

[22] A. Kim, E. Blais, A. Parameswaran, P. Indyk, S. Madden, and R. Rubinfeld. Rapid sampling for visualizations with ordering guarantees. *Proc. VLDB Endow.*, 8(5):521–532, Jan. 2015.

[23] M. Kreuseler, N. Lopez, and H. Schumann. A scalable framework for information visualization. INFOVIS '00, pages 27–, Washington, DC, USA, 2000. IEEE Computer Society.

[24] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[25] L. Lins, J. T. Klosowski, and C. Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.

[26] S. Liu, W. Cui, Y. Wu, and M. Liu. A survey on information visualization: recent advances and challenges. *The Visual Computer*, 30(12):1373–1393, 2014.

[27] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. Graph.*, 5(2):110–141, Apr. 1986.

[28] J. D. Mackinlay et al. Show me: Automatic presentation for visual analysis. *IEEE Trans. Vis. Comput. Graph.*, 13(6):1137–1144, 2007.

[29] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. *Commun. ACM*, 43(8):142–151, Aug. 2000.

[30] K. Morton et al. Support the data enthusiast: Challenges for next-generation data-analysis systems. *PVLDB*, 7(6):453–456, 2014.

[31] C. H. Papadimitriou and M. Yannakakis. Multiobjective query optimization. In P. Buneman, editor, *PODS*. ACM, 2001.

[32] A. Parameswaran, N. Polyzotis, and H. Garcia-Molina. Seedb: Visualizing database queries efficiently. *PVLDB*, 7(4), 2013.

[33] Pedregosa et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[34] T. K. Sellis. Multiple-query optimization. *ACM Trans. Database Syst.*, 13(1):23–52, Mar. 1988.

[35] J. Seo and B. Shneiderman. A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization*, 4(2):96–113, 2005.

[36] T. Siddiqui et al. Fast-forwarding to desired visualizations with zenvisage. In *CIDR*, 2017.

[37] T. Siddiqui, A. Kim, J. Lee, K. Karahalios, and A. Parameswaran. Effortless visual data exploration with zenvisage: An expressive and interactive visual analytics system. In *PVLDB*, 2016.

[38] S. T. Tokdar and R. E. Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):54–60, 2010.

[39] S. Trewin. Knowledge-based recommender systems. *Encyclopedia of Library and Information Science: Volume 69-Supplement 32*, page 180, 2000.

[40] M. Vartak, S. Rahman, S. Madden, A. Parameswaran, and N. Polyzotis. Seedb: efficient data-driven visualization recommendations to support visual analytics. *VLDB*, 8(13):2182–2193, 2015.

[41] J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. In *ECML*, pages 437–448, 2005.

[42] M. Voigt et al. Context-aware recommendation of visualization components. In *eKNOW'12*, pages 101–109, 2012.

[43] L. Wasserman. *All of Statistics*. Springer, 2003.

[44] L. Wilkinson, A. Anand, and R. L. Grossman. Graph-theoretic scagnostics. In *INFOVIS*, volume 5, page 21, 2005.

[45] G. Wills and L. Wilkinson. Autovis: automatic visualization. *Information Visualization*, 9(1):47–69, 2010.

[46] K. Wongsuphasawat et al. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans. Visualization & Comp. Graphics*, 2015.

Rick Hull Speaks Out on Asking the New Question

Marianne Winslett and Vanessa Braganholo

Rick Hull

http://researcher.watson.ibm.com/researcher/view.php?person=us-hull

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 Rick Hull, who is a researcher at IBM. Before that, he was a professor at the University of Southern California for many years. He also managed a research group at Bell Labs, where he was a Bell Labs Fellow. Rick is an ACM Fellow and a coauthor of the classic database theory book Foundations of Databases. His Ph.D. is from Berkeley. So, Rick, welcome!

Thank you!

What could database theory researchers do to increase their impact on the world?

Well, that's a big question to get started with. I think the challenges of data management today are really expanding from what they were 20 and 30 years ago. Data is so pervasive in our lives today, from social media to ecommerce, and things like using weather data for smarter farming, it goes on and on. We've been discussing this in the PODS community broadly – how to study the fundamental issues raised by new kinds of data and new uses of it. I think the main thing is to consider what are the problems today and then what are the techniques that could be brought to bear. So rather than using mathematical logic as the starting point for most explorations, it is time to more fully embrace additional frameworks, including probability, statistics, etc.

But people have been telling me that there is still plenty of room for logic.

Oh, absolutely. I'm not saying that the logic is no longer needed. I am just saying that we need to expand the full range of techniques that we can bring to bear, the kinds of models that we might look for. The logic is still a very important foundational element. I know, for example, some of the basic techniques in the statistical approach start with counting all of the true first order models of a given theory.

> ***[…] with that paper, we were asking a new question. This was part of Seymour Ginsburg's mantra: always ask the new question. It was an unusual question, a non-standard question.***

And is that happening now? Already happened? Or is that the next step?

I think there are very positive signs in the past 2-3 years. Just at this conference, we had the Big Uncertain Data workshop, which was a very deliberate attempt to bring together people from the database theory side (especially work on probabilistic databases) with people from the machine learning side.

What about the fact that there aren't that many commercial probabilistic databases?

It's funny you should ask that. I am also concerned that in the old days the data management activities, including those that the database theory community was contributing to, was a growing industry. It was really a growing, very important field. Now, the big growth seems to be in the big data, the analytics areas. The machine learning community has been producing artifacts that are useful to that industry, and I'm hoping that over time the database community broadly and the database theory community can contribute into that area as well. There is also important work in other areas of data management, for example multimodal data, incomplete data, data-centric workflow.

You stayed at Bell Labs for a very long time. What did you enjoy about your time there?

It was about 12 years. Especially at the beginning it was a very exciting place. The company was growing. It was owned by Lucent at the time. It was my first experience in an industrial research lab. It allowed me to do both -- to continue with theoretical research and advanced research, but it also gave me a chance to be talking with customers, wrestling with the challenges of how you take ideas and bring them into reality. How do you bring some kind of value or capability or something that will be used by the average person on the street?

So that would be true of all industrial labs?

Each lab is different. You know, I'm speaking of my experiences with Bell Labs and then IBM Research. Both I think offer this breadth of opportunity.

And you haven't gotten back to academia, so it seems like you're voting with your feet that you really like that connection to the customer.

You never know what might happen. I think the opportunity to work in a larger group as well as continuing with individual contributions is something that I enjoy.

About the Alice book, that's the nickname for your database theory book. What was it like to write that book?

That was a lot of fun. To understand the context, in the early 1980's I was at the University of Southern California, and Victor Vianu and Serge Abiteboul were there as well, and Seymour Ginsburg was the mentor for all three of us. This was right at the beginning of database theory and so we felt like we were on the ground floor. We were learning some of the early results. We were under Seymour's guidance, starting to build up our own body of results. So we wrote the book about ten years after being together at the University of Southern California.

When we were writing the book, it was a point where the foundations of database theory, at least that the first real era of database theory was, I feel, coming to a kind of closure. Well, not a closure, but there was a feeling of completeness to what had been studied. So the book was at a perfect time to capture and encapsulate that body of work and hopefully provide the foundation for the next generations of work.

Well, that's a good point because a reviewer on Amazon says that although it was published in 1995, it quote "it is still the gold standard… especially in consideration of the fact that nothing much has changed in database technology in the past 30 years or so". Do you agree with that?

No, I wouldn't agree. I mean it's nice to think that it's a gold standard for something. Maybe it is potentially a gold standard for that period of the database theory and the basic logical framework that was set up. At the same time, since then there's been a tremendous body of work in database theory to understand XML as a major area, connections with XML, automata, constraints, etc., further advances in constraint databases, description logics and data, and of course now as we go into the big data period. So there has been a lot of advancements.

Students often choose one of your papers from the class reading list because it's shorter than the other options and then they just knock themselves out trying to understand the paper. For example, many researchers have been influenced by your PODS 1984 paper about when two databases are equivalent[1]. What's so hard about that topic?

(Laughing) The information capacity paper and its follow-ons… Yeah, with that paper, of course, we were asking a new question. This was part of Seymour Ginsburg's mantra: always ask the new question. It was an unusual question, a non-standard question. I think that's why conceptually it has been hard for people to think along that line. Secondly, we had four levels of relative information capacity and each level called for some different techniques. So I think that maybe that also makes it harder than some other papers where there's kind of one core technique and then it's just played out.

I think that whole direction was really important. Nowadays, when query answers have some sort of statistical aspect to them, in my group, we believe that if two databases are equivalent you should get the same answer no matter which one you run your algorithm over, which is kind of a radical notion. But we really believe that should be true and if you can't talk about what's equivalent you can't argue that you should be getting the same answer. So that's a nice example paper I think to pick out. It was back in '84 but it's still important 30 years later.

What are artifact-centric business process models?

That's a big question. Maybe the last four or five years of my research work was in that area of what we call business artifacts, or using business artifacts to support business process models. So business artifacts are really a great opportunity for the database community and others to study a combination of data and process. Kind of married as equal partners. You see traditional business model management is focused on the process side; flowcharts or maybe it's based on Petri nets. A lot of the research in that area has focused just on the process and it has left the data as a second-class citizen. But in reality, the business process is really touching data right and left. So with business artifacts, the core model really focuses on what we call key business-relevant conceptual entities. Key conceptual entities that progress through a business during normal activity. So as an example, we talk about the FedEx package delivery. Not the package, but the package delivery, and think of it in terms of when the package was first received by FedEx, the transportation, the delivery, the sign-off, also the billing, how that goes. With a business artifact model for that, you have an

[1] Richard Hull: Relative Information Capacity of Simple Relational Database Schemata. PODS 1984: 97-109.

information model that holds the relevant data that may be obtained as that entity progresses through its process. Also, you track the life cycle model, the possible ways or possible activities that might happen to the package.

I would argue that the database community does the reverse of what you're saying the business process people do. We care enormously about the data and we don't care at all about the process. So, is this where we're supposed to meet, in the middle?

Definitely, it is one opportunity for the two sides to meet. What we found is that the business artifact perspective gives a very strong intuitively natural top-down view of the business processes. Typically there are 3-7 business artifact types that you need to model a given process. They can be cross-cutting. So if your business process is cutting across multiple different silos of your business, often the business artifacts span multiple silos and give that top-down end-to-end view that's lacking in so many other cases. Let me say that there has been a body of research. There are probably now 20 or 30 active researchers in the database community, the AI community, and the business process management community that have been working on this model and its marriage in areas from efficiency and distributed systems all the way up to verification, really spanning the gamut from systems to theory. Actually, the theory side was discussed in the Diego Calvanese's PODS keynote talk last year (2013).

Is it getting traction in the business world?

I would say absolutely. The work on business artifacts, which started at IBM Research in 2003, was, in fact, the motivation for me to go to IBM Research. By about 2011, people came to realize that business artifacts were actually very much a formalization of the case management approach to business process modeling and now the work we did at IBM Research has provided the foundation for the OMG standard on case management and also for the IBM case management product.

Good! Great, it's great to see that happening. Speaking of IBM, what is next for IBM? They've stayed alive so long so there must be something new just around the corner.

Well, they are very deliberate at the corporate management level of steering the ship, always thinking about what is next. You know we saw recently in the past several years this initiative around this smarter planet, smarter cities, smarter education, and smarter healthcare. Recently, the big topic area both in terms of activity and in terms of the marketing is on cognitive computing, as we call it. It's building on the success of the Watson deep question and answer system. People may recall it had been featured on the Jeopardy television game series maybe a couple of years ago. There, it played against two Jeopardy champions and it demonstrated the ability of a machine to have processed just tons of both structured data and unstructured data, to be able to reason about it, and to be able to, in this case, formulate questions based on all of that learning. Now there is a division of IBM that is focused on Watson and applications of the Watson technology. The research division has also been reorganized a bit and there's now quite a large activity around what is the future of computing given that it can take advantage of this unprecedented amount of processing power and in particular, processing of the unstructured data.

I enjoy being with people and being able to have a diverse set of challenges in front of me.

What kind of applications are we likely to see coming out of that?

I think we're already seeing some of them and they'll just get stronger. I mean one area that even before it was being labeled cognitive computing is in the smarter healthcare area. For example, they are training the Watson system to be able to take the medical board examination. After medical school, the doctors take some kind of exam. Well, they're training Watson to be able to take that exam and also to explain the reasons behind whatever answers they are giving. There's also an activity where IBM is partnering with Sloan Kettering, the Cancer Care Center, to help with cancer diagnosis. That's one area.

Another area that has kind of personally been intriguing for me is in smarter education: enabling students to experience personalized learning pathways. This means they can be working on material that is delivered over a tablet and through the use of analytics, through deep analysis of text material, of the problems, etc., you can really deliver to the student the next best module for that student, his learning style, what he

knows, where he is trying to go, in terms of his academics or his career. We're also seeing it in more business settings, financial analysis, for example, advising people on where to invest their money, how to build their investment portfolio as they move through their lives.

That could really be beneficial for math in the K-12 era education where kids think it has no application to whatever they're interested in and that is so completely false. So if they're interested in construction, there is tons of math in construction, if they were interested in baking or sewing, there's tons of math in that and if the problems they were given whether it's trigonometry, algebra or whatever were tailored to their interest… same formulas but expressed differently then they would see how it connects to the real world, but we don't do a good job of that. Or if we do it's about trains traveling in different speeds and different directions and where they will collide or whatever.

> ***[…] take the time to really work a problem, think about the problem, try to go deeper, try to ask the next provocative question.***

Those are really good examples actually because the idea is that you can start to tailor many aspects of what is being taught to the interest of the kid and also to their aspirations.

Okay, let's see. Have you found it more satisfying to do research or to manage research?

You know, I think it's really the mix that is most exciting for me. I like to have my hands into something concrete, even if it's a mathematical abstraction, it is in a way concrete for me and you're working puzzles with it or you're trying to figure out an algorithm that's going to work or prove that something is correct. At the same time, I enjoy being with people and being able to have a diverse set of challenges in front of me. So in some of my most enjoyable periods both at Bell Labs and IBM Research, that's been the experience: I've been managing, and I've been collaborating with outside universities, maybe working on a project with a customer, but also working out some little theorem to help solidify the foundations of the concept.

Are you still collaborating with Serge and Victor?

Not as much as I had. With Serge, it's been a while. With Victor, he has been involved with the business artifact work. In fact, with Victor and Alin Deutsch, we've started a line on verification of business artifact properties and so that's been quite enjoyable.

So the take home message for all the students reading is that whoever your colleagues are on your grad school days you may still be working with them many years later, so those relationships can really last a long time.

You are perhaps the coolest person in the database research community. Where did you get your cool?

That is a funny question. I wonder where you get those questions… I'm glad at least you think I might be cool. You know, I'll give you three possible factors. So, one is when I was young, my father was into camping, the outdoors and we would go camping or canoeing on the river and be outdoors for two or three days at a time and I think that kind of experience of being in nature (this was long before cell phones, but it was also a time away from a lot of distractions) it's kind of an interesting mindset to carry with me. Of course, being an undergraduate at the University of California Santa Barbara, on the beach, that was a big one. A third one is, Europeans have a certain cool and coming back to Serge and Victor I spent a lot of time with them and then I was able to visit Serge in France for several summers working at Inria and Victor was typically there. It may be the exposure to Serge and Victor that really put it over the top.

A shared cool. I did not make up that question myself. I got it from one of your colleagues and same holds for the next question, which is: tell us about your hair. Can you show us your hair?

(Rick shows his hair.)

Look at all that hair! Ok, so what's the story behind that?

Well, I haven't thought about that recently. I guess one answer is that in research we're always thinking about something new, the new question, and the new technique. At the same time, I still have my roots. I like the old as well and I grew my hair out in the late 60s. It was kind of part of the peace movement back then, and somehow I never cut it off.

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

I think one word would be networking, get to know your fellow researchers better, try to collaborate, that's just such a wealth of stimulation. I think another thing is the power of the human mind. What I found at least is that if I really live with a problem, work with a problem, make time to think deeply, challenge myself, that's when the mind can really go to the deeper insights. So I would recommend: take the time to really work a problem, think about the problem, try to go deeper, try to ask the next provocative question. I think it's so easy in this day and age to get distracted by the next email, the next phone call, the next meeting or whatever. So, you know, trust your mind and dig deep.

They're under so much pressure to get that next paper out so that they can get their first job or whatever that it can be hard to do.

Yes, I agree, and I remember actually that the paper on relative information capacity was published in PODS then it was time to write the full journal version, then there was a deadline, and I realized there was a bug (you know, a minor bug). So I spent some long nights wrestling that to the ground and on the one hand that is fine, I did have the chance to think about it deeply and met the deadline, but it was unfortunate to feel under that pressure.

That means that if they choose that PODS paper for their class because it's shorter, that they should if they really understand it, they should find a bug in there.

I wouldn't want to go on record saying that they should find a bug there, but I think a challenging exercise may be for a very motivated student would be, "What's the difference between the journal version and the conference version?"

Ok, very good!

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

Well, in addition to doing some more of that deep thinking that I don't seem to get a chance for anymore, I would say reading other people's work, I find that I just don't have enough time to read other stuff and to work with other people's material and really have a strong understanding of it.

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

In my case, I grew up through a math major and my Ph.D. was in math, although formal language theory, so from a very mathematical perspective. As time went on, I became more involved with a system side as well as the theory side. I think a change would have been to spend more time on the real computer science side of things, programming, programming languages, abstraction. These are principles and foundations of our field that it would be nice if I understood them a bit better.

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

Well, I think it's the artifact-centric work broadly, but if you wanted me to just pick out one paper, I think what I would pick out is the paper we had in the ICSOC 2009 conference[2]. That's the International Conference on Service Oriented Computing, and the paper is on artifact-centric hubs.

Thank you very much for talking to me today, Rick

Thank you!

[2] Richard Hull, Nanjangud C. Narendra, Anil Nigam: Facilitating Workflow Interoperation Using Artifact-Centric Hubs. ICSOC/ServiceWave 2009: 1-18.

Stratos Idreos Speaks Out on Database Craking

Marianne Winslett and Vanessa Braganholo

Stratos Idreos
http://stratos.seas.harvard.edu/

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 Phoenix, site of the 2012 SIGMOD and PODS conference. I have here with me Stratos Idreos, who is the 2011 recipient of the SIGMOD Jim Gray Dissertation Award for his thesis entitled Database Cracking: Towards Auto-tuning Database Kernels. Stratos's advisors were Stefan Manegold and Martin Kersten, and his PhD is from the University of Amsterdam. Stratos is currently a tenure-track researcher at the Dutch National Research Institute for Mathematics and Computer Science (CWI)[1]. So, Stratos, welcome!

[1] Stratos is currently an Assistant Professor at the Harvard University.

Tell me what your thesis is about.

My thesis introduced the concept of database cracking. The main idea is that every query that comes through the database system will be used as an advice on how data should be stored on disk and on memory. So basically, the system creates indexes incrementally and on the fly during query processing. Normally, database systems would need enough idle time and workload knowledge to create indexes. Now, with database cracking, indexes are created automatically without you having to worry about all these preparations.

So when you say incrementally, do you mean you create the whole index while queries are running, or you create part of a traditional index?

Exactly, that is a very good question. So, creating the whole index while queries are running, this is online indexing. There's a couple of works by Surajit Chaudhuri and Nico Bruno at Microsoft that do online indexing. In addition, there is the work by Alkis Plyzotis which came at about the same time. Our work is about incremental indexing. We create only parts of the index during query processing. So, let me give you a more representative example. I create part of indexes within select operators, for example. So if you have a select operator of a query that says give me everything from this table where values of attribute A are between 20 and 30, then we would take the column of attribute A and we would split it in three parts: from 0-20, from 20-30, from 30-whatever. Then you have introduced range partitioning by splitting the table in three pieces, and that is enough information to improve future queries.

So, then if you want to use an index in the future, you start by checking if the data is covered…

If this partition exists, you can use it, you can explore it, you can refine it even more. So these little pieces that you create, they become smaller and smaller with every other query. And every other query introduces more and more partitioning, which means more knowledge about how data is laid on disk, and then you can explore it. And by pieces becoming smaller, performance becomes better.

So when you say smaller, what do you mean smaller? I would think they would become larger over time.

Larger in terms of how many they are, smaller in terms of how many tuples they have inside.

I see, so they get divided into finer grained courses.

Yes, and if you think about it, at every range select operator you have to touch at most two pieces, because you only have to check the boundaries of the range, and that's at most two pieces. And by pieces becoming smaller, you have to analyze less tuples with every other query.

I stopped taking rejected [papers] reviews very religiously.

Doesn't this make query optimization harder?

Yes and no. No because you have chosen to always use indexes. So every query will use indexes, there is no decision about that. You blindly go and use the same database plan with every query. You could think about it as if you always created clustered indexes, basically. It's not secondary indexes.

I see. When you say it is not secondary indexes, do you mean you rearrange the data on disc to match…?

Yes. We rearrange the actual data. We create copies of the data, and we rearrange these copies. So at the first time that you query, for example, if you want to select over attribute A, we create a copy of this column and we start rearranging this column. And then every query that wants to select on A will go directly there, and won't touch the base data anymore.

So, later queries on attribute B make another copy with B, and not the whole tuple, just the tuple ID is there?

Exactly. It's attribute B and the tuple ID.

But what about updates?

Updates, that's a tricky business. But what we do with updates is that we defer them. When updates come, we just keep them aside. And we only merge them when a relevant query comes. So let's go back to the previous example: attribute A, between 0-10, 20-30 and so on. Then if another query comes and says, okay, I want values between 20-25, if and only if there are pending updates within this range, then we merge them on the fly during query processing. So the select operator would not only fine grain the partitioning information, but it would also merge updates.

And if I understood correctly, if the query is over attributes A and B, you'll have a little index that's just for that range of A and B?

Yes.

Hmmm, very interesting. So how did you show that's better than the alternatives?

First of all, let me clarify that this kind of ideas, this kind of research, is applicable for exploratory dynamic workloads. So in the case that you know exactly what you are looking for, you have enough idle time to prepare your indexes for that, you should not be using database cracking, there's no sense. But in the case where you don't have enough knowledge about the workload, and you don't have enough idle time to prepare, then is when you should be using database cracking. So what we always do in our experiments is we compare database cracking with a plain, non-indexing approach, where you have to scan your data, and we always compare it with the perfect indexes, which in the case of column-stores is when you have basically sorted arrays. And that is the equivalent of offline indexing, in this case, because, in order to sort an array, you need time to do the sorting, and you need to know that this array is useful when sorted. What we typically see in these examples is that the performance of database cracking starts with the first query being almost as expensive as a scan (just a little bit more expensive), and then it quickly improves performance, and after a few queries, it reaches the optimum performance of an index. But the offline indexing approach takes typically 10 times more in order to create the index. So if you don't have idle time, you first have to pay this 10 times more overhead.

Ok, that's interesting.

Take good care of yourself, (…) do some physical activities.

Maybe a more representative example would be queries of TPC-H, for example. In order to create the optimal indexes for the columns in TPC-H in this particular machine that we used and everything, we needed about 3 hours. With cracking we could answer all queries, getting to optimum performance in a matter of seconds, basically.

And then, so if you just did cracking with no previous knowledge of the TPC-H workload, versus if you had created the perfect indexes beforehand, what's the difference in performance at run time of the transactions?

So, we havent't studied extensively the performance of transactions, we typically do only analytical read queries. But the difference compared to the optimal index is basically zero. You reach the optimal performance. You don't expect optimal performance as of query one. In the case of TPC-H (it is actually a good case for us because the workload is skewed), you reach optimal performance in a matter of 5-10 queries, and the good point is that as of query number two, you are way below the performance of a no index approach. But then as of query 5-10, you reach the optimal performance of a perfectly tuned database. Now, if you devise micro-benchmarks, where you have random workloads basically, this optimum performance comes after thousands of queries, not after 5 or 6.

Do you have any words of advice for today's PhD students?

I would have many. My main lesson that I try to remember now after my PhD is that I stopped taking rejected [papers] reviews very religiously. So one big mistake that I think that I made over the years is that sometimes I got reject reviews (and I got many of them), and then I thought that "okay, I should react very very seriously based on this review", and sometimes I ended up basically just destroying papers and making them very dense, just because I was trying to put every little detail in there. So I think this would be good, although we should take rejects very seriously, and put comments to use, but maybe we should also take a step back, and think about it again.

Is there another piece of advice you would like to share?

Yeah, I'll say that it's not only about research, we should also take good care of yourself as well, so maybe sometimes take a step back, don't do so much research, do some physical activities.

Very good! Thank you very much for talking with us today!

Thank you!

Report on the First International Workshop on Reproducible Open Science

Paolo Manghi,
ISTI - Consiglio Nazionale delle Ricerche, Italy
paolo.manghi@isti.cnr.it

Jochen Schirrwagen
Bielefeld University Library, Germany
jochen.schirrwagen@uni-bielefeld.de

Oscar Corcho
Universidad Politecnica de Madrid, Spain
ocorcho@fi.upm.es

Amir Aryani
Australian National Data Service, Australia
amir.aryani@ands.org.au

1. INTRODUCTION

In the last decade, information and communication technology (ICT) advances have deeply affected the scientific process, which increasingly produces and relies on digital research products, such as publications, datasets, experiments, websites, software, blogs, etc. Accordingly, scientific communication has started mutating in order to adapt its mission (and business models) to such new scientific paradigms and benefit from the unprecedented Open Science opportunities that may arise from them: *reproducibility*, i.e., the ability of repeating a digital experiment and reusing its constituent products; and *transparent evaluation*, i.e., the ability of (*i*) effectively evaluating scientific experiments by means of reproducibility and (*ii*) assigning fine--grained scientific reward, based on effective authorship across the overall scientific process. Scientists, research institutions, and funders are pushing for innovative Open Science scholarly communication workflows (i.e., submission, peer-review, access, reuse, citation, and scientific reward), marrying a holistic approach where publishing includes in principle any digital product resulting from a research activity that is relevant to the evaluation and reproducibility of the activity or part of it. Defining, taking up, and supporting Open Science publishing workflows become urgent challenges, to be addressed by ICT solutions capable of fostering and driving radical changes in the way science is developed and disseminated.

The goal of the first International Workshop on Reproducible Open Science[1] was to provide a forum for constructively exploring foundational, organizational and systemic challenges towards the implementation of Open Science publishing principles. Its mission was to contribute to the actual picture of the state of the art approaches and solutions that researchers and practitioners active in these fields have investigated and realized: library and information scientists working on the identification of new publication paradigms, ICT scientists involved in the definition of new technical solutions to these issues, and scientists/researchers who actually demand tools and practices for transparent evaluation and reproducibility of science. The workshop has brought together skills and experiences focusing on the definition and establishment of the next generation scientific communication ecosystem, where scientists can publish research results (including the scientific article, the data, the methods, and any alternative product that may be relevant to the conducted research) in order to enable reproducibility (effective reuse and decrease of cost of science) and rely on novel scientific reward practices.

RepScience2016 has been organized in conjunction with the 20th edition of the International Conference on Theory and Practice of Digital Libraries[2]. Proceedings of the workshop are under publication as a special issue of the Open Access journal D-Lib Magazine[3].

2. WORKSHOP CONTRIBUTIONS

Each submitted contribution was peer-reviewed by three of the seventeen members of the Program Committee and ten were accepted, out of which three reported the results of RDA Working Groups. The workshop structure comprised two invited speak-

[1] *RepScience2016's web site* `http://repscience2016.research-infrastructures.eu`

[2] *TPDL2016's web site*, `http://www.tpdl2016.org`

[3] *D-Lib Magazine* `http://www.dlib.org`

ers and four sessions. In the following we shall first report on the invited talks, then group the presentations according to general topics they covered.

2.1 Invited talks

The workshop had two invited talks respectively covering the foundational and more theoretical aspects of reproducibility and the real case of reproducibility challenges currently studied at CERN's scientific information services.

Carole Goble in the talk entitled "What is Reproducibility? The R* Brouhaha" depicted the challenges of reproducibility in computational science by drawing an analogy between laboratory microscope experiments and e-infrastructure "datascope" experiments. The issues are similar, with "experiments" constituted by materials (e.g., datasets, parameters, algorithm seeds) and methods (e.g., techniques, algorithms, specifications of steps, models); and "set up" constituted by instruments (e.g., codes, services, scripts, libraries, workflows) and laboratory (e.g., e-infrastructure, system software, integrative platforms, engines). The definition of reproducibility is a non-trivial one, and weaker or stronger forms may be defined, depending on the intent of the researchers and the capabilities of the underlying e-infrastructure. Examples are "rerun", i.e., variations on experiment set up to enable robustness, "repeat", i.e., same experiment same laboratory to defend one's thesis, "replicate", i.e., same experiment, same set up, different lab to enable certification, "reproduce", i.e., variations of the same experiment, on different set ups and laboratories, and "re-use", i.e., different experiment using material, methods of the experiment. Overall, reproducibility has a cost, both social/cultural and technological, whose dimensions are portability/preservation, (packaging, containers), access (standards, licensing, PIDs), robustness/versioning (change, variation sensitivity, discrepancy handling), and description (standards, common metadata, ontologies). Finally, the Research Object framework[4] was presented as a possible solution to address these issues.

Sunje Dallmeier-Tiessen in the talk entitled "Enabling reproducible research: community practices, service needs and first lessons learnt" has presented CERN's challenges and vision to provide scientists with an e-infrastructure supporting Open Data principles and analysis preservation and reproducibility. CERN scientific information services serve today a variety of research communities each featuring different but also overlapping requirements on these matters. An important objective is to advocate and establish a culture of open sharing of data and algorithms, to get rid of the current "fear of losing control", by leveraging on the potential and concrete benefits and providing adequate technological support. To this aim, CERN Data Services are developing tools enabling linking data with data (subsets, versions, dynamic data), contributors (who, when, where), articles, institutions, and funders. On the side of analysis preservation and reproducibility, CERN is devising tools for supporting scientists at developing science, since its earliest phases, in such a way that the results will be reproducible, according to a model: save, retrieve, review/compare, and repeat/reproduce. So far, the challenges identified are those of (*i*) granularity, complexity, and dependencies of data and software, (*ii*) identification of solutions for data and software publishing, linking, and citation, and (*iii*) the demanding amount of manual work needed to make experimental material reproducible.

[4]Research Object, `researchobject.org`

2.2 Presentation of contributions

The following sections summarizes the workshop presentations[5] organized according to four themes: *Towards an enabling infrastructure*, *Models and languages*, *Systems*, and *Real-world experiences*.

Towards an enabling infrastructure. Enhancing the current scholarly communication infrastructure and workflows to support Open Science and reproducibility opens up to different visions and questions.

Stephan Pröll presented the paper "Enabling Reproducibility for Small and Large Scale Research Data Sets" where the authors have investigated the problem of guaranteeing transparent citation of subsets of data (e.g., results of queries) from dynamic data sources (e.g., databases). The most intuitive solutions (e.g., digital copies) raise a number of challenges (e.g., time, storage, DOIs/handles) which the framework identified by the authors helps at elegantly describe and solve at different extents, driven by a cost analysis.

Paolo Manghi presented "The Scholix Framework for Interoperability in Data-Literature Information Exchange". Scholix[6] is a framework for enabling exchange of information relative to links between scientific products across sources in the scholarly communication domain. The framework defines an information model and exchange formats for such links to transparently move across independent plat-

[5] *Workshop presentation slides* `http://repscience2016.research-infrastructures.eu/index.php?d=sessions`

[6] *Scholix Framework* `http://www.scholix.org`

forms, scientific domains, and stakeholders (e.g., repositories for data and publications, publishers, research infrastructures, libraries).

Models and languages. Enabling reproducibility requires ways to encode the elements composing the experiments, i.e., scientific products, and possibly the actions, i.e., the steps constituting the experiment.

On this respect, Markus Konkol reported on the paper "Opening the Publication Process with Executable Research Compendia". The authors propose the *executable research compendium* (ERC) as a means to publish and access computational research. ERC provides a new standardisable packaging mechanism which combines data, software, text, and a user interface description. As similar approaches to research objects or packages, ERC aims at satisfying needs of authors, readers, publishers, curators, and preservationists, in terms of scientific evaluation, reward, visibility, and reproducibility.

Markus Stocker presented his work "From Data to Machine Readable Information Aggregated in Research Objects". Data interpretation is an important process in scientific workflows, where scientists are called to interpret data (often) collected using large-scale environmental monitoring infrastructures to gain information about the monitored environment. Such information is typically represented to suit human consumption, while the authors propose an encoding into machine readable information objects that builds on the Research Object framework.

Paolo Manghi described the results of "FLARE: a flexible workflow language for research e-infrastructures", where the authors defined FLARE, a workflow language for the specification (and execution) of a scientific process in highly-heterogeneous environments, i.e., e-infrastructures whose workflow are partly manual and automated. FLARE lays in between business process modelling languages, i.e., high-level specifications of a reasoning, protocol, or procedure, and workflow execution languages, i.e., machine-readable specifications of computational steps executable by dedicated engines. FLARE tools allows the creation and sharing of hybrid workflows and their execution, via "web wizards" guiding the scientists through the manual and automated execution of the individual steps.

Systems. This session focused on new generation repositories required for depositing, sharing, and accessing the products of science, be them datasets or methods, in such a way they can be properly reused and experiments reproduced.

Vidya Ayer presented the article "Conquaire: Towards an architecture supporting continuous quality control to ensure reproducibility of research" reporting on the preliminary results of the project Conquaire, aiming at delivering an infrastructure based on subject-specific components offering functionalities for data deposition and versioning, enabling automated and discipline-specific quality checks over the data. The system architecture relies on a DCVS system for storing data and on continuous integration principles to ensure data quality.

Sheeba Samuel presented "Towards Reproducibility of Microscopy Experiments". The authors have realized an information system (based on the existing OMERO system[7]) that supports scientists in the domain of microscopy techniques at following a rigorous methodology for collecting documentation and research data to the level necessary to reproduce scientific experiments. Although the approach addresses the specific requirements of an interdisciplinary team of scientists from experimental biology to store, manage, and reproduce the workflow of their research experiments, it can also be extended to the requirements of other scientific communities.

Real-world experiences. Many scientists are today using tools to (*i*) publish their research products in order to achieve degrees of reproducibility or (*ii*) search out for the products needed to reproduce experiments. Such real-world experiences make a fertile ground where to identify common requirements for an open and reproducible science.

Jingbo Wang reported on two experiences in different contexts. The first was titled "Graph connections made by RD-Switchboard using NCIs metadata", where she demoed connectivity graphs linking datasets, papers, authors, and grants, built using the Research Data Switchboard[8] using NCIs metadata database[9]. By means of such graphs, the NCI database was enriched with critical but missing information in the network of researchers and article-dataset links, thereby enhancing the search capabilities of the system and enabling fit-for-purpose (e.g., research topic/context-driven) dataset discovery. The second experience was titled "Supporting Data Reproducibility at NCI Using the Provenance Capture System". The National Computational Infrastructure (NCI) of Australia has realised

[7] *Open Microscopy Environment Remote Objects* `http://www.openmicroscopy.org/site/products/omero`
[8] *The Research Data Switchboard* `http://www.RD-Switchboard.org`
[9] *The Australian National Computational Infrastructure* `https://nci.org.au/`

a system supporting researchers at modelling their workflows, including those that have been used to create data extracts (e.g., queries to databases), using a standards-based provenance representation. This information, combined with access to the original dataset and other related information systems, allows data extracts to be easily regenerated to support experiment reproducibility, limiting preservation of data extracts to very specific cases.

Finally, Jan H. Höffler presented the experience of "ReplicationWiki: Improving Transparency in Social Sciences Research", an attempt to compensate in the field of empirical social sciences the lack of scientific reward regarding authoring of "replicable studies" and authors of the required "replicable products". ReplicationWiki[10] today documents 2500 empirical studies and the relative replication products found in the literature, so far mainly in economics. The wiki is populated by professors and students in economics across several participating institutions, with the aim of establishing the culture of open reproducible science, as well as facilitating academic teaching, and setting incentives for replicability and replication.

3. WORKSHOP DISCUSSION

The concluding brainstorming session brought up two main relevant considerations and future issues with respect to open and reproducible science.

Experimental context (or set up). Computational reproducibility spins around the concept of experimental context (or set up), namely the components required to execute an experiment by applying computation over data, or to evaluate the quality of digital products, be them data or computation. The experimental context must be shared by scientists, to ensure a common ground of evaluation and execution, thereby enabling transparent evaluation, comparisons, and reproducibility. The ability of sharing an experimental context to the largest and agnostic audience entails a trade off between "ability to adopt" and "portability of experiments". On the one hand scientists can assume to share experiments and relative products based on common and agreed on experimental context and methodologies. This will allow them to compile minimal descriptions on how products are to be combined to reproduce and experiment, hence making it easy for scientists to adopt reproducibility practices, but in turn making the experiments effectively reproducible solely to scientists aware of the underlying "commons". On the other hand, scientists can instead share the components and the relative descriptions, so as to ensure the experiments can be reproduced beyond the borders of their community. The extent of details for such descriptions may ensure a broader coverage but in general hinders the adoption by scientists, e.g., tedious metadata provision or evolution of external software components. Identifying the optimal balance is not trivial and also depends on the maturity of a common experimental context, typically research e-infrastructures, and its components.

Roadmap to reproducibility. Open Science has become more and more relevant and appealing for all stakeholders of scientific communication, i.e., research and academic organizations, researchers, publishers, libraries, and funders. The first results are visible with the strong shift of funders and organizations towards mandates for Open Access to publications, which started less than a decade ago, and more recently to ensuring research data sharing (i.e., deposition, description, and preservation), e.g., Data Pilot of the European Commission. Countries, libraries, and research communities (research infrastructures) are moving towards economy of scale solutions for the storage of data, and several initiatives are suggesting methodologies and cost/sustainability analyses that may facilitate this highly expensive process. As reproducibility is gaining relevance and appeal among the very same stakeholders, it is reasonable to expect that similar initiatives will face the problem of how a research community or a library can initiate supporting reproducibility of science for a community or multiple communities starting from a given e-infrastructural setting.

4. ACKNOWLEDGMENTS

Special thanks are also due to the members of the program committee[11] whose research experience largely contributed in making this workshop an attractive venue and constructive experience. Moreover, our sincere gratitude goes to all participants, invited speakers and authors, whose enthusiasm and vision constitute the soul of this workshop and future research in the field. This event was funded by the RDA Europe project[12] under the H2020 program of the European Commission (grant agreement: 653194).

[10] *ReplicationWiki* `http://replication.uni-goettingen.de/`

[11] For *RepScience Program Committee* follow `http://repscience2016.research-infrastructures.eu`

[12] *RDA Europe*, `http://europe.rd-alliance.org`