SIGMOD Officers, Committees, and Awardees

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

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

Secretary/Treasurer
Fatma Ozcan
Google
Sunnyvale
California
USA
fozcan <at> google.com

SIGMOD Executive Committee:
Juliana Freire (Chair), Ihab Francis Ilyas (Vice-Chair), Fatma Ozcan (Treasurer), K. Selçuk Candan, Rada Chirkova, Chris Jermaine, Wang-Chiew Tan, AnHai Doan, Leonid Libkin, and Curtis Dyreson

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

SIGMOD Information Director:
Curtis Dyreson, Utah State University

Associate Information Directors:
Huiping Cao, Georgia Koutrika, Wim Martens, and Sourav S Bhowmick

SIGMOD Record Editor-in-Chief:
Rada Chirkova, NC State University

SIGMOD Record Associate Editors:
Azza Abouzied, Lyublena Antova, Marcelo Arenas, Renata Borovica-Gajic, Vanessa Braganholo, Aaron J. Elmore, Wim Martens, Kyriakos Mouratidis, Dan Olteanu, Tamer Özsu, Divesh Srivastava, Pınar Tözün, Immanuel Trummer, Yannis Velegrakis, Marianne Winslett, and Jun Yang

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

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

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

SIGMOD Awards Committee:
M. Tamer Özsu (Chair), Stefano Ceri, Yanlei Diao, Volker Markl, Renee Miller, and Sunita Sarawagi

Jim Gray Doctoral Dissertation Award Committee:
Pınar Tözün (co-Chair), Viktor Leis (co-Chair), Peter Bailis, Alexandra Meliou, Bailu Ding, Vanessa Braganholo, Immanuel Trummer, and Joy Arulraj

SIGMOD Edgar F. Codd Innovations Award

For innovative and highly significant contributions of enduring value to the development, understanding, or use of database systems and databases. Recipients of the award are the following:

Michael Stonebraker (1992)
David DeWitt (1995)
Serge Abiteboul (1998)
Rudolf Bayer (2001)
Ronald Fagin (2004)
Jennifer Widom (2007)
Umeshwar Dayal (2010)
Stefano Ceri (2013)
Gerhard Weikum (2016)
Anastasia Ailamaki (2019)
Jim Gray (1993)
C. Mohan (1996)
Hector Garcia-Molina (1999)
Patricia Selinger (2002)
Michael Carey (2005)
Moshe Y. Vardi (2008)
Surajit Chaudhuri (2011)
Martin Kersten (2014)
Goetz Graefe (2017)
Beng Chin Ooi (2020)
Philip Bernstein (1994)
David Maier (1997)
Rakesh Agrawal (2000)
Don Chamberlin (2003)
Jeffrey D. Ullman (2006)
Masaru Kitsuregawa (2009)
Bruce Lindsay (2012)
Laura Haas (2015)
Raghu Ramakrishnan (2018)

SIGMOD Systems Award

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

Michael Stonebraker and Lawrence Rowe (2015); Martin Kersten (2016); Richard Hipp (2017); Jeff Hammerbacher, Ashish Thusoo, Joydeep Sen Sarma; Christopher Olston, Benjamin Reed, and Utkarsh Srivastava (2018); Xiaofeng Bao, Charlie Bell, Murali Brahmadesam, James Corey, Neal Fachan, Raju Gulabani, Anurag Gupta, Kamal Gupta, James Hamilton, Andy Jassy, Tengiz Kharatishvili, Sailesh Krishnamurthy, Yan Leshinsky, Lon Lundgren, Pradeep Madhavarapu, Sandor Maurice, Grant McAlister, Sam McKelvie, Raman Mittal, Debanjan Saha, Swami Sivasubramanian, Stefano Stefani, and Alex Verbitski (2019); Don Anderson, Keith Bostic, Alan Bram, Grg Burd, Michael Cahill, Ron Cohen, Alex Gorrod, George Feinberg, Mark Hayes, Charles Lamb, Linda Lee, Susan LoVerso, John Merrells, Mike Olson, Carol Sandstrom, Steve Sarette, David Schacter, David Segleau, Mario Seltzer, and Mike Ubell (2020)

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)
Ahmed Elmagarmid (2019)
Stratos Idreos (2020)
Dennis Shasha (2020)
Gio Wiederhold (1995)
Avi Silberschatz (1997)
Michael Carey (2000)
Richard Snodgrass (2002)
Hongjun Lu (2005)
Klaus R. Dittrich (2008)
Gerhard Weikum (2011)
Kyu-Young Whang (2014)
Yannis E. Ioannidis (2017)
Philipe Bonnet (2020)
Stefan Manegold (2020)
Yahiko Kambayashi (1995)
Won Kim (1998)
Laura Haas (2000)
Michael Ley (2003)
Tamer Özsu (2006)
Beng Chin Ooi (2009)
Marianne Winslett (2012)
Curtis Dyreson (2015)
Z. Meral Özsoyoğlu (2018)
Juliana Freire (2020)
Ioana Manolescu (2020)

SIGMOD Jim Gray Doctoral Dissertation Award

SIGMOD has established the annual SIGMOD Jim Gray Doctoral Dissertation Award to *recognize excellent research by doctoral candidates in the database field.* Recipients of the award are the following:

- **2006** *Winner*: Gerome Miklau. *Honorable Mentions*: Marcelo Arenas and Yanlei Diao
- **2007** *Winner*: Boon Thau Loo. *Honorable Mentions*: Xifeng Yan and Martin Theobald
- **2008** *Winner*: Ariel Fuxman. *Honorable Mentions*: Cong Yu and Nilesh Dalvi
- **2009** *Winner*: Daniel Abadi. *Honorable Mentions*: Bee-Chung Chen and Ashwin Machanavajjhala
- **2010** *Winner:* Christopher Ré. *Honorable Mentions*: Soumyadeb Mitra and Fabian Suchanek
- **2011** *Winner*: Stratos Idreos. *Honorable Mentions*: Todd Green and Karl Schnaitterz

- **2012** *Winner*: Ryan Johnson. *Honorable Mention*: Bogdan Alexe
- **2013** *Winner*: Sudipto Das, *Honorable Mention*: Herodotos Herodotou and Wenchao Zhou
- **2014** *Winners*: Aditya Parameswaran and Andy Pavlo.
- **2015** *Winner*: Alexander Thomson. *Honorable Mentions*: Marina Drosou and Karthik Ramachandra
- **2016** *Winner*: Paris Koutris. *Honorable Mentions*: Pinar Tozun and Alvin Cheung
- **2017** *Winner*: Peter Bailis. *Honorable Mention*: Immanuel Trummer
- **2018** *Winner*: Viktor Leis. *Honorable Mention*: Luis Galárraga and Yongjoo Park
- **2019** *Winner*: Joy Arulraj. *Honorable Mention*: Bas Ketsman
- **2020** *Winner:* Jose Faleiro. *Honorable Mention:* Silu Huang

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

[Last updated: June 30, 2021]

Editor's Notes

Welcome to the June 2021 issue of the ACM SIGMOD Record!

This issue starts with the Database Principles column featuring an article by Doleschal, Kimelfeld, and Martens. The article starts with an observation that information extraction can be viewed as a process in which relational operators are applied to relations extracted from text. Given this conceptual connection between relational databases and information extraction, the main question addressed in the article is whether principles of relational data management can be leveraged in text analytics. The authors examine this and related questions through the lens of document spanners, a framework that has been established to build the foundations of relational principles in information extraction. Formally in this framework, a document is a string d over a finite alphabet, a span of d represents a substring of d by its start and end positions, and a spanner is a function that maps every document d into a relation over the spans of d. The article presents an overview of recent research efforts in this area, including query evaluation and complexity analysis, aggregate queries, provenance, and parallel correctness. The authors also provide pointers to work in related areas and describe open directions of further research.

The Surveys column features an article by Doulkeridis, Vlachou, Pelekis, and Theodoridis that provides an overview of state-of-the-art big-data processing frameworks and techniques for data of interest in mobility analytics, that is, for spatial, spatio-temporal, and trajectory data. The need for mobility analytics gives rise to research and practical challenges in many domains, including urban contexts, air-traffic management, and marine applications. The authors survey prototype systems and frameworks that represent key building blocks for applications that involve mobility analytics. The article presents overviews of storage techniques and of spatial and spatio-temporal parallel processing frameworks. The presentation of individual systems is coupled with explanations of prominent underlying methods for partitioning, indexing, and query processing, thus serving as a guide for further research in the area. The authors conclude with a discussion of challenging open problems in the field of mobility analytics.

It is a pleasure to introduce two new SIGMOD Record columns, Advice to Mid-Career Researchers and DBrainstorming. The former column opens with an introduction by the column editor Tamer Özsu. The first article featured in the column is by Patrick Valduriez; it discusses challenges and presents experience-based hints for researchers who have successfully completed the junior stage of their career and are looking at the next step. The author invites the reader to examine why they want to do (or keep doing) research, describes the challenge of shifting from being "woodcutter" to becoming "forester," examines approaches to coming up with research projects and to cultivating collaborations, and shares many other ideas and insights with mid-career researchers. As part of the bigger picture, the article will also be of interest to mentors of mid-career researchers, to junior faculty, as well as to graduate students and to those undergraduate students who are thinking about research-oriented careers.

The goal of the second new SIGMOD Record column, DBrainstorming, is to discuss new and potentially controversial ideas that might be of interest and potentially of benefit to the research community. The column series opens with an article by Manos Athanassoulis on transparent data transformations; the article explores what would happen if there were a way to decouple physical data layout from data-access performance. The ideas explored in the article culminate in calls for collaboration between the data-management community and researchers in other disciplines, including programming languages and software engineering.

The Distinguished Profiles column features two articles. The first article is an interview with Sourav Bhowmick, professor at Nanyang Technological University in Singapore (NTU), ACM Distinguished Member, recipient of the VLDB Service Award and of a Lecturer of the Year Award from NTU, and a member of the initiative on Diversity and Inclusion in Database Conference Venues. Sourav's Ph.D. is from NTU. In this interview, he talks about the areas of research in which he has been working, including the unusual and fascinating area of human-data interaction. Sourav explains the importance of detecting conflicts of interest in database-conference venues, which can be viewed as a data-cleaning problem, and shares his thoughts and experience in addressing major challenges in this area. He speaks about contributions of his own visual art to humanitarian causes, shares thoughts on parallels between creating art and creating science, and shares ideas of interest to fledgling and mid-career database researchers. The second interview in the column is with Viktor Leis, who won the 2018 ACM SIGMOD Jim Gray Dissertation Award for his thesis entitled Query Processing and optimization in Modern Database Systems. Viktor is now at the University of Erlangen-Nuremberg; his Ph.D. is from the Technical University of Munich. In the interview, Viktor talks about his thesis, how it has advanced the state of the art, and what impact it has had in industry. He reminisces on his Ph.D. time and provides advice for graduate students.

The Reports column features an article by Dosso, Ferilli, Manghi, Poggi, Serra, and Silvello. The authors report on the outcomes of the Italian Research Conference on Digital Libraries (IRCDL), which took place online in February 2021, with 145 participants in attendance. The focus of the conference this year was on bridging the field of research and information science with the related field of digital libraries. The article summarizes the keynote by Prof. Carole Goble and overviews the papers presented in six thematic sessions, as well as the panel organized to discuss the future of Digital Libraries. The conference materials are available online; in addition, the authors point the readers to the next IRCDL conference, which is to be held in Italy in 2022.

On behalf of the SIGMOD Record Editorial board, I hope that you enjoy reading the June 2021 issue of the SIGMOD Record!

Your submissions to the SIGMOD Record are welcome via the submission site:
https://mc.manuscriptcentral.com/sigmodrecord

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

Rada Chirkova

June 2021

Past SIGMOD Record Editors:

Yanlei Diao (2014-2019)	Ioana Manolescu (2009-2013)	Alexandros Labrinidis (2007–2009)
Mario Nascimento (2005–2007)	Ling Liu (2000–2004)	Michael Franklin (1996–2000)
Jennifer Widom (1995–1996)	Arie Segev (1989–1995)	Margaret H. Dunham (1986–1988)
Jon D. Clark (1984–1985)	Thomas J. Cook (1981–1983)	Douglas S. Kerr (1976-1978)
Randall Rustin (1974-1975)	Daniel O'Connell (1971–1973)	Harrison R. Morse (1969)

Database Principles and Challenges in Text Analysis

Johannes Doleschal
University of Bayreuth &
Hasselt University
johannes.doleschal@uni-bayreuth.de

Benny Kimelfeld
Technion, Israel
bennyk@cs.technion.ac.il

Wim Martens
University of Bayreuth
wim.martens@uni-bayreuth.de

ABSTRACT

A common conceptual view of text analysis is that of a two-step process, where we first extract relations from text documents and then apply a relational query over the result. Hence, text analysis shares technical challenges with, and can draw ideas from, relational databases. A framework that formally instantiates this connection is that of the document spanners. In this article, we review recent advances in various research efforts that adapt fundamental database concepts to text analysis through the lens of document spanners. Among others, we discuss aspects of query evaluation, aggregate queries, provenance, and distributed query planning.

1. INTRODUCTION

Different tools and paradigms have been developed over the past decades to facilitate the challenge of extracting structured information from text—a task generally referred to as *Information Extraction* (IE). Common textual sources include natural language from a variety of sources such as scientific publications, customer input and social media, as well as machine-generated activity logs. Instantiations of IE are central components in text analytics and include tasks such as segmentation, named-entity recognition, relation extraction, and coreference resolution [55]. Rules and rule systems have consistently been key components in such paradigms, yet their roles have varied and evolved over time. Systems such as Xlog [59] and IBM SystemT [13] use IE rules for materializing relations inside *relational query languages*. Machine-learning classifiers and probabilistic graphical models (e.g., Conditional Random Fields) use rules for *feature generation* [38, 62]. Rules serve as *weak constraints* in Markov Logic Networks [51] and in the DeepDive system [60]. Rules are also used for generating *noisy training data* ("labeling functions") in the state-of-the-art Snorkel system [54].

Even though there is a fundamental difference in the structure of the underlying data, there is a tight connection between IE rules and relational databases: both provide machinery for manipulating base relations, either given explicitly (relational databases) or extracted from the text (IE). In the latter case, the base relations are typically constructed via generic extractors implemented in a variety of ways, from regular expressions (e.g., dictionary lookups) to machine-learned networks. We refer to these extractors as *primitive extractors*; hence, we view IE as a process where relational operators are applied to the relations extracted via primitive extractors.

Given the conceptual connection between relational databases and IE, can we leverage the principles of relational data management, as established over decades of research and practice, in the world of IE (and text analytics in general)? Particular questions include the following.

- *What is the expressive power of extraction languages?* What is the contribution of the relational operators to the expressiveness of the language of the primitive extractors? Does the relational component add power or just facilitates query formulation?
- *What is the computational complexity of evaluating IE programs?* How does it depend on the query and the textual data (combined/data complexity)? What guarantees can be made by an algorithm for streaming out many answers (enumeration complexity)? Can we understand their fine-grained complexity as we do for database algorithms [20]? Can we evaluate *aggregate queries* efficiently without materializing the aggregated tuples?
- *How do we approach query planning for IE?* Can we come up with useful plans that *parallelize* the task at hand among many independent computational units? Can we analyze the query to infer such independence as done in the context of *parallel-correctness* in databases [5]?
- *How can we leverage and efficiently manage the* provenance *accumulated in the process of extracting information from text?* Can we use machinery that is based on firm mathematical foundations such as the *provenance semirings* in databases [30]?

The framework of *document spanners* (*spanners* for short) has been established with the aim of providing the theoretical basis to pursue the above questions and build the foundations of relational principles in IE [21]. It has been originally intro-

duced as the theoretical basis underlying IBM SystemT. Formally, in this framework a *document* is a string d over a finite alphabet, a *span* of d represents a substring of d by its start and end positions, and a *spanner* is a function that maps every document d into a relation over the spans of d [21].

The most studied instantiation of spanners is the class of *regular spanners*—the closure of *regex-formulas* (regular expressions with capture variables) under the standard operations of the relational algebra (projection, natural join, union, and difference). Equivalently, the regular spanners are the ones expressible as *variable-set automata* (vset-automata for short)—nondeterministic finite-state automata that can open and close capture variables. These spanners extract from the text relations wherein the capture variables are the attributes. The vset-automata are computationally challenging since number of extracted tuples can be exponential in the size of the automaton; hence, combined with the input string, a vset-automaton constitutes a compact representation of a relation, similarly to the concept of a Factorized Database (FDB) [44]; and as in FDB, we aim to evaluate queries over the represented relations efficiently, without materializing these relations.

In the remainder of this article, we will describe some of the research progress that has been made over recent years in an attempt of addressing the aforementioned questions. While we skip (for lack of space) any discussion on aspects of expressiveness that have been thoroughly studied [21, 25, 41, 48, 50, 57, 58], we will review recent progress on the evaluation complexity of spanners [4, 6] (Section 3), the incorporation of provenance and aggregate queries [15, 18] (Section 4) and parallel evaluation [16] (Section 5).

We note that much of the content of this article is based on prior publications of the authors [14, 15, 16, 17, 18, 19], where the reader can find the technical details that we omit here.

2. SPANNERS IN A NUTSHELL

We view a *document* (or *word*) d as a finite sequence of symbols from a finite alphabet Σ. That is, we have $d = \sigma_1 \cdots \sigma_n$ where $\sigma_i \in \Sigma$ for every $i \in \{1, \ldots, n\}$. We denote the *length* n *of* d as $|d|$. A *span of* d is an expression of the form $[i, j\rangle$ with $1 \leq i \leq j \leq n+1$, representing the interval of d that starts with the i-th symbol and ends right before the j-th symbol. For instance, the span $[23, 30\rangle$ on the document in Figure 1 represents the interval that starts at position 23 and ends right before position 30. For a span $[i, j\rangle$ of d, we denote by $d_{[i,j\rangle}$ the word $\sigma_i \cdots \sigma_{j-1}$. That is, for the document d in Figure 1, $d_{[23,30\rangle}$ is the word *Belgium*.

A *document spanner* is a function that transforms documents to *span relations*, which are database relations wherein every value is a span [21]. To a span relation we can associate a *string relation*, which is obtained by replacing every span $[i, j\rangle$ in the span relation with the string $d_{[i,j\rangle}$.

Example 2.1. *Consider the document in Figure 1. The table at the bottom left depicts a span relation over the document. The relation at the bottom right is the corresponding string relation, from which we see that the spanner extracts locations along with the corresponding number of events from the document.*

In more formal terms, spanners use *span variables* from an infinite set Vars, which is disjoint from the alphabet Σ. Let V be a finite subset of Vars. A *document spanner* (henceforth *spanner*) over V is a function S that maps each document d to a finite $|V|$-ary relation where the variables in V serve as attribute names and all the values are spans of d. (We denote by $|X|$ the cardinality of a set X.)

Spanners can be specified using a wide range of formalisms. We focus here on formalisms that are based on *regular languages*, but the literature has examples that go beyond that, such as core spanners [21, 57] (that can also be represented also via SpLog [25]), context-free languages [48], and Datalog for spanners [50].

2.1 Regular Spanners

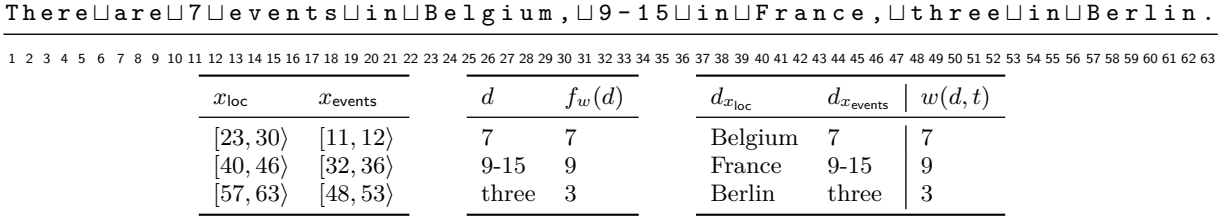
The most studied class of spanners is the class of *regular spanners*. Such spanners can be defined using *generalized regex formulas*, which are regular expressions with capture variables. Formally, we define their syntax with the inductive rule

$$\alpha := \varepsilon \mid \sigma \mid x\vdash \mid \dashv x \mid (\alpha \vee \alpha) \mid (\alpha \cdot \alpha) \mid \alpha^* ,$$

where ε denotes the empty word, $\sigma \in \Sigma$, $x \in \text{Vars}$, $\vee$ denotes disjunction, $\cdot$ denotes concatenation, and * denotes Kleene closure. We usually leave the notation of concatenation implicit. Here $x\vdash$ and $\dashv x$ are the operations that do not match a symbol in the input document, but rather "open" and "close" the variable x, respectively. To each generalized regex formula α, we can associate a language $L(\alpha)$ over the set of symbols $\Sigma \uplus \{x\vdash \mid x \in \text{Vars}\} \uplus \{\dashv x \mid x \in \text{Vars}\}$ using the standard semantics of regular expressions. Furthermore, we denote the set of variables that occur in α by $\text{Vars}(\alpha)$ (similarly for words w). In order to properly define a spanner, generalized regex formulas need some syntactic restrictions. In particular, we need to ensure that

(a) every variable is "opened" before it is "closed": for every word $w \in L(\alpha)$ and every $x \in \text{Vars}(\alpha)$, we have that $x\vdash$ occurs before $\dashv x$ in w *and*
(b) every variable is "opened" and "closed" exactly once: for every word w in $L(\alpha)$ and every $x \in \text{Vars}(\alpha)$, each of $x\vdash$ and $\dashv x$ appears exactly once in w.

We call α *functional* if it meets these two conditions. From now on in the paper, we assume that all generalized regex formulas are functional.



x_{loc}	x_{events}
$[23, 30\rangle$	$[11, 12\rangle$
$[40, 46\rangle$	$[32, 36\rangle$
$[57, 63\rangle$	$[48, 53\rangle$

d	$f_w(d)$
7	7
9-15	9
three	3

$d_{x_{\mathsf{loc}}}$	$d_{x_{\mathsf{events}}}$	$w(d,t)$
Belgium	7	7
France	9-15	9
Berlin	three	3

Figure 1: A document d (top), a span relation R (bottom left), a partial function f_w (bottom middle), and the corresponding string relation with weights $w(d,t)$ from weight function $w := (x_{\mathsf{events}}, f_w)$ (bottom right).

Before we explain the semantics of generalized regexes, we look at an example, in which we use Σ as a shorthand for $\bigvee_{\sigma\in\Sigma}\sigma$.

Example 2.2. *Consider again the document in Figure 1. Extracting the numbers (sequences of digits) from this document can be done with the regex*

$$\Sigma^*(\sqcup \vee \text{-})\ x\vdash\ (0 \vee 1 \vee \cdots \vee 9)^*\ \dashv x\ (\sqcup \vee \text{-})\Sigma^*\ .$$

Intuitively, the this regex starts by reading an arbitrary prefix, then reads the blank symbol $\sqcup$ or a dash -, and "opens" the variable x. The variable x is "closed" after reading an arbitrary number of digits, followed by a blank symbol or a dash and an arbitrary postfix. As such, the regex extracts the span relation containing the spans $[11, 12\rangle$, $[32, 33\rangle$, and $[34, 36\rangle$ from the document in Figure 1.

We define the semantics of a (functional) regular spanner α. Observe that each word $w \in L(\alpha)$ encodes two things:

- a *document* $\mathrm{doc}(w)$, which is obtained from w by deleting the symbols of the form $x\vdash$ and $\dashv x$ for variables x; and
- a *tuple* $\mathrm{tup}(w)$ *of spans* that is obtained from the positions where $x\vdash$ and $\dashv x$ occur in w.

The former is rather clear, but we explain the latter a bit more precisely. Since α is functional, every word $w \in L(\alpha)$ can be written as

$$w\ =\ u_x\ \ x\vdash\ v_x\ \ \dashv x\ \ w_x\ ,$$

in a unique manner for each $x \in \mathrm{Vars}(\alpha)$. The tuple $\mathrm{tup}(w)$ maps each variable $x \in \mathrm{Vars}(w)$ to the span $[i_x, j_x\rangle$ where $i_x = |\,\mathrm{doc}(u_x)|$ and $j_x = i_x + |\,\mathrm{doc}(v_x)|$. Notice that doc ensures that the indices i_x and j_x refer to positions in the document and do not consider other variable operations.

We can now associate to α a spanner that maps every document d to

$$\alpha(d) := \{\mathrm{tup}(w) \mid w \in L(\alpha) \text{ and } \mathrm{doc}(w) = d\}\ .$$

Notice that $\alpha(d)$ is indeed a span relation.

Spanners can be defined using finite automata. Indeed, since $L(\alpha)$ is just a regular language, we can just as well use non-deterministic finite automata over the alphabet $\Sigma \uplus \{x\vdash \mid x \in \mathrm{Vars}\} \uplus \{\dashv x \mid x \in \mathrm{Vars}\}$ to define it. Such automata, that therefore also define the class of regular spanners, are called *variable-set automata* (*vset-automata* for short) in the literature. Just as generalized regexes, they can open and close variables.

2.2 Regular Spanners Are Robust

Generalized regex formulas can define words of the form $x\vdash a\, y\vdash b \dashv x\, c \dashv y$, where the opening and closing operations of variables is not "properly" nested. The more conventional way of incorporating variables (e.g., in Perl regular expressions) is via "capturing" (or "capture groups") where variables take on complete sub-expressions of the regular expression and, in particular, feature proper nesting of variable assignments. In our formalism, such expressions correspond to a restriction of the generalized regex formulas, as we do next.

We define the syntax of *regex formulas* with the recursive rule

$$\alpha := \emptyset \mid \varepsilon \mid \sigma \mid (\alpha \vee \alpha) \mid (\alpha \cdot \alpha) \mid \alpha^* \mid x\vdash \alpha \dashv x\ ,$$

Although spanners defined by regex formulas are expressively weaker than those defined by generalized regex formulas, their expressiveness becomes the same again when they are *closed under the relational algebra operators*. Since spanners produce span relations, thus objects that live in relational database systems, it is natural to ask ourselves whether the expressiveness of a class spanners increases if we additionally allow their output span relations to be manipulated by the relational algebra operations: select, project, join, union, and difference.

Note that the algebraic operators view spans as atomic values and, in particular, disregard the underlying document. For instance, in the natural join of two spanners S and S', the join condition on two tuples $t \in S(d)$ and $t' \in S'(d)$ is that, for all $x \in \mathrm{Vars}(S) \cap \mathrm{Vars}(S')$, the spans $[i_x, j_x\rangle := t(x)$ and $[i'_x, j'_x\rangle := t'(x)$ are equal: $i_x = i'_x$ and $j_x = j'_x$. Similarly, the atomic predicates of the selection involve equality and disequality of spans (and not their associated strings). Selection conditions that involve the equality of the associated substrings give rise to *core spanners* that we discuss in the next section.

For any class $\mathcal{S}$ of spanners, we write $\mathcal{S}$ *with RA* to denote the closure of $\mathcal{S}$ under these relational algebra operations.

Theorem 2.3 (Fagin et al. [21, 22])**.** *The following are equally expressive:*
(a) vset-automata
(b) generalized regex formulas
(c) regex formulas with RA
(d) vset-automata with RA
(e) generalized regex formulas with RA

The result shows that regular spanners are a very robust class. For this reason, we will focus our attention in this article mostly on regular spanners.

2.3 Extensions of Regular Spanners

Other languages have been studied in connection to regular spanners. For example, the core spanners are the closure of the regular spanners under the positive relational algebra (union, projection and natural join) along with the selection operator that is based on the string-equality predicate. Fagin et al. [21] showed that every core spanner can be represented as a regular spanner followed by a filtering based on string equality among variables (namely the *Core Simplification Lemma* [21, Lemma 4.19]). Peterfreund et al. [50] considered the application of Datalog on top of regular spanners, and showed that the resulting language has precisely the expressiveness of the class of spanners computable in polynomial time.

3. EVALUATION

We are now ready to discuss central problems that are associated to evaluating spanners.

3.1 Enumerating the Output

Similarly to database queries, complexity analysis for spanners should account for the fact that they are required to produce a stream with potentially many answers. Hence, we adopt complexity notions of enumeration problems. Moreover, the complexity of database queries has been studied under various yardsticks of tractability and hardness, including data complexity (where the query is fixed and the data is the input), combined complexity (where both are the input), course-grained complexity (e.g., polynomial vs. exponential time) and fine-grained complexity (e.g., linear vs. quadratic time). Notably, past research has established characterizations of conjunctive queries that admit enumeration with a constant delay (i.e., time between consecutive answers) after linear-time preprocessing, that is, the time that is required just to read the input and then write the answers one by one [7, 11]. The complexity of evaluating regular spanners has been studied under these complexity concepts, and the strongest guarantee is by Amarilli et al. [3].

EVAL	
Input:	Spanner S and document d.
Task:	Compute $S(d)$.

Theorem 3.1 ([3, Theorem 1.1])**.** *Let α be a generalized regex formula and d be a document. It is possible to enumerate the span relation $\alpha(d)$ with linear preprocessing and constant delay in $|d|$, and polynomial preprocessing and delay in $|\alpha|$.*

This result was a culmination of a line of work on enumeration of the answers of MSO-definable queries on words and trees [1, 2, 4, 9, 24, 39, 42, 43, 46, 47]. It should be noted that some of this work considers the underlying word or tree to be static, while others allow updates [2, 4, 9, 24, 39, 42, 43, 46, 47], which is technically a very different setting.

3.2 Random Sampling and Counting

The problem of answer enumeration goes hand in hand with those of counting answers (i.e., calculating $|S(d)|$) and sampling answers (i.e., producing a random tuple $t \in S(d)$ with a uniform probability $1/|S(d)|$). When enumerating a large number of answers, one wishes to count the answers, faster than actually producing all answers, in order to know how far along she is, and one can sample answers in order to get insights with statistical significance about the answer space. The two tasks are related, as sampling techniques often require counting and (approximate) counting often requires sampling [6, 32].

We parameterize the counting problem by a class $\mathcal{S}$ of spanners, since the problem is intractable in general and we are interested in classes that make the problem efficiently solvable.

COUNT for $\mathcal{S}$	
Input:	Spanner S and document d.
Task:	Compute $\lvert S(d)\rvert$.

This problem has mostly been studied for spanners that are represented by vset-automata. The reason is that notions such as *unambiguity*, which lead to better complexities, are more established on automata models. Indeed, the following theorem states that, COUNT is tractable if the vset-automaton is unambiguous, whereas it is intractable for the general class of vset-automata. Loosely explained, a vset-automaton A is said to be unambiguous if, for every document d, every tuple in $A(d)$ is witnessed by a single run of A.

Throughout this article, we denote by VSA the set of all functional vset-automata and by uVSA the subset thereof that is unambiguous.

Theorem 3.2 ([6, 24])**.** *The following holds for spanners given as vset-automata:*
(a) COUNT *is* spanL*-complete.*
(b) COUNT *is in* FP *for* uVSA*.*
(c) COUNT *can be approximated by an* FPRAS*.*

Part (a) of the theorem was proved by Florenzano et al. [24, Theorem 5.2] and parts (b) and (c) were proved by Arenas et al. [6, Corollaries 4.1 and 4.2]. The complexity class spanL is generally considered to be an intractable class. A canonical problem that is spanL-complete is #NFA: given a non-deterministic finite automaton A and an integer n, how many words in $L(A)$ have length n? The #NFA problem has also been proved to be #P-complete by Kannan et al. [33]. (Indeed, under the type of reductions considered by Kannan et al., spanL = #P.)

Parts (b) and (c) of Theorem 3.2 are tractability results. That is, COUNT can be solved exactly in polynomial time (FP) if spanners are given by unambiguous vset-automata. Part (c) of Theorem 3.2 states that, even though exactly solving COUNT is intractable for VSA, which we know by part (a), it is possible to approximate the answer with a *fully polynomial-time approximation scheme* (FPRAS): a randomized algorithm that, given S, d and $\varepsilon > 0$, returns an approximation of $|S(d)|$ within a multiplicative factor of $1 \pm \varepsilon$, with high probability (say 3/4) and running time bounded by a polynomial in $|S|$, $|d|$ and $1/\varepsilon$.

Furthermore, the scientific breakthrough of finding the FPRAS in Theorem 3.2 also led to the discovery of a *polynomial-time Las Vegas uniform generator (PLVUG)* for the outputs of vset-automata: a polynomial-time randomized algorithm that, given S and d, returns a uniformly sampled $t \in S(d)$ and is allowed to fail with a small probability (say, 1/4).

Counting the answers, as well as sampling thereof, is a special form of an aggregation operation over the answers. In the next section, we look more generally at aggregate queries with spanners.

4. WEIGHTS AND AGGREGATION

The list of practically relevant computational tasks for spanners does not end with computing the entire output or counting the size of the output. In many text analysis tasks, it is desirable to compute aggregate functions over the output. For example, Radinsky et al. [52] investigate patterns in textual news in order to make predictions. In this context, one may be interested in sequences of events that are close to each other in the text and compute aggregate functions over values assigned to such events. Such values can be, for instance, monetary quantities if we focus on financial events, or casualty counts if we look for conflicts. Such scenarios are common in subsequence mining [8, 52].

We now give a drastically simplified scenario where aggregation plays a central role. Furthermore, the example illustrates why it may be interesting to avoid the computation of huge intermediate results.

Example 4.1. *Consider the following document d, describing car configurations.*

> *There are 30 additional options that you can add to the default configuration of your car. Option 1) for €140, 2) for €900,* [...]*, and the last option for €405.*

Even though there are only 30 options that one can choose from, they give rise to 2^{30}, hence over one billion possible configurations of cars. Let α be a spanner with $\mathrm{Vars}(\alpha) = \{x_1, \ldots, x_{30}\}$, extracting all possible car configurations, that is, each tuple $t \in \alpha(d)$ encodes one configuration. Therefore, the relation $\alpha(d)$ contains $2^{30} = 1,073,741,824$ tuples. Let $w(d,t)$ be the price of the configuration, encoded by t. If we would want to do aggregation over these tuples, like computing the average or median price of a car configuration, is it possible to avoid materializing the relation containing the 2^{30} tuples?

The above example is indeed just a toy example, but in effect the question is whether the materialization of an intermediate result of size in $O(|d|^{|\alpha|})$ can be avoided if one is interested in computing an aggregate value. A scenario where this question may also arise is in the development phase, where one wishes to get quick statistics about intermediate IE functions in a live manner without actually spending the time computing the entire set of answers.

This setting poses a range of new research questions. In fact, computing aggregate queries for spanners gives rise to at least two research challenges:

(a) Spanners have tuples of spans as output, but aggregation functions act on numerical values. So, how do we assign such numerical values, i.e., *weights* to the tuples in $S(d)$?
(b) How do we compute the aggregation over these weights efficiently?

In the remainder of this section, we will discuss initial approaches that we have investigated to tackle these challenges.

4.1 Aggregation Functions

Before we dive into the two aforementioned research challenges, we give definitions of some commonly used aggregation functions in databases. The definitions assume the existence of a *weight function* w that assigns weights to tuples of spans. In fact, we will formally model weight functions more generally as functions of the signature

$$w : \Sigma^* \times T \to \mathbb{Q} \,,$$

where Σ^* is the set of all documents (that use symbols in Σ) and T is the set of all tuples of spans. Using this definition, one can also take the context of a tuple inside the document into account.

Let d be a document and S be a spanner such that $S(d) \neq \emptyset$. Let w be a weight function. We define the following spanner aggregation functions:

$$\mathrm{Sum}(S, d, w) := \sum_{t \in S(d)} w(d, t)$$

$$\text{Avg}(S,d,w) := \frac{\text{Sum}(S,d,w)}{\text{Count}(S,d)}$$
$$\text{Median}(S,d,w) := \underset{t\in S(d)}{\text{median}}\, w(d,t)$$
$$\text{Min}(S,d,w) := \min_{t\in S(d)} w(d,t)$$
$$\text{Max}(S,d,w) := \max_{t\in S(d)} w(d,t)$$

It remains to discuss the weight functions in more detail. We will discuss one instantiation here and discuss less restrictive options in Section 4.3. The simplest weight functions that we have considered are *single-variable weight functions* w, which assign values based on the strings selected by one variable in the spanner. Single variable weight functions can be specified in the input as a pair (x, f_w) where x is a variable and f_w is a partial function from the set of all words to $\mathbb{Q}$. The weight of each tuple t is then defined as

$$w(d,t) = f_w(d_{t(x)}).$$

Recall that our tuples of spans are partial functions t that map variables to spans. Therefore, $t(x)$ is a span and $d_{t(x)}$ is a subword of d. We note that this notion can be easily extended to constantly many variables [14]. We illustrate these notions on a simple example.

Example 4.2. *Consider again the document in Figure 1 and assume that we wish to calculate the total number of mentioned events. The table at the bottom left depicts a possible extraction of locations with their number of evens. The table at the bottom middle depicts the partial function f_w and the table on the bottom right depicts the corresponding string relation with the associated weights. To get an understanding of the total number of events, we may want to take the sum over the weights of the extracted tuples, namely $7 + 9 + 3 = 19$.* □

4.2 Computational Complexity

A natural question is now in which cases it is possible to avoid the materialization of the potentially huge output $S(d)$, which can be in the order of $O(|d|^{2k})$, where k is the number of variables of the spanner, and at what computational cost. To this end, one can study computational problems such as the following, which are parameterized by a class $\mathcal{S}$ of spanners.

SUM for $\mathcal{S}$	
Input:	Spanner $S \in \mathcal{S}$, document $d \in \Sigma^*$, and a weight function w.
Task:	Compute $\text{Sum}(S,d,w)$.

The problems AVERAGE, MEDIAN, MIN, and MAX for a class $\mathcal{S}$ of spanners are defined analogously and just use a different aggregation function.

Theorem 4.3 ([15])**.**
(a) MAX *and* MIN *are in* FP *for* VSA.
(b) SUM, AVERAGE, *and* MEDIAN *are in* FP *for* uVSA *and are intractable for* VSA.

Here, by *intractable* we mean spanL-hard or #P-hard. Note that Theorem 4.3 states that SUM, AVERAGE and MEDIAN behave similarly to COUNT. Furthermore, as long as the weights assigned to tuples are non-negative, we can obtain a similar result as Theorem 3.2: the output of these problems can be approximated by an FPRAS. If weight functions can assign both positive and negative weights (-1 and $+1$ suffice), then the output of these problems cannot be approximated unless commonly believed conjectures do not hold.[1]

4.3 More Powerful Weight Functions

In principle, the framework does not need to limit itself to single-variable weight functions—the weight of a tuple could be any function that maps the tuple (alongside the document) into a number: the product of multiple variables, the sum of all numbers in the tuple, the difference between the leftmost and rightmost, and so on. The difficulty with this formulation is that we need to assume that this function is given as input, yet its naive representation is a table with an exponential number of rows (in the size of the spanner) and it is not realistic to assume that one can prepare it in advance. Therefore, we need to consider compact specifications of weights, and we do so via machine representations.

Polynomial-Time Weight Functions. A *polynomial-time weight functions* w is given in the input as a polynomial-time Turing Machine M that maps (d,t)-pairs to values in $\mathbb{Q}$ and defines $w(d,t) = M(d,t)$.

Not surprisingly, there are multiple drawbacks of having arbitrary polynomial-time weight functions. The first is that all considered aggregates become intractable (i.e., #P-hard or OptP-hard), even if the vset-automata in the input are already unambiguous. On the other hand, a "positional" approximation of the median is possible in the following sense. Given a vset-automaton, a document, and a parameter $\varepsilon > 0$, there is a randomized algorithm that runs in time polynomial in the input and $1/\varepsilon$ and returns a value in the $(0.5 \pm \varepsilon)$-quantile of the data with probability at least 3/4. (Notice that the median is the 0.5-quantile of the data.)

Regular Weight Functions. Since single-variable weight functions are too verbose and polynomial-time weight functions can be considered as too powerful, the question is which representation of weight functions strikes a nice balance between complexity and expressiveness. One candidate is the class of

[1] That is, depending on the aggregate, the existence of an FPRAS would either imply that RP = NP or that the polynomial hierarchy collapses.

regular weight functions that is based on the concept of a $\mathbb{K}$-*Annotator* [18].

We consider (unambiguous) functional weighted vset-automata over the tropical semiring (also called min/plus semiring)[2] and the numerical semiring.

A *regular weight function* w is represented by a functional weighted vset-automaton W (over a semiring $\mathbb{K}$) and defines $w(d,t)$ as the $\mathbb{K}$-sum of the weights that W assigns to the ref-words w that produce the tuple t on the document d, that is, the ref-words w such that $\mathrm{doc}(w) = d$ and $\mathrm{tup}(w)$ is the tuple t, restricted to the variables $\mathrm{Vars}(W)$.

There is indeed a natural hierarchy in these classes of weight functions. If we denote by SVar the single-variable weight functions, Reg the regular weight functions, UReg the regular weight functions given by unambiguous weighted spanners, and Poly the polynomial-time weight functions, then we have the following.

Theorem 4.4 ([14, 15, 18])**.** SVar $\subseteq$ UReg $\subseteq$ Reg $\subseteq$ Poly.

We will now give some complexity results for regular weight funcitons.

Theorem 4.5 ([15])**.** *The following holds for spanners given as* uVSA*-automata and* UReg *weight functions over the tropical or numerical semiring:*

(a) The problems Min*,* Max*,* Sum*, and* Average *are in* FP*.*

(b) The problem Median *is* #P*-hard.*

Recall that, even though Median is #P-hard for UReg weight functions and uVSA-automata, the $(0.5 \pm \varepsilon)$-quantile can be approximated, even for Poly weight functions and vset-automata.

The complexity landscape for regular weight functions is a bit more involved and strongly depends on the semiring of the weight function. For instance, Sum and Average for uVSA and regular weight functions over the numerical semiring are tractable, whereas in the same setting Min and Max are intractable. Orthogonaly, Min is tractable for VSA and regular weight functions over the tropical semiring whereas Max, Sum, and Average are intractable. We refer to Doleschal et al. [15] and Doleschal [14] for a more complete list of results.

5. PARALLELIZATION

In this section, we discuss the aspect of *parallelization* in query evaluation. When applied to a large document, an IE function may incur a high computational cost and, consequently, an impractical execution time. However, it is frequently the case that the program, or at least most of it, can be distributed by separately processing smaller chunks in parallel. For instance, *Named Entity Recognition* (NER) is often applied separately to different sentences [34, 35], and so are instances of *Relation Extraction* [40, 63]. Algorithms for *coreference resolution* (identification of spans that refer to the same entity) are typically bounded to limited-size windows; for instance, Stanford's well known *sieve* algorithm [53] for coreference resolution processes separately intervals of three sentences [36]. Sentiment extractors typically process individual paragraphs or even sentences [45]. It is also common for extractors to operate on windows of a bounded number N tokens (a.k.a. N*-grams* or *local contexts*) [12, 29]. Finally, machine logs often have a natural split into semantic chunks: query logs into queries, error logs into exceptions, web-server logs into HTTP messages, and so on.

Tokenization, N-gram extraction, paragraph segmentation (identifying paragraph breaks, whether or not marked explicitly [31]), sentence boundary detection, and machine-log itemization are all examples of what we call *splitters*. When IE is programmed in a development framework such as the aforementioned ones, we aspire to deliver the premise of being declarative—the developer specifies *what* end result is desired, and not *how* it is accomplished efficiently. In particular, we would like the system to automatically detect the ability to split and distribute. This ability may be crucial for the developer (e.g., data scientist) who often lacks the expertise in software and hardware engineering. We recently embarked on a principled exploration of automated inference of *split-correctness* for information extractors [17]: the ability to detect whether an IE function can be applied separately to the individual segments of a given splitter, *without changing the semantics*.

The basic motivation comes from the scenario where a long document is pre-split by some conventional splitters (like the aforelisted ones), and developers provide different IE functions. If the system detects that the provided IE function is correctly splittable, then it can utilize its multi-processor or distributed hardware to parallelize the computation. Moreover, the system can detect that IE programs are frequently splittable, and recommend the system administrator to materialize splitters upfront. Even more, the split guarantee facilitates *incremental maintenance*: when a large document undergoes a minor edit, like in the Wikipedia model, only the relevant segments (e.g., sentences or paragraphs) need to be reprocessed.

5.1 Splittability and Split-Correctness

A *splitter* is just a spanner with one variable. As such, it always transforms a document into a set of spans, which means that it can be understood as a spanner that splits the input document into pieces of text, hence the name *splitter*. Typical such pieces of text in practice are sentences, paragraphs,

[2]One can also consider the tropical semiring with max/plus, in which case the complexity results are analogous to the ones we have for the tropical semiring with min/plus, with Min and Max interchanged.

N-grams or HTTP requests. Notice that the spans in the output can overlap, as in N-grams.

In order to define splittability, we need to define the *composition* $S \circ P$ of a spanner S and a splitter P. Intuitively, $S \circ P$ is the spanner that results from evaluating S on every part of the document extracted by P, with a proper shift of the indices. More formally we obtain the output of $S \circ P$ on a document d as follows. For each span $s \in P(d)$, we consider the word d_s. We then run S on each such d_s and shift the indices of the output so that the spans refer to the intended place in d instead of d_s. Concretely, this means that, if $s = [i, j\rangle$ and if $[i_s, j_s\rangle \in S(d_s)$, then we output $[i+i_s-1, i+j_s-1\rangle$.

Example 5.1. *Consider the document in Figure 2 and a splitter P that extracts subsentences. This can be done, for example, by the following extended regular expression, where $\Sigma' = \Sigma - \{,\}$.*

$$(\varepsilon \vee (\Sigma^*\,,))\ \ x\vdash\ \Sigma'^*\ \ \dashv x\ ((\,,\Sigma^*) \vee .)\ .$$

Figure 2 shows the corresponding span relation. The composition of the spanner S from our running example and P is obtained by executing S on the subdocuments extracted by P (cf. Figure 3) and taking the union of the three relations, where every span is shifted accordingly. That is, the spans of the second relation – $[10, 16\rangle, [2, 6\rangle$ – are shifted by $31 - 1$ and the spans of the last relation – $[11, 17\rangle, [2, 7\rangle$ – are shifted by $48 - 1$. Observe that the resulting span relation is exactly the span relation in Figure 1.

Since executing S on each individual output of P enables parallelization, it is interesting if there is a difference between the output of S and $S \circ P$ on some document d. This property clearly depends on the definitions of S and P. We define this formally next. For the following definitions, recall that a spanner is a function from documents to span relations. As such, we consider two spanners to be equal if this function is the same.

A spanner S is *self-splittable* by a splitter P if

$$S = S \circ P\ .$$

If this is the case, then one can always run P over the input document d, run S over every extracted subdocument *in parallel*, and output the union of the obtained results (with indices properly shifted). The obtained result is then the same as $S(d)$, for every document d.

Another interesting scenario is the more general one where we allow the spanner on the chunks produced by P to be some spanner S_P different from S. In this case, we say that S *is splittable by P via S_P*, which is formally defined as

$$S = S_P \circ P\ .$$

If, for given S and P, such a spanner S_P exists, we say that S is *splittable* by P.

With these definitions, we formally define the following computational problems, which are again parameterized by a class $\mathcal{C}$ of spanners.

Split-Correctness for $\mathcal{C}$	
Input:	Spanners $S, S_P \in \mathcal{C}$ and splitter $P \in \mathcal{C}$.
Question:	Is S splittable by P via S_P? In other words, is $S = S_P \circ P$?

Splittability for $\mathcal{C}$	
Input:	Spanner $S \in \mathcal{C}$ and splitter $P \in \mathcal{C}$.
Question:	Is S splittable by P? In other words, is there a spanner $S_P \in \mathcal{C}$, such that $S = S_P \circ P$?

Self-Splittability for $\mathcal{C}$	
Input:	Spanner $S \in \mathcal{C}$ and splitter $P \in \mathcal{C}$.
Question:	Is S self-splittable by P? In other words, is $S = S \circ P$?

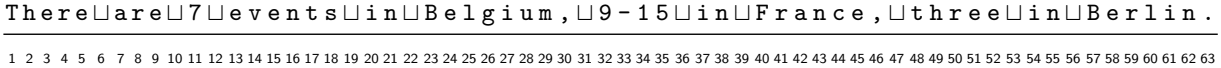
Note that the problem Self-Splittability is a special case of Split-Correctness by choosing $S_P = S$. It can also be seen as a restriction of Splittability in the sense that it implies Splittability.

We illustrate these notions by a few examples. Many spanners S that extract person names do not look beyond the sentence level. This means that, if P splits to sentences, it is the case that S is self-splittable by P. Now suppose that S extracts mentions of email addresses and phone numbers based on the formats of the tokens, and moreover, it allows at most three tokens in between; if P is the N-gram splitter, then S is self-splittable by P for $N \geq 5$ but not for $N < 5$. As another example, suppose that we analyze financial reports. Assume that S extracts those paragraphs that contain specific keywords and that P splits reports into single paragraphs. Then S is self-splittable by P. It is also splittable by P via the spanner S_P that selects the entire document if it contains the keywords.

5.2 Splitter Synthesis

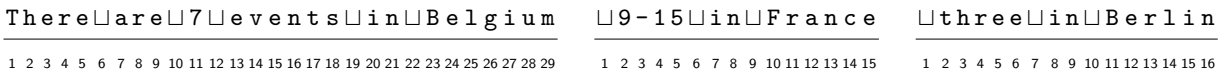
When a spanner S is splittable by P, it is natural to ask whether one can synthesise a spanner S_P such that $S = S_P \circ P$. It turns out that, in the case of regular spanners, this is indeed the case. One can define a *canonical spanner* S_P^{can} such that S is splittable by P via S_P^{can} if and only if S is splittable by P *at all*.

Theorem 5.2 ([17])**.** *Let S be a spanner and P be a splitter. Then S is splittable by P if and only if S is splittable by P via S_P^{can}.*



x	d_x
$[1, 30\rangle$	There are 7 events in Belgium
$[31, 46\rangle$	9-15 in France
$[47, 63\rangle$	three in Berlin

Figure 2: A document d (top) and the span relation (bottom) extracted by a splitter P, which splits a document into its sub sentences.



x_{loc}	x_{events}
$[23, 30\rangle$	$[11, 12\rangle$

x_{loc}	x_{events}
$[10, 16\rangle$	$[2, 6\rangle$

x_{loc}	x_{events}
$[11, 17\rangle$	$[2, 7\rangle$

Figure 3: The three sub sentences, extracted from the document in Figure 2 and the corresponding span relations.

The definition of S_P^{can} is the following:

$$S_P^{\text{can}}(d) := \big\{t \mid \forall d' \in \Sigma^*, \forall s \in P(d') \text{ such that } d'_s = d, \text{ it holds that } (t \gg s) \in S(d')\big\}.$$

Intuitively, S_P^{can} selects all tuples that are "safe to select", since they don't contradict splittability, independent of the context.

Here, if $s = [i, j\rangle$, then the tuple $(t \gg s)$ denotes the tuple obtained from t by "shifting it $i - 1$ positions to the right,", that is, adding $i - 1$ to every index in t. Interestingly, Theorem 5.2 does not assume that S or P are regular. But if they are, then the definition of S_P^{can} is useful to actually construct S_P.

5.3 Complexity of Splitting Problems

We now have a look at the complexity of the aforementioned splitting problems. It turns out that they are all decidable and even polynomial-time solvable if the involved spanners are unambiguous and they satisfy a condition that we call *highlander condition* that we explain next.

We say that $[i, j\rangle$ *covers* $[i', j'\rangle$ if $i \leq i' \leq j' \leq j$. Furthermore, if t is a tuple, we say that $[i, j\rangle$ covers t if $[i, j\rangle$ covers $t(x)$ for every variable $x \in \text{Vars}(t)$. A spanner S and a splitter P *satisfy the highlander condition* if for every document d and every tuple $t \in S(d)$ there exists at most one span $s \in P(d)$ which covers t.[3]

We note that the highlander condition is expected to be satisfied in many cases in practice. For instance, if the splitter is *disjoint* (i.e., only outputs spans $[i, j\rangle$, $[i', j'\rangle$ such that $i \leq j \leq i' \leq j'$ or $i' \leq j' \leq i \leq j$) and the spanner is *proper* (which is the case if it has at least one variable and does not return empty spans) then the highlander condition is satisfied. Typical splitters that are used in the context of tokenization, sentence boundary detection, paragraph splitting, and paragraph segmentation are disjoint. Examples of *non-disjoint* splitters include N-grams and pairs of consecutive sentences.

Theorem 5.3 ([17])**.** *The following holds for spanners and splitters given as functional vset-automata:*

(a) The Split-Correctness *problem is complete for* PSPACE*,* Splittability *is* PSPACE*-hard and in* EXPSPACE*, and* Self-Splittability *is* PSPACE*-complete.*

(b) Assuming the highlander condition and unambiguity of vset-automata, Split-Correctness *is in* PTIME *while* Splittability *is* PSPACE*-complete and, yet,* Self-Splittability *is in* PTIME*.*

Finally, we note that the highlander condition can be efficiently tested.

Proposition 5.4 ([17])**.** *Let S be a regular spanner and P be a splitter, given as functional vset-automata. Then it can be checked in polynomial time whether S and P satisfy the highlander condition.*

5.4 Reasoning with Black-Box Spanners

While we have a good understanding of splittability in the case of regular spanners, spanners in practice are often not regular and can be defined by programs that are way more complex than regular expressions or automata. It is even possible that they are just given to us as black-box algorithms for which we know some properties, such as the fact that they do not look beyond chunks of consecutive

[3]This is in acclimation to the tagline "There can be only one" of the Highlander movie.

sentences. In the following examples, we denote by $S(x, y)$ that spanner S uses the variables x and y.

Example 5.5. *Consider the spanner S that seeks to extract adjectives for Galaxy phones from reports. We define this spanner by joining three spanners:*

The spanner $S_1(x, y)$ is given by the regex formula

$$\Sigma^* \, x\vdash \mathsf{Galaxy[A-Z]} \setminus \mathsf{d}^* \dashv x \, \Sigma^* \, y\vdash \Sigma^* \dashv y \, \Sigma^*$$

that extracts mentions of Galaxy brands (e.g., Galaxy A72 and Galaxy S21) followed by substrings y that occur right before a period.

The spanner $S_2(x, x')$ is a coreference resolver (e.g., the sieve *algorithm [53]) that finds spans x' that coreference spans x. The spanner $S_3(x', y)$ finds pairs of noun phrases x' and attached adjectives y (e.g., based on a Recursive Neural Network [61]).*

For example, consider the review "I am happy with my Galaxy A72. It is stable." *Here, in one particular match, x will match (the span of)* Galaxy A72*, x' will match* it *(being an anaphora for* Galaxy A72*), and y will match* stable*. (Other matches are possible too.)*

How should a system find an efficient query plan to this join on a long report? Naively materializing each relation might be too costly: $S_1(x, y)$ may produce too many matches, and $S_2(x, x')$ and $S_3(x', y)$ may be computationally costly. Nevertheless, we may have the information that S_2 is splittable by paragraphs and that S_3 is splittable by sentences (hence, by paragraphs). This information suffices to determine that the entire join $S_1(x, y) \bowtie S_2(x, x') \bowtie S_3(x', y)$ is splittable, hence parallelizable, by paragraphs. □

Example 5.6. *Now consider the spanner that joins two spanners: $S(x)$ extracts spans x followed by the phrase* "is kind" *(e.g.,* "Barack Obama is kind"*). The spanner $S'(x)$ extracts all spans x that match person names. Clearly, the spanner $S(x)$ does not split by a natural splitter, since it includes, for instance, the entire prefix of the document before* "is kind." *However, by knowing that $S'(x)$ splits by sentences, we know that the join $S(x) \bowtie P'(x)$ splits by sentences. Moreover, by knowing that $S'(x)$ splits by 3-grams, we can infer that $S(x) \bowtie S'(x)$ splits by 5-grams. Here, again, the holistic analysis of the join infers splittability in cases where intermediate spanners are not splittable.* □

It is therefore also important to develop a theory of splittability in the presence of such black-box spanners. Next, we mention a few results that do not assume regularity. The first one states that the composition operator $\circ$ is associative and transitive.

Theorem 5.7 ([17])**.** *The spanner/splitter composition is associative and transitive. That is, for all spanners S and splitters P_1, P_2 it holds that*

(a) $S \circ (P_1 \circ P_2) = (S \circ P_1) \circ P_2$,

(b) if S is splittable by P_1 and, furthermore, P_1 is splittable by P_2, then S is splittable by P_2, and

(c) if S is self-splittable by P_1 and, furthermore, P_1 is self-splittable by P_2 then S is self-splittable by P_2.

As we will see in the following example, spanner composition does not distribute over the join operator in general.

Example 5.8. *Let*

$$S_1 := \Sigma^* \, x_1\vdash a \dashv x_1 \, x_2\vdash b \dashv x_2 \, \Sigma^* \, ,$$
$$S_2 := \Sigma^* \, x_2\vdash b \dashv x_2 \, x_3\vdash a \dashv x_3 \, \Sigma^* \, , \text{ and}$$
$$P := \Sigma^* \, x\vdash \Sigma\Sigma \dashv x \, \Sigma^* \, .$$

That is, S_1 extracts every "a" in variable x_1 which is followed by a "b", extracted in variable x_2, S_2 extracts every "b" in variable x_2 which is followed by an "a" that is extracted in variable x_3, and P extracts all substrings of length two.

Let $S := S_1 \bowtie S_2$ be the join of both spanners and let $d = aba$. It follows that $P(d) = \{[1, 3\rangle, [2, 4\rangle\}$ and $S(d) = \{t\}$, where $t(x_1) = [1, 2\rangle$, $t(x_2) = [2, 3\rangle$, and $t(x_3) = [3, 4\rangle$. As there is no span $s \in P(d)$ that covers $t \in S(d)$ it follows directly that S is not splittable by P and therefore $S \neq S \circ P$. However, both spanners, S_1 and S_2, are self-splittable by P which implies that

$$(S_1 \circ P) \bowtie (S_2 \circ P) = S_1 \bowtie S_2 = S \, .$$

It follows directly that

$$(S_1 \bowtie S_2) \circ P \;\; \neq \;\; (S_1 \circ P) \bowtie (S_2 \circ P).$$

Therefore, in general it is not true that spanner composition does distributes over join.

However, composition distributes over join if the splitter is disjoint, the spanners share at least one variable, and the join of both spanners restricted to the shared variables is proper.

Theorem 5.9 ([17])**.** *Let P be a disjoint splitter and S_1 and S_2 be spanners such that $X := \mathrm{Vars}(S_1) \cap \mathrm{Vars}(S_2) \neq \emptyset$ and the spanner $S_1 \bowtie S_2$ restricted to the variables in X is proper. Then*

$$(S_1 \bowtie S_2) \circ P \;\; = \;\; (S_1 \circ P) \bowtie (S_2 \circ P) \, .$$

Hence, Theorem 5.9 illustrates that knowing properties of spanners allows to establish query plans that might considerably optimize the computation.

6. CONCLUSIONS AND OUTLOOK

Viewing IE as a relational query over relations extracted from text allows us to incorporate and benefit from database paradigms in text processing. We have given an overview of a list of research efforts in this general direction, including query evaluation and complexity analysis, aggregate queries, provenance, and parallel-correctness. This list excludes some other efforts (due to space limitation). In particular, other database problems have been

studied in the context of document spanners, including recursion and expressiveness aspects [21, 25, 26, 27, 41, 48, 50, 57, 58], extracting incomplete information and data cleaning [17, 23, 41, 49], ranked enumeration [10, 18], queries over dynamic data (incremental maintenance) [28], and ontology mediated IE [37, 56].

Many directions are left open for future research, including aspects of system implementation, a theory of spanners based on artificial neural networks, aspects of explanations to query answers (that has gained considerable attention by the database community over the past decade), and so on. Moreover, some of the past research has only scratched the surface of the studied topics; to highlight a particular direction, we believe that there is much impactful investigation to be done on query optimization with black-box extractors based on known behavior constraints (e.g., splittability [17]). Finally, we believe that understanding queries over non-relational data using the relational view and its establishment over decades, as spanners facilitate in the case of text, could be followed by other modalities of data such as graphs, images, and voice.

Acknowledgment. This work was supported by the German-Israeli Foundation for Scientific Research and Development (GIF), grant I-1502-407.6/2019.

References

[1] A. Amarilli, P. Bourhis, L. Jachiet, and S. Mengel. A circuit-based approach to efficient enumeration. In *ICALP*, pages 111:1–111:15, 2017.

[2] A. Amarilli, P. Bourhis, and S. Mengel. Enumeration on trees under relabelings. In *ICDT*, pages 5:1–5:18, 2018.

[3] A. Amarilli, P. Bourhis, S. Mengel, and M. Niewerth. Constant-delay enumeration for nondeterministic document spanners. In *ICDT*, pages 22:1–22:19, 2019.

[4] A. Amarilli, P. Bourhis, S. Mengel, and M. Niewerth. Constant-delay enumeration for nondeterministic document spanners. *ACM Trans. Database Syst.*, 46(1):2:1–2:30, 2021.

[5] T. J. Ameloot, G. Geck, B. Ketsman, F. Neven, and T. Schwentick. Parallel-correctness and transferability for conjunctive queries. *Journal of the ACM*, 64(5):36:1–36:38, 2017.

[6] M. Arenas, L. A. Croquevielle, R. Jayaram, and C. Riveros. Efficient logspace classes for enumeration, counting, and uniform generation. In *PODS*, pages 59–73, 2019.

[7] G. Bagan, A. Durand, and E. Grandjean. On acyclic conjunctive queries and constant delay enumeration. In *CSL*, pages 208–222, 2007.

[8] K. Beedkar, R. Gemulla, and W. Martens. A unified framework for frequent sequence mining with subsequence constraints. *ACM Trans. Database Syst.*, 44(3), 2019.

[9] H. Björklund, W. Gelade, and W. Martens. Incremental XPath evaluation. *ACM Trans. Database Syst.*, 35(4):29:1–29:43, 2010.

[10] P. Bourhis, A. Grez, L. Jachiet, and C. Riveros. Ranked enumeration of MSO logic on words. In *ICDT*, pages 20:1–20:19, 2021.

[11] N. Carmeli and M. Kröll. Enumeration complexity of conjunctive queries with functional dependencies. *Theory Comput. Syst.*, 64(5):828–860, 2020.

[12] J. Chen, D. Ji, C. L. Tan, and Z. Niu. Unsupervised feature selection for relation extraction. In *IJCNLP*, 2005.

[13] L. Chiticariu, R. Krishnamurthy, Y. Li, S. Raghavan, F. Reiss, and S. Vaithyanathan. SystemT: An algebraic approach to declarative information extraction. In *ACL*, pages 128–137, 2010.

[14] J. Doleschal. *Optimization and Parallelization of RegEx Based Information Extraction*. PhD thesis, University of Bayreuth and Hasselt University, 2021.

[15] J. Doleschal, N. Bratman, B. Kimelfeld, and W. Martens. The Complexity of Aggregates over Extractions by Regular Expressions. In *ICDT*, pages 10:1–10:20, 2021.

[16] J. Doleschal, B. Kimelfeld, W. Martens, Y. Nahshon, and F. Neven. Split-correctness in information extraction. In *PODS*, pages 149–163, 2019.

[17] J. Doleschal, B. Kimelfeld, W. Martens, F. Neven, and M. Niewerth. Split-correctness in information extraction. *CoRR*, abs/1810.03367, 2021.

[18] J. Doleschal, B. Kimelfeld, W. Martens, and L. Peterfreund. Weight annotation in information extraction. In *ICDT*, pages 8:1–8:18, 2020.

[19] J. Doleschal, B. Kimelfeld, W. Martens, and L. Peterfreund. Weight annotation in information extraction. *CoRR*, 2020.

[20] A. Durand. Fine-grained complexity analysis of queries: From decision to counting and enumeration. In *PODS*, pages 331–346. ACM, 2020.

[21] R. Fagin, B. Kimelfeld, F. Reiss, and S. Vansummeren. Document spanners: A formal approach to information extraction. *Journal of the ACM*, 62(2):12:1–12:51, 2015.

[22] R. Fagin, B. Kimelfeld, F. Reiss, and S. Vansummeren. A relational framework for information extraction. *SIGMOD Rec.*, 44(4):5–16, 2015.

[23] R. Fagin, B. Kimelfeld, F. Reiss, and S. Vansummeren. Declarative cleaning of inconsistencies in information extraction. *ACM Transactions on Database Systems*, 41(1):6:1–6:44, 2016.

[24] F. Florenzano, C. Riveros, M. Ugarte, S. Vansummeren, and D. Vrgoč. Constant delay algorithms for regular document spanners. In *PODS*, pages 165–177, 2018.

[25] D. D. Freydenberger. A logic for document spanners. *Theory Comput. Syst.*, 63(7):1679–1754, 2019.

[26] D. D. Freydenberger and M. Holldack. Document spanners: From expressive power to decision problems. *Theory Comput. Syst.*, 62(4):854–898, 2018.

[27] D. D. Freydenberger and L. Peterfreund. Finite models and the theory of concatenation. *CoRR*, abs/1912.06110, 2019.

[28] D. D. Freydenberger and S. M. Thompson. Dynamic complexity of document spanners. In *ICDT*, pages 11:1–11:21, 2020.

[29] C. Giuliano, A. Lavelli, and L. Romano. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *EACL*, 2006.
[30] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40, 2007.
[31] M. A. Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, 1997.
[32] M. R. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.
[33] S. Kannan, Z. Sweedyk, and S. Mahaney. Counting and random generation of strings in regular languages. In *SODA*, pages 551–557. SIAM, 1995.
[34] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *NAACL-HLT*, pages 260–270, 2016.
[35] R. Leaman and G. Gonzalez. BANNER: an executable survey of advances in biomedical named entity recognition. In *PSB*, pages 652–663, 2008.
[36] H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, and D. Jurafsky. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *CoNLL*, pages 28–34, 2011.
[37] D. Lembo, Y. Li, L. Popa, and F. M. Scafoglieri. Ontology mediated information extraction in financial domain with mastro system-t. In *DSMM*, 2020.
[38] Y. Li, K. Bontcheva, and H. Cunningham. SVM based learning system for information extraction. In *DSMML*, pages 319–339, 2004.
[39] K. Losemann and W. Martens. MSO queries on trees: enumerating answers under updates. In *LICS*, pages 67:1–67:10, 2014.
[40] A. Madaan, A. Mittal, Mausam, G. Ramakrishnan, and S. Sarawagi. Numerical relation extraction with minimal supervision. In *AAAI*, pages 2764–2771, 2016.
[41] F. Maturana, C. Riveros, and D. Vrgoc. Document spanners for extracting incomplete information: Expressiveness and complexity. In *PODS*, pages 125–136, 2018.
[42] M. Niewerth. MSO queries on trees: Enumerating answers under updates using forest algebras. In *LICS*, pages 769–778, 2018.
[43] M. Niewerth and L. Segoufin. Enumeration of MSO queries on strings with constant delay and logarithmic updates. In *PODS*, pages 179–191, 2018.
[44] D. Olteanu and M. Schleich. Factorized databases. *SIGMOD Rec.*, 45(2):5–16, 2016.
[45] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, pages 271–278, 2004.
[46] Y. Papakonstantinou and V. Vianu. Incremental validation of XML documents. In *ICDT*, pages 47–63, 2003.
[47] S. Patnaik and N. Immerman. Dyn-fo: A parallel, dynamic complexity class. In *PODS*, pages 210–221, 1994.
[48] L. Peterfreund. Grammars for document spanners. In *ICDT*, pages 7:1–7:18, 2021.
[49] L. Peterfreund, D. D. Freydenberger, B. Kimelfeld, and M. Kröll. Complexity bounds for relational algebra over document spanners. In *PODS*, pages 320–334, 2019.
[50] L. Peterfreund, B. ten Cate, R. Fagin, and B. Kimelfeld. Recursive Programs for Document Spanners. In *ICDT*, pages 13:1–13:18, 2019.
[51] H. Poon and P. M. Domingos. Joint inference in information extraction. In *AAAI*, pages 913–918, 2007.
[52] K. Radinsky, S. Davidovich, and S. Markovitch. Learning causality for news events prediction. In *WWW*, pages 909–918, 2012.
[53] K. Raghunathan, H. Lee, S. Rangarajan, N. Chambers, M. Surdeanu, D. Jurafsky, and C. D. Manning. A multi-pass sieve for coreference resolution. In *EMNLP*, pages 492–501, 2010.
[54] A. Ratner, S. H. Bach, H. R. Ehrenberg, J. A. Fries, S. Wu, and C. Ré. Snorkel: Rapid training data creation with weak supervision. *PVLDB*, 11(3):269–282, 2017.
[55] S. Sarawagi. Information extraction. *Foundations and Trends in Databases*, 1(3):261–377, 2008.
[56] F. M. Scafoglieri and D. Lembo. A formal framework for coupling document spanners with ontologies. In *AIKE*, pages 155–162, 2019.
[57] M. L. Schmid and N. Schweikardt. A purely regular approach to non-regular core spanners. In *ICDT*, pages 4:1–4:19, 2021.
[58] M. L. Schmid and N. Schweikardt. Spanner evaluation over SLP-compressed documents. In *PODS*, 2021.
[59] W. Shen, A. Doan, J. F. Naughton, and R. Ramakrishnan. Declarative information extraction using Datalog with embedded extraction predicates. In *VLDB*, pages 1033–1044, 2007.
[60] J. Shin, S. Wu, F. Wang, C. D. Sa, C. Zhang, and C. Ré. Incremental knowledge base construction using DeepDive. *PVLDB*, 8(11):1310–1321, 2015.
[61] R. Socher, C. C. Lin, A. Y. Ng, and C. D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, pages 129–136, 2011.
[62] C. A. Sutton and A. McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2012.
[63] D. Zeng, K. Liu, Y. Chen, and J. Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In *EMNLP*, pages 1753–1762, 2015.

A Survey on Big Data Processing Frameworks for Mobility Analytics

Christos Doulkeridis[1], Akrivi Vlachou[2], Nikos Pelekis[3], Yannis Theodoridis[4]
[1]Department of Digital Systems, University of Piraeus, Greece
[2]Department of Information & Communication Systems Engineering, University of the Aegean, Greece
[3]Department of Statistics and Insurance Science, University of Piraeus, Greece
[4]Department of Informatics, University of Piraeus, Greece
{[1]cdoulk, [3]npelekis, [4]ytheod}@unipi.gr, [2]avlachou@aegean.gr

ABSTRACT

In the current era of big spatial data, the vast amount of produced mobility data (by sensors, GPS-equipped devices, surveillance networks, radars, etc.) poses new challenges related to mobility analytics. A cornerstone facilitator for performing mobility analytics at scale is the availability of big data processing frameworks and techniques tailored for spatial and spatio-temporal data. Motivated by this pressing need, in this paper, we provide a survey of big data processing frameworks for mobility analytics. Particular focus is put on the underlying techniques; indexing, partitioning, query processing are essential for enabling efficient and scalable data management. In this way, this report serves as a useful guide of state-of-the-art methods and modern techniques for scalable mobility data management and analytics.

1. INTRODUCTION

Nowadays, the ever-increasing rate of mobility data generation has resulted in vast volumes of spatio-temporal data, thus leading to new challenges for scalable processing and analysis of big mobility data. Interestingly, this applies to different domains of everyday life, from urban, to marine, and even further to air-traffic management. Miscellaneous applications and systems, such as surveillance networks, sensor readings on moving objects, human-related mobile data, social activity in location-based social networks, produce and gather positional data at rapid rates and at global scale.

In tandem with this explosion of mobility data, management of big data raises numerous research challenges [34] in different phases of the big data processing and analysis pipeline, including: (a) data acquisition, (b) data pre-processing and cleaning, (c) data integration, aggregation, and representation, (d) modeling and analysis, and (e) interpretation. The modern trend for scalable storage of massive datasets is by means of a NoSQL store [13, 16]. The exact choice depends on numerous parameters, including the type of data, the data access patterns, the purpose of data processing (read/write, read-only, etc.), as well as any special requirements with respect to the Consistency, Availability, and Partition-tolerance (also known as CAP).

Also, the current landscape of big data management comprises multiple frameworks targeting different aspects of big data. One major separating line is drawn between frameworks for batch and real-time processing, although lately some systems have been designed to tackle both cases. In the batch processing domain, Spark [65] is one of the most popular solutions nowadays with a large and growing user-base. However other solutions, such as Flink [12], are also applicable with success. In particular, Spark has successfully addressed many of the limitations of Hadoop [18], and operates in main-memory by its core abstraction: RDDs (Resilient Distributed Datasets) [64]. In the real-time processing domain, the most notable systems in use today are Storm [53] and Flink [12].

This paper provides an overview of the state-of-the-art in big data storage and processing, focusing primarily on *scalable solutions for mobility data*, i.e., spatial but most importantly spatio-temporal data. Despite the rich literature on management of spatio-temporal and mobility data, only a limited number of research prototypes attempt to address this problem in the context of big data, while most evaluations and benchmarks focus mostly on big spatial data [3, 22, 29], rather than spatio-temporal data [40]. The majority of developed prototypes extend Hadoop or Spark in order to be applicable for spatial data. In this survey, we also cover big data approaches that handle the temporal dimension.

The remaining of this survey is structured as follows: Section 2 provides background concepts related to spatio-temporal and mobility data. In Section 3, we present typical partitioning techniques

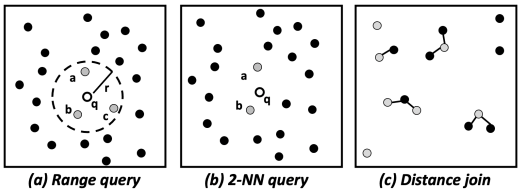


Figure 1: Basic spatial query types.

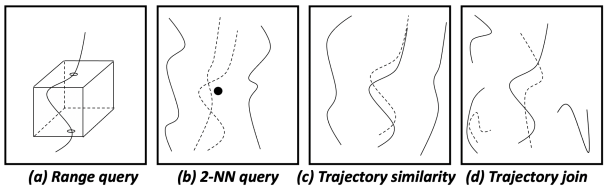


Figure 2: Basic trajectory queries.

used for mobility data. Then, in Section 4, we describe distributed indexing techniques for big spatial and spatio-temporal data. Section 5 provides an overview of query processing, focusing on range and k-NN queries as well as joins. Section 6 classifies existing storage systems and processing frameworks based on the underlying techniques that were presented in the previous sections. Finally, we conclude the paper in Section 7 and sketch future research directions.

2. BACKGROUND

In this section, we provide some basic background concepts related to query types for spatial, spatio-temporal and trajectory data.

Figure 1 depicts the most basic spatial query types for spatial point data. Obviously, these queries can be generalized for other types of spatial objects, such as polygons. In Figure 1(a), a range query is depicted which is defined by a query point q and a radius r, and retrieves all objects within distance r from q (in this example: $\{a, b, c\}$). Other ways to express the spatial constraint also exist, e.g., as a 2D box instead of a circle, but the concept remains the same. Figure 1(b) shows the case of a k-nearest neighbor (k-NN) query, defined by a point q and an integer k (in this example, the 2-NN of q are: a and b). In Figure 1(c), the case of a distance join between two data sets is depicted, where the result is pairs of objects from the two data sets that are within a user-specified distance.

Extending these queries for spatio-temporal data points is straightforward by adding time as third dimension to the query. In the case of k-nearest neighbors, different options exist, such as retrieval of the spatially k nearest objects that satisfy a temporal constraint, or the k temporally closest objects that satisfy a spatial constraint.

Figure 2 shows basic trajectory queries. On the left, a spatio-temporal range query for trajectories is depicted, which retrieves all portions of trajectories inside a spatial region during a temporal interval. Then, a k-NN query is shown, which retrieves the 2 trajectories closest to a given point. In Figure 2(c), a trajectory similarity query is depicted which, given a trajectory similarity function, retrieves the most similar trajectory to a given query trajectory. Variants of this query can use a distance threshold on similarity or retrieve the k most similar trajectories. Lastly, in Figure 2(d), the case of trajectory join is shown, where two data sets of trajectories are given, and the task is to identify pairs of trajectories that satisfy a condition (typically expressed as similarity constraint).

3. PARTITIONING TECHNIQUES

Data partitioning is the key technique for achieving efficient parallel processing of mobility data. Partitioning techniques for big mobility data have the following distinguishing features: (a) they operate on a sample of data in order to produce partitions for the complete data set, (b) they need to cope with skewed data distributions, (c) they should be adaptive both with respect to changing data distributions as well as changes in the query workload, (d) they need to balance the workload to the available nodes, which is further complicated by object duplication to nearby partitions.

3.1 Spatial and Spatio-temporal Partitioning

Partitioning techniques for the 2D space include partitioning based on Grid, STR (sort-tile-recursive), Quadtree, k-d tree, as well as mapping to 1D values using space-filling curves followed by 1D partitioning. All partitioning techniques for spatial data can also be applied for spatio-temporal data, if we consider time as another dimension. However, some frameworks for big spatio-temporal data organize data based on temporal partitions, which are further partitioned in the 2D space. As an example, ST-Hadoop [4, 6] follows this approach.

Grid partitioning. This is a standard space partitioning technique that splits the underlying space in non-overlapping cells. Several frameworks use grid partitioning, including SpatialHadoop [21], SpatialSpark [60], and GeoSpark [61].

It should be noted that sometimes the partition-

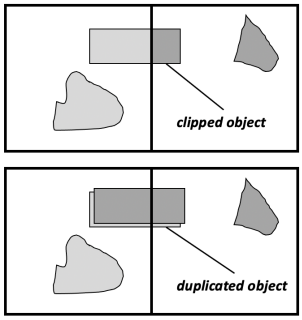


Figure 3: Object duplication vs. object clipping.

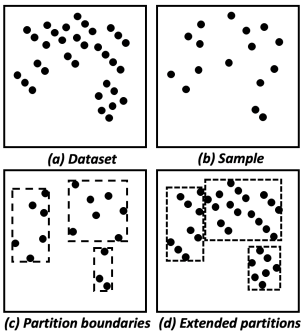


Figure 4: Sample-based data partitioning, followed by extension of partition boundaries.

ing step is followed by object duplication to neighboring cells. This is typically the case for distance joins over point data or spatial joins over data with extent. Also, in the latter case, another alternative is to perform object clipping, thus separating an object to multiple parts and assign each part to a different cell, as shown in Figure 3.

STR partitioning. A widely used partitioning scheme adopted by several prototypes (Simba [58, 59], SpatialHadoop [21], DITA [49], UlTraMan [17]) is Sort-Tile-Recursive (STR) [35], which is considered one of the best partitioning schemes for spatial data. For example, in Simba [58, 59], random sampling is performed over the input data, and then the first iteration of STR is executed to produce partition boundaries. Obviously, these partitions may not cover the entire data space, as they have been constructed based on a sample only, therefore they need to be extended to cover the entire data space, as shown in Figure 4.

R*-Grove [55] has been recently proposed for spatial partitioning, aiming to address some limitations of pre-existing partitioning schemes, such as STR. Its core idea is to use the node split algorithm of R*-tree in order to create compact, square-like partitions, in contrast to the thin and wide partitions that are often produced by STR. In addition, R*-Grove adopts a load-balancing mechanism that forces the generation of full blocks.

Quadtree partitioning. Other prototypes support Quadtree-based partitioning on a data sample as an alternative technique. Again, the aim is to produce partitions that take into account the (inferred) data distribution, and handle skewed spatial distributions gracefully. This approach is supported by frameworks such as SpatialHadoop [21], LocationSpark [52], and STJoins@ESRI [56, 57].

K-d tree partitioning. Another approach to handle skewness of input data is to use a k-d tree, in which the leaves correspond to data partitions in the distributed file system. This approach is adopted by AQWA [7] and some statistics are maintained in main memory, in order to capture the distribution of data. Furthermore, AQWA adopts an adaptive mechanism for partitioning aiming to handle changes in the query workload, where a partition may be further decomposed in case of updated data distribution or queries. Other frameworks that support partitioning based on k-d trees include SpatialHadoop [21] and SpatialSpark [60].

Partitioning based on space-filling curves. In this approach, the 2D data is mapped in 1D values using a space-filling curve (such as Z-order or Hilbert curve), and then partitions are generated by grouping the 1D values into intervals. SpatialHadoop [21] supports this type of partitioning. Also, this partitioning scheme is popular in big data storage systems, such as MD-HBase [42], Pyro [36], and QUILTS [43].

3.2 Trajectory Partitioning

Some frameworks for trajectory data management also adopt partitioning techniques such as those described above. However, other specialized partitioning techniques are also employed. For example, HadoopTrajectory [9] supports both partitioning per moving object (so as to have the complete trajectory in the same partition), as well as spatio-temporal partitions. These partitions must be small in order to avoid accessing trajectories that do not match with the query, but also large enough in order to avoid splitting trajectories into multiple partitions.

Finally, a different approach is used by DITA [49], where the STR algorithm is used, but it operates on selected points of trajectories, namely the first and last points of each trajectory. The trajectories

are grouped based on their first points, and then subgroups are created by grouping based on the last points. Intuitively, this partitioning technique aims to group together trajectories with similar starting positions and similar ending positions.

4. DISTRIBUTED INDEXING

The basic idea behind distributed indexing, which is adopted by most existing prototypes and systems, is to employ a two-level indexing scheme.

At the local level, index structures such as R-trees, Quadtrees, and Grids are typically used. An alternative approach is to map data to 1D values using space-filling curves and use traditional B-trees for local indexing. This latter approach is widely adopted by NoSQL stores. In the case of spatio-temporal data, some approaches index first the temporal dimension, and then the spatial dimensions.

At the global level, the most common approach is to assemble summary information from the local indexes of nodes, in order to build a global index for directing queries to nodes. Essentially, this summary is partition boundary information, such the Minimum Bounding Rectangles (MBRs) that describe the local data on each node. This is depicted in Figure 5, which presents the approach adopted by Simba [58, 59].

4.1 Spatial and Spatio-temporal Indexing

The combination of local and global indexing is used by several frameworks for big spatial data processing. Indicative examples of such frameworks include Hadoop-GIS [2], SpatialHadoop [21] and LocationSpark [52]. For the local indexes, classic 2D data structures are employed: R-tree, Quadtree, as well as Grid.

In the case of spatio-temporal indexing, one approach is to first organize data based on time, and then based on space. ST-Hadoop [4, 6] builds a temporal hierarchy of spatial indexes. This approach favors queries with high selectivity in the temporal dimension. Other approaches handle the three dimensions equally and build spatial indexes in the 3D space. STARK [30] uses R-trees to index spatio-temporal data, by following this idea.

4.2 Indexing Trajectory Data

In the case of trajectory data, one indexing approach is to employ the afore-mentioned solutions for 3D spatio-temporal data. As an indicative example, HadoopTrajectory [9] follows this approach and builds a grid in the 3D space or a 3DR-tree.

However, more specialized indexing techniques tailored for trajectory data are also used. DITA [49] uses global/local indexes, but proposes an approximate representation technique for trajectories based on pivot points. Two indexes are built, one for the first points of trajectories, and another one for the last points.

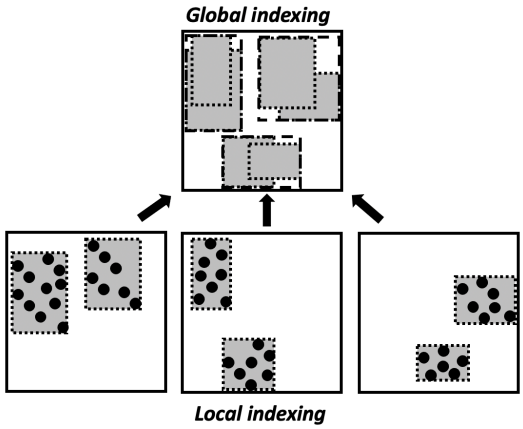


Figure 5: Example of global/local indexing.

5. QUERY PROCESSING

5.1 Big Spatial and Spatio-temporal Data

In the following, we review query processing techniques for the most standard query types for big mobility data under the global/local indexing scheme.

Range queries. Range queries comprise the most standard query type supported by all prototypes and systems. In a distributed system, processing a range query typically starts at the level of global indexing, where the partitions that overlap with the query range are identified. Then, each of these partitions is queried in parallel using its local index, and the local results are collected and returned to the user.

Nearest-neighbor queries. Typically, nearest-neighbor queries can be processed as range queries, as long as a good radius can be estimated that is guaranteed to include the k nearest neighbors. Ideally, if the distance to the k-th nearest neighbor was known in advance, we would retrieve exactly the k nearest neighbors. Consequently, the major challenge is accurate radius estimation for the query range. This issue relates to selectivity estimation for spatial queries [14, 15]. Also, in a distributed setting, the range query may intersect with multiple partitions that belong to different nodes in the system, which need to process the query, thus increasing the cost of query execution.

This approach is followed in AQWA [7], where

given a query q the surrounding cells are visited in increasing order of minimum distance (MinDist). As soon as the aggregate count of objects in the visited cells reaches k, the largest MaxDist of these cells is used as query radius.

Simba [58, 59] attempts to improve the tightness of the estimated query radius, by following a two-step approach. In the first step, the nearest partitions to q that are guaranteed to contain k points are actually queried, in order to find the k-nearest neighbors of each partition. Assuming l such partitions, then Simba uses the $l \cdot k$ candidates in order to compute the k-th minimum distance from q, and uses this value as query radius in the second step in order to ensure that the k-nearest neighbors are found. Essentially, Simba computes a tighter radius, at the cost of processing a k-nearest neighbor query locally over few partitions.

Joins. For an elaborate survey on spatial join processing, we refer to [33]. In Hadoop-based big data systems, such as SpatialHadoop [19, 21] and Hadoop-GIS [2], spatial joins are processed using the following approach: first, one data set is sampled and an in-memory index is built using the sample. Then, the leaf nodes of the index are mapped to HDFS blocks, which now contain data with spatial locality. Finally, objects are assigned to HDFS blocks based on the MBR of the block, and the join is performed between blocks, since pairs of blocks from each relation have the same MBR.

A different approach is applied in the case that data is stored without a spatial partitioning method. Quite often, a Grid-based structure is used in order to re-partition data to grid cells, followed by local join processing in each cell in parallel. To guarantee that each partition can be processed independently, one must handle the case of spatial objects that may join with objects in other cells, therefore object duplication to such cells is performed. For example, this is the approach followed in GeoSpark [61, 62]. Variations of this method include the use of data structures that are data-aware (e.g., dynamic Grid, Quadtree, R-tree, etc.) and use of leaf nodes as cells. Such data structures can typically cope better with skewed data distributions. LocationSpark [51, 52] follows a similar approach where a sample is taken from the first input, followed by the construction of a global spatial index. Then, workers re-partition their data based on the leaf nodes of the global index. When the join is processed, data from the second input are sent to the corresponding overlapping partitions, in order to produce join results locally. Notice that the overlapping partitions depend on the type of the join and the type of spatial objects. In [51], LocationSpark also reports a method for detecting skewed partitions and splitting them to smaller partitions that are repartitioned, in order to reduce the execution time.

More complex joins, as for instance k-NN joins, have also been studied in a MapReduce context, for example in [38] using Voronoi partitioning, and in [66] using Z-order and resulting in an approximate algorithm. Later, in Simba [59], a centralized processing step is used that exploits the partitions of the first dataset (built using STR), an R-tree built over a sample of the second dataset, in order to transform the join to n local k-NN joins. DISON [63] is a Spark-based approach for distributed similarity search and join for trajectory data in road networks, which computes signatures for trajectories that are used in a filter-and-refine framework. Finally, comparisons between the different systems for different join types are also of interest, e.g., for distance joins [26].

5.2 Big Trajectory Data

The most prominent and generic works in this field include HadoopTrajectory [9], DITA [49], UlTraMan [17], and MobilityDB [67], which are reviewed in Section 6.

Fang et al. [23] address the problem of k-NN join by using the MapReduce framework. More specifically, given two sets of trajectories R and S, an integer k and a time interval $[t_s, t_e]$, the objective is to return the k nearest neighbors from R for each object in S during this interval. In order to achieve this, a five step procedure (five MR jobs) is adopted, where the data are preprocessed, subtrajectories are extracted, the time-dependent upper bound is computed, candidates are found and the trajectories are joined. The intuition is to find a distance upper bound d for each trajectory of S, that includes at least k trajectories from R and then perform a plane sweep distance join based on d.

Tampakis et al. [50] study a more generic problem of sub-trajectory join, where the aim is to retrieve maximal portions of similar trajectories. Their most efficient algorithm uses a MapReduce job as preprocessing step in order to repartition data in balanced partitions based on the temporal dimension, followed by a second MapReduce job that produces the maximal sub-trajectories that join.

There is also work on joins over objects moving on road networks [47, 48]. The objective is to find all pairs of network-constrained trajectories that exceed a similarity threshold in a parallel manner. In [47], a two-phase algorithm is proposed that is parallelized and computes for each trajectory other

similar trajectories in its first phase. Then, during the second phase, it performs result merging in order to deliver the final result. However, the parallelization adopted is per trajectory, which assumes that all data need to be replicated for each trajectory, a fact that makes such a solution hard to apply in a big data setting.

6. SYSTEMS & FRAMEWORKS

In this section, we classify existing systems and frameworks for big spatial and spatio-temporal data processing. We review systems for scalable storage (Section 6.1) separately from big data processing frameworks (Section 6.2). The reason for this distinction is that although storage systems support (basic) processing, they offer only limited querying capabilities. For instance, join processing is not supported by the storage systems, whereas big data processing frameworks can compute parallel joins be re-partitioning data across nodes.

6.1 Scalable Storage

Systems that extend scalable storage solutions for multidimensional data have been proposed, most notably MD-HBase [42], but also solutions tailored specifically for spatio-temporal data (Pyro [36]) and for spatio-temporal RDF data [54], as well as spatio-textual data (ST-HBase [39]). In all these storage systems the main underlying challenge is to map spatial or spatio-temporal data (2D or 3D) to 1-dimensional (1D) values, which are used as keys for storage in key-value based NoSQL storage systems. The mapping is typically achieved using variants of space-filling curves, such as Z-order, Hilbert, or Moore encoding. An overview of the different systems is provided in Table 1.

Essentially, this mapping is necessary in order to bridge the gap between mobility data and (1D) key-based NoSQL stores. Based on this mapping to keys, data is distributed, replicated and stored based on partitioning techniques that operate at the level of 1-dimensional key. The challenge is then to translate spatial and spatio-temporal queries to multiple 1D range scans and discover efficient and scalable processing algorithms.

6.1.1 Individual Systems & Techniques

MD-HBase [42] encodes multidimensional data in 1D values using Z-order encoding. This 1D representation is then used by an index layer as a key for storing data in HBase (the storage layer). In this way, standard multidimensional index structures, such as k-d trees and Quadtrees, can be implemented on top of a distributed key-value store. By using the properties of a technique called longest common prefix naming scheme, this mapping of multidimensional indexes to 1D ranges is achieved, offering, in turn, the fundamental mechanism for answering point, range, and nearest-neighbor queries. R-HBase [32] follows a similar approach.

Pyro [36] employs the Moore encoding algorithm, inspired from the Moore space-filling curve, in order to transform (map) spatio-temporal data to 1D values. Then, range queries are translated to multiple 1D range scans, which are processed efficiently by means of different optimizations introduced at the storage layer of HDFS, resulting in PyroDFS, and at an extension of HBase, named PyroDB. In addition, a multi-scan optimizer is used to find the best reading strategy from HBase by considering multiple range scans. Also, a new block grouping algorithm is introduced at the level of the PyroDFS, which preserves data locality and improves the efficiency of dynamic load rebalancing. Pyro is shown to outperform MD-HBase by one order of magnitude for rectangular range queries.

ST-HBase [39] focuses on spatio-textual data, namely data that combines spatial location with textual description. Typical examples of spatio-textual data include geo-tagged objects, for instance tweets, images, etc. ST-HBase resembles the approach followed by MD-HBase, since it also exploits Z-order to transform spatial data to 1D values. However, it goes one step further to support combined spatial and textual retrieval, by introducing the functionality of an inverted index and representing keywords along with 1D values as key in HBase. In this way, textual filtering is supported together with spatial filtering.

GeoHashes. The concept of GeoHashes has been exploited to map spatio-temporal data to 1D values that are stored in Accumulo [25]. Practically, it relies on a hierarchical spatial data structure that partitions the data space in cells, which are then used to build string keys encoded using Base32. These keys resemble the cell identifiers produced by a space-filling curve, such as Z-order. A similar approach is taken by ST-Hash [28], where the generated 1D values are stored in MongoDB.

datAcron Encoding. In [54], an 1D encoding scheme for spatio-temporal data is proposed in the context of the H2020 datAcron project[1], which is applicable for online settings, where the temporal partitioning needs to be performed as data arrive in the system [46]. One challenge addressed by this work is dynamic temporal partitioning, since the temporal extent of the data is not known in

[1] http://datacron-project.eu/

System	1D Mapping	Data type	Data space	Data skew	Adaptive	Queries	NoSQL store
MD-HBase [42]	Z-order	multidimensional	static	–	–	range, k-NN	HBase
Pyro [36]	Moore	spatio-temporal	static	–	–	range	PyroDB
GeoHashes [25]	Z-order	spatio-temporal	static	–	–	range	N/A
ST-Hash [28]	Z-order	spatio-temporal	static	–	–	range	MongoDB
ST-HBase [39]	Z-order	spatio-textual	static	–	–	range	HBase
datAcron [54]	Z-order, Hilbert	spatio-temporal	dynamic	✓	–	range	N/A
QUILTS [43]	Generic	multidimensional	static	✓	✓	range	HBase

Table 1: Comparative overview of storage techniques.

advance, with the objective to keep compact 1D values. A space-filling curve (Z-order or Hilbert) is used for the spatial domain, while the temporal part is encoded in the same identifier. This encoding scheme has been applied for encoding spatio-temporal RDF data, and specifically in the storage layer of the DiStRDF [41] engine, which has been developed in Apache Spark.

QUILTS [43] investigates space-filling curves that fit a given data skew and query characteristics. A new method is proposed for partitioning multidimensional data based on a family of space-filling curves that take into account data skewness and query workload characteristics. QUILTS is implemented on top of HBase.

6.1.2 Classification

Even though the systems above share many similarities, they also have subtle differences that are important for providing a comprehensive classification of storage solutions. We identify four significant dimensions for the classification: data dimensionality, statically/dynamically defined data space, data skew, adaptivity to query workload. The result of this classification is summarized in Table 1.

First, regarding the *data dimensionality*, practically all approaches that rely on space-filling curves can be applied to multidimensional data. This also includes spatio-temporal data, which can be seen as 3D data. One exception is ST-HBase [39] which targets 2D spatial data and text.

The second observation is whether the underlying data is constrained to a *statically defined space* or whether the data space changes dynamically. Practically all systems make the assumption that data is defined within a multidimensional box of known size. Although data insertions and updates can be supported, the size of the data space must remain unchanged, otherwise the space partitions need to be redefined. The only notable exception is the mapping proposed in [54], which targets applications that collect streaming spatio-temporal data and need to accommodate this data in an online manner. In this setup, the problem is that the temporal dimension cannot be statically defined, and its range increases as new data arrive in the system.

Also, only limited works have explicitly focused on handling *data skew*. A noteworthy approach is QUILTS [43], which aims to identify the most appropriate 1D mapping for a given data set, under the presence of data skew. The approach in [54] also tackles this problem, focusing mainly on temporal skew in mobility data.

Finally, *adaptive* mappings have not been explored yet. The problem can be stated as finding the best mapping for a given query workload, and selecting when the system should adapt its storage at runtime (e.g., build a different 1D mapping). QUILTS [43] is the only system that identifies changes in the query workload that lead to decisions for adapting the storage scheme.

6.2 Processing Frameworks

Lately, several research projects have extended popular parallel data processing platforms, such as Hadoop or Spark, in order to provide customized solutions for big spatial or spatio-temporal data. The most prominent prototypes and systems in this field include Hadoop-GIS [2], SpatialHadoop [21], AQWA [7], ST-Hadoop [4, 6], SpatialSpark [60], GeoSpark [61] (recently joined Apache Incubator as Apache Sedona[2]), LocationSpark [52], Simba [58, 59], and STARK [30]. We also refer to [29] for a comparative evaluation of big spatial data processing systems.

6.2.1 Spatial and Spatio-temporal Frameworks

Hadoop-GIS [2] is a large-scale spatial data warehousing system for executing spatial queries in parallel. It is available both as a library and as an integrated package in Hive, thus facilitating ease of use. To support indexing, global indexes are built and replicated on all nodes using Hadoop's Distributed Cache. Thus, each node can efficiently determine the regions of the space that contain relevant re-

[2] `http://sedona.apache.org`

	Framework	Partitioning	Indexing	Queries
Spatial	Hadoop-GIS [2]	N/A	Global/local indexing (global region indexes, on demand local indexing)	Range queries (box), spatial joins
	Spatial-Hadoop [21]	Space partitioning (Grid, Quadtree), data partitioning (STR, k-d tree), space-filling curves (Z-order, Hilbert)	Global/local indexing (R-trees, Grid files)	Range queries (box), k-NN queries, spatial join
	AQWA [7]	Adaptive (based on k-d tree)	N/A	Range queries, k-NN queries
	Spatial-Spark [60]	Fixed Grid partitioning, binary space partitioning, tile partitioning	Pre-built local indexes on HDFS	Range queries, spatial join
	GeoSpark [61]	Grid-based partitioning	Local indexes (R-tree and Quadtree)	Range queries, k-NN query, spatial join
	Location-Spark [52]	Data partitioning e.g. using Quadtree (based on sampling)	Global/local indexing (global: Grid and region Quadtree, local: Grid, R-tree, Quadtree, IR-tree)	Range queries, k-NN query, spatial join, k-NN join, spatio-textual queries
	Simba [58, 59]	STRPartitioner (sampling and STR)	IndexRDD	Range queries, k-NN query, distance join, k-NN join
Spatio-temporal	ST-Hadoop [4, 6]	Multi-level temporal partitioning	Temporal hierarchy of spatial indexes at multiple levels of temporal resolution	Spatio-temporal range queries and joins
	STARK [30]	Spatial-only	R-trees	Spatio-temporal range queries and joins
	STJoins@ESRI [56, 57]	Data (re-)partitioning based on Quadtree decomposition	Equi-sized splitting of complete data set and local Quadtrees	Spatio-temporal join
Trajectory	Hadoop-Trajectory [9]	MBR-based grouping and partitioning	Global index in the form of 3D Grid or 3DR-tree	Range queries
	Parallel Secondo [37]	N/A	Local indexing using full-featured Secondo DBMS	All those offered by Secondo
	UlTraMan [17]	Supports a repartition operator to support different partitioning strategies (including STR)	In-memory: random access RDD using on-heap arrays or using ChronicleMap, an embedded, key-value store	Range query, k-NN, aggregation, comovement pattern queries
	DITA [49]	Grouping of trajectories based on first & last point, and use of STR for partitioning	Global/local indexing: (global: two R-trees built on MBR of first & last points respectively, local: trie-like index on selected points)	Similarity search, similarity join
	MobilityDB [10, 11, 67]	Hierarchical partitioning, multidimensional partitioning	Local indexing based on PostGIS	Range, k-NN, not distributed joins

Table 2: Overview of spatial and spatio-temporal parallel processing frameworks.

sults for the spatial query at hand. Local indexes are dynamically constructed on demand, using main memory. Regarding query types, Hadoop-GIS supports range queries and spatial joins.

SpatialHadoop [21] is an extension of the basic Hadoop implementation, designed for efficient processing of spatial data, that supports spatial indexing, a feature missing from basic Hadoop. SpatialHadoop utilizes a two-layered spatial index which enables selective access to data by spatial operations. Implemented indexes include R-trees, R^+-trees and Grid files. In more detail, SpatialHadoop uses a single global index and several local indexes. The global index maintains information about the data partitions across cluster nodes. The local indexes organize data stored on single nodes. Different partitioning techniques have been studied and evaluated [21] in the context of SpatialHadoop, including Grid, Quadtree, STR, STR+ and k-d trees, as well as partitioning based on Z-order and Hilbert curve. Also, a spatial MapReduce language called Pigeon [20] is also provided as part of SpatialHadoop, thus easing the development of scalable applications that process vast-sized spatial data.

AQWA [7] is a research prototype system that focuses on adaptive partitioning for big spatial data, with a strong emphasis on query-workload-aware partitioning. AQWA is demonstrated on top of Hadoop, but its techniques are in principle applicable to other systems as well. In contrast to Spa-

tialHadoop that uses static partitioning, AQWA incrementally updates the partitions based on data changes and the distribution of queries.

SpatialSpark [60] is a prototype implementation that focuses mainly on efficient processing of spatial joins in parallel, although range queries are also supported. For data partitioning to machines, data partition strategies such as fixed Grid or k-d tree are employed. SpatialSpark has implemented several spatial indexing and spatial filtering techniques, and it reuses (at the local level) the popular JTS[3] API for spatial refinement, i.e., testing whether two geometric objects satisfy a certain spatial relationship (e.g., point-in-polygon) or calculating a certain metric between two geometric objects (e.g., Eucledian distance).

GeoSpark [61] (Apache Sedona) is a framework for processing large spatial data. Essentially, it offers a spatial layer built on top of Apache Spark, aiming at providing efficient support for spatial data processing. GeoSpark uses JTS to create and process geometries in order to support different query types: range queries, k-NN, and spatial join. It provides a new abstraction named Spatial Resilient Distributed Datasets (SRDDs). Spatial RDDs, such as PointRDD and RectangleRDD, are used in order to effectively partition spatial data to different machines. Partitioning is achieved using a uniform Grid partitioning mechanism, and spatial objects that intersect more than one Grid cells are duplicated to all cells. Each RDD partition can be indexed locally using Quadtree and R-tree indexes. However, global indexing is not supported.

LocationSpark [52] is a spatial data processing system developed on top of Spark that supports different spatial operators (e.g., range, k-NN, spatial join, k-NN join). It follows the global/local indexing approach, where a global index is used (based on sampling) to partition data to cluster nodes, while local indexes are built for each partition. Different options are implemented in terms of global and local indexes. Global indexing of data partitions is achieved by sampling the data and creating equisized partitions. Each partition is locally indexed on each machine using a local index of choice, including a Grid index, an R-tree, a Quadtree, or an IR-tree. In this way, data skew can be effectively addressed. Also, the authors address query skew, by means of a query scheduler that identifies data partitions that are queried by many queries and chooses to reallocate partitions when this cost is affordable. Interestingly, processing of range queries is performed by exploiting a Spatial Bloom Filter that efficiently determines whether a point is contained in a spatial range, thus avoiding the overhead of typical cases for parallel range query processing: (a) replicating points to neighboring partitions, or (b) directing a range query to all overlapping partitions. Experiments report one order of magnitude improvement in performance compared to GeoSpark.

Simba [58, 59] is a system for in-memory spatial analytics implemented in Spark. It extends the Spark SQL engine to support spatial query processing and develops an optimizer that can exploit indexes in order to improve the performance of query processing. At a technical level, Simba introduces the concept of IndexRDDs, thus allowing efficient random access in large datasets in memory, thereby avoiding the limitation of linear (in-memory) scan of Spark when accessing RDDs. Simba supports a new partitioning type, named STRPartitioner, which performs random sampling on the input and then runs one iteration of the STR algorithm [35] in order to determine the partition boundaries. The computed partition boundaries need to be extended in order to cover the space of the complete data set.

In terms of query operators, Simba supports range queries, k-NN, distance join, and k-NN joins, and introduces new physical execution plans to Spark SQL, in order to efficiently process such spatial queries. This is a notable difference to other systems, such as GeoSpark and SpatialSpark, which are libraries implemented on top of Spark, whereas Simba introduces changes to the kernel of Spark SQL. In this way, cost-based optimization of spatial queries is also provided in Simba. Moreover, Simba supports multiple dimensions, in contrast to most other systems that are constrained to 2 dimensions. Simba is evaluated against SpatialHadoop and Hadoop GIS and is considerably faster, due to the in-memory processing. Also, Simba is shown to be more efficient than in-memory parallel processing systems, such as GeoSpark and SpatialSpark, because of its indexing and query optimizer which are built inside the query engine of Spark.

ST-Hadoop [4, 6] is an open-source MapReduce extension of Hadoop tailored for spatio-temporal data processing. Support for spatio-temporal indexing is a core feature of ST-Hadoop. It is achieved by means of a multi-level temporal hierarchy of spatial indexes. Each level corresponds to a specific time resolution (e.g., day, month, etc.). Also, at each level the entire data set is replicated and spatio-temporally partitioned based on the temporal resolution of that particular level. ST-Hadoop supports spatio-temporal range queries, aggregations

[3] `https://sourceforge.net/projects/jts-topo-suite/)`

and spatio-temporal joins. Another recent work called Summit [5] is based on ST-Hadoop, and focuses on trajectory data.

STARK [29, 30] is one of the few existing solutions targeting big spatio-temporal data. STARK addresses query processing of spatio-temporal data in Spark, whereas other approaches only consider the spatial dimensions. STARK supports spatio-temporal partitioning and indexing using R-trees. Thus, it supports spatio-temporal filtering and join operations. However, the temporal dimension is not treated equally to the spatial dimensions. For example, partitioning in [30] is performed solely based on spatial criteria, and the temporal part of a query is used to filter out data objects that do not satisfy the temporal constraint. In essence, the temporal dimension is treated as yet another dimension that can be queried, and it cannot be used for eager pruning of data in the case of a very selective temporal constraint.

STJoins@ESRI [56, 57] presents an algorithm for spatio-temporal join over large spatio-temporal data sets. It is not a complete system of a framework that supports different functionalities, rather the focus is on a specific operation. In the case that one of the inputs is relatively small and fits in memory of cluster nodes, broadcast join is employed, where the small data set is sent to all nodes, whereas the other one is partitioned to the nodes. In the case that both inputs are large, a repartition join algorithm is employed, which is called bin join.

6.2.2 Trajectory Management Frameworks

HadoopTrajectory [9] is an extension of Hadoop that integrates spatio-temporal data types, indexed access and trajectory operators. At the indexing level, a global index is constructed (either as 3D Grid or in the form of 3DR-tree), and it can be used at query time to identify relevant partitions at worker nodes to the query at hand. Partitioning of trajectories can be performed either at trajectory level or per moving object. At the processing level, various trajectory operators are implemented as MapReduce jobs, most notably trajectory range queries.

Parallel Secondo [37] is a hybrid system that is built using Hadoop in order to efficiently process mobility data. It combines Hadoop with a set of single node instances of Secondo database, which has been built for mobility data management and processing. This hybrid coupling is inspired by an earlier attempt, namely HadoopDB [1], to couple Hadoop with relational DBMSs. Parallel Secondo offers the data types and execution language as a front-end, thus enabling users to express their queries to the parallel engine transparently, while using the features of the execution language.

UlTraMan [17] proposes a unified platform for the complete management cycle of big trajectory data. It provides both storage and processing layer for trajectory data. In the storage layer, ChronicleMap is used, an embedded key-value store, which is integrated in the block manager of Apache Spark. In the processing layer, UlTraMan employs an enhanced MapReduce paradigm that provides flexible APIs to applications. Interestingly, this is one of the few approaches that target the entire lifecycle of big trajectory data, from data loading and indexing, to processing and analytics. Supported query operators include range queries, k-NN queries, and aggregation queries. In addition, co-movement pattern mining on trajectory data is also supported, demonstrating the trajectory analytics capabilities of UlTraMan. Dragoon [24] is an extension that supports both offline and online analytics.

DITA [49] is another recent research prototype that targets in-memory trajectory analytics, also extending Apache Spark. It offers an extended Spark SQL language that facilitates the declarative specification of queries, but also index construction. Furthermore, DITA extends the Catalyst optimizer of Spark SQL in order to optimize trajectory similarity queries, using cost-based optimization. At the indexing level, DITA uses global/local indexes and proposes an approximate representation technique for trajectories based on pivot points. For data partitioning the STR algorithm is used, operating on selected points of trajectories, namely the first and last points of each trajectory. The trajectories are grouped based on their first points, and then subgroups are created by grouping based on the last points. Then, the global indexing mechanism consists of two R-trees, one constructed on the MBRs of first points and another one constructed on the MBRs of last points. The local indexing is a variant of trie-based indexing which is built on top of the pivot points of trajectories. At the algorithmic/processing level, DITA adopts the filter-and-refine paradigm, in order to efficiently process similarity search and similarity joins.

MobilityDB [10, 11, 67] provides a distributed system that scales PostgreSQL horizontally over multiple workers. It uses two partitioning schemes: hierarchical (first based on time, then space) and multidimensional (where the 3D space is partitioned to cells) to distribute the data to workers in a load-balanced way. Distributed MobilityDB supports many spatio-temporal query types, but not the generic

case of distributed spatio-temporal joins (e.g., self-join on positions of moving objects) that require re-distribution of data.

7. CONCLUSIONS & OUTLOOK

In this paper, we provided a succinct overview of big data processing frameworks and techniques for spatial, spatio-temporal, and trajectory data, which are key building blocks for applications that involve mobility analytics. We presented prototype systems and frameworks both at the storage layer and at the processing layer. Moreover, we couple the presentation of individual systems with an explanation of the most prominent underlying techniques for partitioning, indexing and query processing, as a guide for further research in this field.

Regarding open problems and research directions, a challenging problem is extending big data frameworks towards handling spatio-temporal-textual retrieval, which is very common in modern applications. Incorporating text makes the setup high-dimensional and this raises challenges for effective indexing and partitioning. Existing efforts [8, 31] have so far focused on mapping/encoding the textual information in numeric values to be handled uniformly with the spatio-temporal information, however there exist limitations with respect to the used mappings and they still focus on centralized settings.

Another challenge in a big data setting relates to management of skewed spatio-temporal data, which may result in partitions of uneven size. Addressing the problem of data skew, in order to achieve load balancing and minimize the execution time of distributed query processing, is still a promising research field. In addition to this, learned indexes [44, 45] are researched lately, as an alternative way to index spatial data by learning the data distribution

Last, but not least, real-time processing and analysis of spatio-temporal data calls for stream processing frameworks and online techniques, often in conjunction with building concise data summaries and approximate processing, aiming at low latency execution. Although this direction is outside the scope of this survey, we acknowledge research initiatives in this direction [27, 46].

Acknowledgements

This work was supported by EU projects Track&Know and VesselAI (under Grant Agreements No 780754 and No 957237, respectively), and by the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Innovation (GSRI), under Grant Agreement No 1667.

8. REFERENCES

[1] A. Abouzeid, K. Bajda-Pawlikowski, D. J. Abadi, A. Rasin, and A. Silberschatz. HadoopDB: An architectural hybrid of MapReduce and DBMS technologies for analytical workloads. *Proc. VLDB Endow.*, 2(1):922–933, 2009.

[2] A. Aji, F. Wang, H. Vo, R. Lee, Q. Liu, X. Zhang, and J. H. Saltz. Hadoop-GIS: A high performance spatial data warehousing system over MapReduce. *PVLDB*, 6(11):1009–1020, 2013.

[3] M. M. Alam, S. Ray, and V. C. Bhavsar. A performance study of big spatial data systems. In *Proc. of SIGSPATIAL*, pages 1–9, 2018.

[4] L. Alarabi and M. F. Mokbel. A demonstration of ST-Hadoop: A MapReduce framework for big spatio-temporal data. *PVLDB*, 10(12):1961–1964, 2017.

[5] L. Alarabi and M. F. Mokbel. A demonstration of Summit: A scalable data management framework for massive trajectory. In *Proc. of MDM*, pages 226–227, 2020.

[6] L. Alarabi, M. F. Mokbel, and M. Musleh. ST-Hadoop: A MapReduce framework for spatio-temporal data. In *Proc. of SSTD*, pages 84–104, 2017.

[7] A. M. Aly, A. R. Mahmood, M. S. Hassan, W. G. Aref, M. Ouzzani, H. Elmeleegy, and T. Qadah. AQWA: Adaptive query-workload-aware partitioning of big spatial data. *PVLDB*, 8(13):2062–2073, 2015.

[8] Y. Arseneau, S. Gautam, B. G. Nickerson, and S. Ray. STILT: Unifying spatial, temporal and textual search using a generalized multi-dimensional index. In *Proc. of SSDBM*, pages 11:1–11:12. ACM, 2020.

[9] M. S. Bakli, M. A. Sakr, and T. H. A. Soliman. HadoopTrajectory: A Hadoop spatiotemporal data processing extension. *J. Geogr. Syst.*, 21(2):211–235, 2019.

[10] M. S. Bakli, M. A. Sakr, and E. Zimányi. Distributed mobility data management in MobilityDB. In *Proc. of MDM*, pages 238–239, 2020.

[11] M. S. Bakli, M. A. Sakr, and E. Zimányi. Distributed spatiotemporal trajectory query processing in SQL. In *Proc. of SIGSPATIAL*, pages 87–98, 2020.

[12] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas. Apache Flink™: Stream and batch processing in a single engine. *IEEE Data Eng. Bull.*, 38(4):28–38, 2015.

[13] R. Cattell. Scalable SQL and NoSQL data stores. *SIGMOD Record*, 39(4):12–27, 2010.

[14] H. Chasparis and A. Eldawy. Experimental evaluation of selectivity estimation on big spatial data. In *Proc. of GeoRich*, pages 8:1–8:6, 2017.

[15] A. Das, J. Gehrke, and M. Riedewald. Approximation techniques for spatial data. In *Proc. of SIGMOD*, pages 695–706, 2004.

[16] A. Davoudian, L. Chen, and M. Liu. A survey on NoSQL stores. *ACM Comput. Surv.*, 51(2), 2018.

[17] X. Ding, L. Chen, Y. Gao, C. S. Jensen, and H. Bao. UlTraMan: A unified platform for big trajectory data management and analytics. *PVLDB*, 11(7):787–799, 2018.

[18] C. Doulkeridis and K. Nørvåg. A survey of large-scale analytical query processing in MapReduce. *VLDB J.*, 23(3):355–380, 2014.

[19] A. Eldawy, L. Alarabi, and M. F. Mokbel. Spatial partitioning techniques in SpatialHadoop. *PVLDB*, 8(12):1602–1605, 2015.

[20] A. Eldawy and M. F. Mokbel. Pigeon: A spatial MapReduce language. In *Proc. of ICDE*, pages 1242–1245, 2014.

[21] A. Eldawy and M. F. Mokbel. SpatialHadoop: A MapReduce framework for spatial data. In *Proc. of ICDE*, pages 1352–1363, 2015.

[22] A. Eldawy and M. F. Mokbel. The era of big spatial data: A survey. *Foundations and Trends in Databases*, 6(3-4):163–273, 2016.

[23] Y. Fang, R. Cheng, W. Tang, S. Maniu, and X. S. Yang. Scalable algorithms for nearest-neighbor joins on big trajectory data. *IEEE Trans. Knowl. Data Eng.*, 28(3):785–800, 2016.

[24] Z. Fang, L. Chen, Y. Gao, L. Pan, and C. S. Jensen. Dragoon: A hybrid and efficient big trajectory

management system for offline and online analytics. *The VLDB Journal*, 30:287–310, 2021.
[25] A. D. Fox, C. N. Eichelberger, J. N. Hughes, and S. Lyon. Spatio-temporal indexing in non-relational distributed databases. In *Proc. of IEEE Big Data*, pages 291–299, 2013.
[26] F. García-García, A. Corral, L. Iribarne, M. Vassilakopoulos, and Y. Manolopoulos. Efficient distance join query processing in distributed spatial data management systems. *Inf. Sci.*, 512:985–1008, 2020.
[27] N. Giatrakos, E. Alevizos, A. Artikis, A. Deligiannakis, and M. N. Garofalakis. Complex event recognition in the big data era: A survey. *VLDB J.*, 29(1):313–352, 2020.
[28] X. Guan, C. Bo, Z. Li, and Y. Yu. ST-hash: An efficient spatiotemporal index for massive trajectory data in a NoSQL database. In *Proc. of Geoinformatics*, pages 1–7, 2017.
[29] S. Hagedorn, P. Götze, and K. Sattler. Big spatial data processing frameworks: Feature and performance evaluation. In *Proc. of EDBT*, pages 490–493, 2017.
[30] S. Hagedorn, P. Götze, and K. Sattler. The STARK framework for spatio-temporal data analytics on Spark. In *Proc. of BTW*, pages 123–142, 2017.
[31] T. Hoang-Vu, H. T. Vo, and J. Freire. A unified index for spatio-temporal keyword queries. In *Proc. of CIKM*, pages 135–144, 2016.
[32] S. Huang, B. Wang, J. Zhu, G. Wang, and G. Yu. R-HBase: A multi-dimensional indexing framework for cloud computing environment. In *Proc. of ICDMW*, pages 569–574, 2014.
[33] E. H. Jacox and H. Samet. Spatial join techniques. *ACM Trans. Database Syst.*, 32(1):7, 2007.
[34] H. V. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, and C. Shahabi. Big data and its technical challenges. *Commun. ACM*, 57(7):86–94, 2014.
[35] S. T. Leutenegger, J. M. Edgington, and M. A. López. STR: A simple and efficient algorithm for R-tree packing. In *Proc. of ICDE*, pages 497–506, 1997.
[36] S. Li, S. Hu, R. K. Ganti, M. Srivatsa, and T. F. Abdelzaher. Pyro: A spatial-temporal big-data storage system. In *Proc. of USENIX*, pages 97–109, 2015.
[37] J. Lu and R. H. Güting. Parallel SECONDO: Practical and efficient mobility data processing in the cloud. In *Proc. of IEEE Big Data*, pages 17–25, 2013.
[38] W. Lu, Y. Shen, S. Chen, and B. C. Ooi. Efficient processing of k nearest neighbor joins using MapReduce. *Proc. VLDB Endow.*, 5(10):1016–1027, 2012.
[39] Y. Ma, Y. Zhang, and X. Meng. ST-HBase: A scalable data management system for massive geo-tagged objects. In *Proc. of WAIM*, pages 155–166, 2013.
[40] S. Maguerra, A. Boulmakoul, L. Karim, and B. Hassan. A survey on solutions for big spatio-temporal data processing and analytics. In *Proc. of INTIS*, pages 127–140, 2018.
[41] P. Nikitopoulos, A. Vlachou, C. Doulkeridis, and G. A. Vouros. DiStRDF: Distributed spatio-temporal RDF queries on Spark. In *Proc. of BMDA*, pages 125–132, 2018.
[42] S. Nishimura, S. Das, D. Agrawal, and A. El Abbadi. MD-HBase: A scalable multi-dimensional data infrastructure for location aware services. In *Proc. of MDM*, pages 7–16, 2011.
[43] S. Nishimura and H. Yokota. QUILTS: Multidimensional data partitioning framework based on query-aware and skew-tolerant space-filling curves. In *Proc. of SIGMOD*, pages 1525–1537, 2017.
[44] V. Pandey, A. van Renen, A. Kipf, J. Ding, I. Sabek, and A. Kemper. The case for learned spatial indexes. In *Proc. of AIDB*, 2020.
[45] J. Qi, G. Liu, C. S. Jensen, and L. Kulik. Effectively learning spatial indices. *Proc. VLDB Endow.*, 13(11):2341–2354, 2020.
[46] G. M. Santipantakis, A. Glenis, K. Patroumpas, A. Vlachou, C. Doulkeridis, G. A. Vouros, N. Pelekis, and Y. Theodoridis. SPARTAN: Semantic integration of big spatio-temporal data from streaming and archival sources. *Future Gener. Comput. Syst.*, 110:540–555, 2020.
[47] S. Shang, L. Chen, Z. Wei, C. S. Jensen, K. Zheng, and P. Kalnis. Trajectory similarity join in spatial networks. *Proc. VLDB Endow.*, 10(11):1178–1189, 2017.
[48] S. Shang, L. Chen, Z. Wei, C. S. Jensen, K. Zheng, and P. Kalnis. Parallel trajectory similarity joins in spatial networks. *VLDB J.*, 27(3):395–420, 2018.
[49] Z. Shang, G. Li, and Z. Bao. DITA: Distributed in-memory trajectory analytics. In *Proc. of SIGMOD*, pages 725–740, 2018.
[50] P. Tampakis, C. Doulkeridis, N. Pelekis, and Y. Theodoridis. Distributed subtrajectory join on massive datasets. *ACM Trans. Spatial Algorithms Syst.*, 6(2):8:1–8:29, 2020.
[51] M. Tang, Y. Yu, W. G. Aref, A. R. Mahmood, Q. M. Malluhi, and M. Ouzzani. LocationSpark: In-memory distributed spatial query processing and optimization. *CoRR*, abs/1907.03736, 2019.
[52] M. Tang, Y. Yu, Q. M. Malluhi, M. Ouzzani, and W. G. Aref. LocationSpark: A distributed in-memory data management system for big spatial data. *PVLDB*, 9(13):1565–1568, 2016.
[53] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, S. Mittal, and D. V. Ryaboy. Storm@twitter. In *Proc. of SIGMOD*, pages 147–156, 2014.
[54] A. Vlachou, C. Doulkeridis, A. Glenis, G. M. Santipantakis, and G. A. Vouros. Efficient spatio-temporal RDF query processing in large dynamic knowledge bases. In *Proc. of SAC*, pages 439–447, 2019.
[55] T. Vu and A. Eldawy. R*-Grove: Balanced spatial partitioning for large-scale datasets. *Frontiers Big Data*, 3:28, 2020.
[56] R. T. Whitman, B. G. Marsh, M. B. Park, and E. G. Hoel. Distributed spatial and spatio-temporal join on Apache Spark. *ACM Trans. Spatial Algorithms Syst.*, 5(1):6:1–6:28, 2019.
[57] R. T. Whitman, M. B. Park, B. G. Marsh, and E. G. Hoel. Spatio-temporal join on Apache Spark. In *Proc. of SIGSPATIAL*, pages 20:1–20:10, 2017.
[58] D. Xie, F. Li, B. Yao, G. Li, Z. Chen, L. Zhou, and M. Guo. Simba: Spatial in-memory big data analysis. In *Proc. of SIGSPATIAL*, pages 86:1–86:4, 2016.
[59] D. Xie, F. Li, B. Yao, G. Li, L. Zhou, and M. Guo. Simba: Efficient in-memory spatial analytics. In *Proc. of SIGMOD*, pages 1071–1085, 2016.
[60] S. You, J. Zhang, and L. Gruenwald. Large-scale spatial join query processing in cloud. In *Proc. of ICDEW*, pages 34–41, 2015.
[61] J. Yu, J. Wu, and M. Sarwat. A demonstration of GeoSpark: A cluster computing framework for processing big spatial data. In *Proc. of ICDE*, pages 1410–1413, 2016.
[62] J. Yu, Z. Zhang, and M. Sarwat. Spatial data management in Apache Spark: The GeoSpark perspective and beyond. *GeoInformatica*, 23(1):37–78, 2019.
[63] H. Yuan and G. Li. Distributed in-memory trajectory similarity search and join on road network. In *Proc. of ICDE*, pages 1262–1273, 2019.
[64] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proc. of NSDI*, pages 15–28, 2012.
[65] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. Apache Spark: A unified engine for big data processing. *Commun. ACM*, 59(11):56–65, 2016.
[66] C. Zhang, F. Li, and J. Jestes. Efficient parallel kNN joins for large data in MapReduce. In E. A. Rundensteiner, V. Markl, I. Manolescu, S. Amer-Yahia, F. Naumann, and I. Ari, editors, *Proc. of EDBT*, pages 38–49, 2012.
[67] E. Zimányi, M. A. Sakr, and A. Lesuisse. MobilityDB: A mobility database based on PostgreSQL and PostGIS. *ACM Trans. Database Syst.*, 45(4):19:1–19:42, 2020.

We are starting a new series to provide advice to mid-career researchers. There are a number of programs that SIGMOD organizes for researchers at the beginning of their careers (PhD Symposium and the like) and senior people do not (or should not) need much help. There are considerable challenges for those who are about to transition from an early researcher to a more senior role. In academia, these are people who are about to get tenured that comes with starting to think of moving from shorter-term research objectives to longer-term ones. In industrial research, this corresponds to the transition from participating in projects to initiating and leading them. As a community we don't seem to talk about these challenges much. That is the gap this series attempts to fill. We will get the views of senior researchers from diverse backgrounds and diverse geographies. We will continue as long as we find original advice and the views are not repetitions.

M. Tamer Özsu
University of Waterloo

Making the Right Move to Senior Researcher: Some Challenges and Hints

Patrick Valduriez, Inria, France

I have been working on research in data management for the last 40 years. I like my job and my research institution (Inria, the French national research institute for computer science), which have offered me great opportunities to learn a lot, do good work, get to know smart and nice people and overall feel useful. However, since the early days of my mid-career, the research environment, including academia and industry, has certainly become more complex, making the move from junior (or pre-tenure) researcher to senior researcher quite challenging. Based on my experience, I review some of the main questions and challenges and give some hints on how to deal with them. I'll sometimes use stories and anecdotes to illustrate the point.

Let's start with a basic question: *why do you want to do* (*or keep doing*) *research*? This is an important question you should reflect on, since research requires major personal investment. I have heard many different answers, e.g., I want to make the world a better place, I like the freedom given by my job, it is fun (e.g., programming), I want to be the best, … Whatever your reason is, the motivation must be very strong and profound (not just guided by your ego), in particular, to deal with ups and downs in funding, projects and results. The question may also help you to consider careers outside of research. After all, if you survived as a junior researcher, you have developed proficiencies such as creativity, autonomy, rigor, reliability, communication, etc., which will be invaluable in many other domains.

The first challenge in rising to a senior researcher is the necessary shift from "woodcutter" to "forester". In the mid 1980s, as a junior researcher at Microelectronics and Computer Technology Corporation (Austin, Texas), during one of my yearly performance reviews, my boss said to me something like: "So far, you have been a great woodcutter, taking one problem after the other and giving it a nice solution. But it is now time to move on as a forester, looking at the overall forest, so you can choose the best trees by yourself." This was a great advice, which required a radical shift in the way to do research. A junior researcher is often given research directions to work on, with full control over the solutions and implementation details. The shift to becoming a forester requires working on problems for which you may not have all the knowledge and skills to solve. Thus, you will have to learn new scientific methods and move outside of your comfort zone. You should also work with experts from other fields, which requires much humility and patience as you realize they know so much more in their field.

The next challenge is to shape a research agenda. This is the hard part as a serious project typically involves several other researchers and students, so you will also have to turn into a team leader. You may also choose to work on several unrelated projects, which will be even harder as it may lead to dispersion of attention and effort. I've always preferred to focus on a single, ambitious (high-risk) project, perhaps decomposed into several smaller projects, as it yields more overall

coherence and synergy between people working on different pieces. A good example is building a new database system (a big project), with smaller projects such as query engine, transaction manager and storage engine.

There are two approaches to come up with a project: reactive and proactive. The reactive approach, e.g., proposing a project to answer a call for funding, is easier as most of the burden of identifying the research directions may have been done by the people who wrote the call, but may not benefit from all your creative potential. The proactive approach is harder as you need to identify a real, challenging problem that you can solve. This may take time, talking to many people, e.g., close colleagues as well as people from applications and industry, to understand the domain and state of the art. For instance, to come up with my Zenith project on scientific data management, I spent much time talking to colleagues in machine learning and data mining, as well as to scientists, e.g., plant biologists, to understand their requirements in data management. The proactive approach yields more freedom than the reactive approach, as you are not bound by specific calls, and may have higher impact, e.g., by setting a completely new research direction. And you can either find generic calls for funding, or apply to specific calls by adapting the project to fit in.

To lead the team in charge of the project, you will need to develop management skills. This takes time as just taking a management class is not going to be enough. You will also have to learn how to deal with smart but very different people and have them work together as a unified body. Books and courses will help, but real experience will be critical. At the same time, you should be careful not to lose technical skills, which are useful to communicate with the other researchers, students and engineers who will do the actual work. One way to do this is to take your share in the project development, e.g., do programming, which may be challenging but also fun. In a world controlled by technology, technical skills will always be an asset to realizing your own vision and not to be fooled by techie woodcutters. Many years ago, I was teaching databases at a top engineering school in Paris (a so-called "grande école" in France). A student was complaining that it was too technical, which is true: databases is a very technical subject. So, I asked: "But aren't you supposed to learn engineering, which is quite technical?". And the student gave me that answer: "Yes, but I want to be a manager". Then, I continued with the advice I just gave you.

Having a great research project is a first step, but not enough as you will also need the right people with complementary expertise to help you along the way. Within your organization, collaborating with colleagues from other teams will be much more productive and more pleasant than competition. It will also save you much pain and time lost in all kinds of useless fights. Collaboration is also more and more international, leveraging diversity of ideas, practices and resources. International cooperation is also a good way to obtain funding, e.g., the European Commission is instrumental in supporting big projects in several countries, even outside Europe. Furthermore, some research institutions have very strong international programs, e.g., Inria's associated team program that funds tight collaboration with a foreign team for up to five years. To succeed in collaboration, you must develop a solid network of collaborators, both within and outside your organization, and keep nurturing it. This takes time and patience as you will have to learn to work with different people. In the long run, it is very rewarding and pleasant, and some collaborators become best friends.

Once you have the right project and the right people, you need to produce results, yet another major challenge. I advise producing high-quality papers, always favoring quality and long-term impact over quantity. Thus, it is best to avoid the LPU (least publishable unit) strategy. Not only does it generate just too many papers to get read and referenced, it is also counter-productive if you look for promotion at a serious research institution, where for instance, you will be asked to give your top 5 papers that will get read by the selection committee. Publishing impactful papers is hard and takes time, as extensive validation and often reproducible results are required. Producing software is also important, not only to validate the research results but to deliver artefacts that other researchers can use and build on, e.g., open-source software. Some highly successful projects even go the extra mile of creating a user community around the software, with much long-term impact. For instance, the Ingres DBMS project at UC Berkeley in the mid-1970s has been the basis for the PostgreSQL community that has thousands of developers today. Another example is the Pl@nNet

platform for plant identification developed in my Zenith team with plant scientists from other French institutions (CIRAD, INRAe and IRD), which has millions of mobile phone end-users world-wide.

A final challenge is to be a role model for people around you, which means behaving ethically. The ACM Code of Ethics and Professional Conduct is an excellent guide for ethical decision-making. In a world that is more and more controlled by algorithms and data science, it is important to act responsibly, with a good understanding of the impact of our work on people's lives and planet Earth. This raises many questions on how to make algorithmic decision-making fair, accountable and transparent (FAT), as discussed at the VLDB 2018 panel on data and algorithmic ethics.

Let me end with the common issue of searching for the elusive perfect place to do research. I have met many researchers during my career who, even though they were relatively happy with their job and organization, were always looking for the next better place. Of course, there are many good places, both in academia and industry, and doing the right move at the right time, e.g., at the end of a project, should be quite beneficial for your career. However, searching for the perfect place to work may be endless and illusive. Let me illustrate with this nice tale. There was an old man sitting at the entrance of a city. A stranger comes and asks: "I have never been to this city; what are the people who live here like?" The old man replies: "How were the people in the city you came from?" The stranger says: "Selfish and mean. That's why I left." The old man continues: "You will find the same here." A little later, another stranger approaches and asks: "I have just arrived, tell me what are the people who live in this city like." The old man replies: "How were the people in the city you came from?" The stranger says: "They were good and welcoming. I had many friends." The old man says: "You will find the same here." Someone who watched the scene asks: "How can you give two completely different answers to the same question?" The old man replies: "Well, each of us carries its own universe." Thus, you may want to consider changing the way you look at your current situation, before considering moving to the next place.

Transparent Data Transformation

Can I have my Rows and eat my Columns too?

Manos Athanassoulis
Boston University

A big dichotomy in data system design is the column- vs. row-stores one. The first supports analytical, and the latter transactional workloads. There have been several efforts to bridge the two, especially in light of new hybrid transactional analytical processing workloads.

For example, SAP HANA [11] and Oracle TimesTen [8] use in-memory column-stores to offer efficient analytical processing and employ a row-wise write-store to support ACID transactions. MemSQL uses a row-store for data ingestion in memory, and writes columns on disk to reduce future I/O [13]. IBM dashDB is a hybrid store that supports mixed workloads [4]. Academic systems [2, 3, 9] also combine columnar and row-wise architectures. They all require conversions between row-wise and columnar formats, and, hence, they have to balance between efficient analytics and data freshness. ***What if we had a way to decouple the physical data layout from the data access performance?***

In other words, what if we could store *any layout* L_1, and ship through the memory hierarchy *any layout* L_2, transparently converting rows to columns and vice versa? We now have two fundamental questions: (a) how to offer such *transparent data transformation*? (b) once we have it, how to *change the data system's architecture*? In this short note we focus on the first challenge.

Data movement is one of the biggest inefficiencies in computing [5], so the initial motivation of column-stores is to avoid unnecessary column accesses. Row-wise layouts allow transactional workloads to ensure update and insert locality and avoid updating multiple locations.

In an in-memory system, we can address both these requirements with a *smart* memory controller that *implements projection and offers access to arbitrary groups of columns*. As Moore's law is slowing down, domain-specific accelerators are becoming viable in the long run [6], hence, we envision to embed a light-weight vertical partitioner in the memory c ontroller to on-the-fly project arbitrary groups of columns. This controller will have access to a much higher bandwidth than what is exposed to the processor, as it will be able to exploit in full the parallelism of memory chips. The query processor can decide at compile-time the optimal layout for each query and request it from the memory controller. From the software side, a special type of *ephemeral variables* can act as pointers to virtual hybrid layouts (similar to a non-materialized view). Accessing them would start the projection machinery and propagate through the cache hierarchy the desired layout without creating a physical copy in main memory. This vision has similar motivation with disaggregated memory [10], however, it aims to build local *smart* memory, that can also allow cloud systems to access disaggregated *smart* memory.

Similarly, in a disk-based system, we can address these challenges by pushing projection to storage and send up the memory hierarchy the desired data layout for each query. By employing modern devices like *smart SSDs* [7], we can implement in-storage custom logic to vertically partition data. Analytical queries will access read-only versions of their optimal layout without creating permanent copies in memory. Updates will access row-oriented base data, and existing read-only versions of columns will be invalidated when relevant updates are received and old queries complete. Pages with column projections are marked as read-only, while pages with full rows are marked as read/write. The two levels of *transparent data transformation* can work synergistically.

How to achieve this? Building an FPGA prototype memory controller requires collaboration between data management and hardware researchers [1]. Near-data execution of relational operators [14] using programmable logic in the middle [12] will use an on-the-fly column extractor which compacts the useful data and ships it through the cache hierarchy. Ultimately, our goal is to build new memory and disk controllers to offer *transparent data transformation*.

What is next? If a query can consume data with the ideal layout, how should query optimization change? What about heuristics, like pushing selection? Can this be extended to allow for efficient access in arbitrary slices of tensors and matrices? *This vision will fuel strong interaction between data management, hardware, programming languages and software engineering.*

1. REFERENCES

[1] G. Alonso, T. Roscoe, D. Cock, M. Ewaida, K. Kara, D. Korolija, D. Sidler, and Z. Wang. Tackling Hardware/Software co-design from a database perspective. In *Proceedings of the Biennial Conference on Innovative Data Systems Research (CIDR)*, 2020.

[2] R. Appuswamy, M. Karpathiotakis, D. Porobic, and A. Ailamaki. The Case For Heterogeneous HTAP. In *Proceedings of the Biennial Conference on Innovative Data Systems Research (CIDR)*, 2017.

[3] J. Arulraj, A. Pavlo, and P. Menon. Bridging the Archipelago between Row-Stores and Column-Stores for Hybrid Workloads. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 583–598, 2016.

[4] R. Barber, G. M. Lohman, V. Raman, R. Sidle, S. Lightstone, and B. Schiefer. In-Memory BLU Acceleration in IBM's DB2 and dashDB: Optimized for Modern Workloads and Hardware Architectures. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, 2015.

[5] W. J. Dally. Power, programmability, and granularity: The challenges of ExaScale computing. In *Proceedings of the IEEE International Test Conference (ITC)*, page 12, 2011.

[6] W. J. Dally, Y. Turakhia, and S. Han. Domain-specific hardware accelerators. *Communications of the ACM*, 63(7):48–57, 2020.

[7] J. Do, Y.-S. Kee, J. M. Patel, C. Park, K. Park, and D. J. DeWitt. Query processing on smart SSDs: opportunities and challenges. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1221–1230, 2013.

[8] T. Lahiri, M.-A. Neimat, and S. Folkman. Oracle TimesTen: An In-Memory Database for Enterprise Applications. *IEEE Data Engineering Bulletin*, 36(2):6–13, 2013.

[9] H. Lang, T. Mühlbauer, F. Funke, P. A. Boncz, T. Neumann, and A. Kemper. Data Blocks: Hybrid OLTP and OLAP on Compressed Storage using both Vectorization and Compilation. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2016.

[10] K. T. Lim, Y. Turner, J. R. Santos, A. AuYoung, J. Chang, P. Ranganathan, and T. F. Wenisch. System-level implications of disaggregated memory. In *Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 189–200, 2012.

[11] N. May, A. Böhm, and W. Lehner. SAP HANA - The Evolution of an In-Memory DBMS from Pure OLAP Processing Towards Mixed Workloads. In *Proceedings of the Datenbanksysteme fur Business, Technologie und Web (BTW)*, pages 545–563, 2017.

[12] S. Roozkhosh and R. Mancuso. The Potential of Programmable Logic in the Middle: Cache Bleaching. In *Proceedings of the Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 296–309, 2020.

[13] N. Shamgunov. The MemSQL In-Memory Database System. In *Proceedings of the International Workshop on In-Memory Data Management and Analytics (IMDM)*, 2014.

[14] S. L. Xi, O. Babarinsa, M. Athanassoulis, and S. Idreos. Beyond the Wall: Near-Data Processing for Databases. In *Proceedings of the International Workshop on Data Management on New Hardware (DAMON)*, page 2, 2015.

Sourav Bhowmick Speaks Out on Drawing Your Queries, Ethics, and Drawing for Ethics

Marianne Winslett and Vanessa Braganholo

Sourav Bhowmick

https://personal.ntu.edu.sg/assourav/

Welcome to ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today I have here with me Sourav Bhowmick, who is a professor at Nanyang Technological University in Singapore, better known as NTU. Sourav is an ACM Distinguished Member. He received the VLDB Service Award, and a Lecturer of the Year Award from NTU. Sourav is also a member of the Initiative on Diversity and Inclusion in Database Conference Venues. His PhD is from NTU. So, Sourav, welcome!

Thank you very much for inviting me, Marianne. I'm really honored to be in this set of interviews.

You've worked in several classical areas of data research, but also in an unusual one: human data interaction. Can you tell me about that?

I explore the interaction of users and data through a visual interface. We database folks love query languages. We have invented query languages for all sorts of different data types – we started with SQL and now we have graph query languages. But most domain experts don't use these languages. Many do not want to write any queries using query languages: they prefer to draw or to have a very intuitive natural language-based interaction. That's why visual interfaces for queries are very powerful for many domain experts.

Visual query interfaces offer a lot of interesting research opportunities which we haven't explored in the past. For example, twenty years ago I was working on XML because a lot of biological data was in XML format. If the user queries this XML data by drawing a tree structure, we get the benefits of the query being revealed incrementally to the query engine, because the user is drawing the query incrementally. We also get the benefit of the latency of a human drawing on an interface, and the research community had never explored how to leverage that. Our paradigm was that the query engine was not going to do anything until it got the complete query, and then it would find the best way of processing the query.

In a scenario where the queries are getting expressed incrementally as the user draws them, we have the opportunity to partially process these queries while the user is drawing, by exploiting the latency of the human interaction with the user interface. This allows us to integrate query processing and query formulation much more tightly than was possible before, opening up new opportunities such as efficient query feedback and query response time. But instead of doing that, we were still slapping a visual query interface on top of a database and that's it: the query engine still waits for the entire query to be formulated before it starts processing.

Our first paper about how to change this was a short paper in ICDE 2006[1], long before people were thinking about what we now call *exploiting think time*. That's become a buzzword nowadays, for interactive exploration, but back then, it was hard to get the paper accepted. The reviewers were saying that this is HCI related, so hey, don't submit it to database conferences! As if the usability of data systems were not important! But that's how we database researchers used to view our own field: we were working under the hood, and user interfaces, how the users search or query, is not our problem. That really limited the usability of database systems. I'm glad that that attitude has changed dramatically in the past decade.

> ***If you pick up any […] textbook, its authors will use diagrams to explain the concept of graphs to you. […] But when we are using a query language to search a particular graph data source, we are asking the query writers to write their query in text form, even though that is not intuitive for humans.***

After getting started with human data interaction research through XML we moved on to graphs, where we worked on blending visual graph query formulation with processing. And then we moved on to the idea of creating visual query interfaces in a data driven manner. That means that the user gives us a dataset and then we give them a visual interface for that data, automatically generating it for them. We did this for graph data.

The reason we focus a lot on graphs is because graphs are very intuitive to draw. If you pick up any data structures textbook or any algorithms textbook, its authors will use diagrams to explain the concept of graphs to you. They don't want to describe the graph in text form. But when we are using a query language to search a particular graph data source, we are asking the query writers to write their query in text form, even though that is not intuitive for humans. In fact, humans using drawing to express themselves predates any form of textual expressions.

Suppose that the user has a graph data source and I give them two hypothetical options. Option 1 is that I

[1] Sourav S. Bhowmick, Sandeep Prakash: Every Click You Make, I Will Be Fetching It: Efficient XML Query Processing in RDMS Using GUI-driven Prefetching. ICDE 2006: 152.

give them a box with hundreds of knobs to turn, and I tell the user, "Hey, this is the fastest data processing system you can ever have. Your every query can be answered in constant time. But you figure out how to write your query by tuning these knobs." Option 2 is that I give the user a nice visual interface, and I tell the user, "Hey, I'm not always going to give you every answer very fast, but you don't need to learn query languages. You can just draw your query intuitively, the way you see it, and search." Which one do you think a domain expert will prefer? That's why I believe there's a lot of potential for graph queries to be visual in nature: it's much more intuitive to visualize a structure than to write it down in text form.

What's the state of the art for visual queries today?

Now we know how to process a set of subgraph search queries by interleaving query formulation and query processing. We have a series of papers in SIGMOD[2] that describe how to do this.

We also know how to create the visual interface for a graph data source automatically for large networks, as well as for a large collection of small or medium size data graphs, like chemical compounds[3].

What are you doing with the Initiative on Diversity and Inclusion in Database Conference Venues?

Recently, I was invited to co-lead its subgroup that is looking at issues in managing conflicts of interest in database conferences.

What's a conflict of interest?

A *conflict of interest* is a set of circumstances that creates the risk that your professional judgment or actions regarding a primary interest will be unduly affected by a secondary interest. That's the definition, independent of any specific area. In our context, the primary interest might be an impartial review of a paper, and you can imagine secondary interests as possible benefits like financial gain, favors for your friends, professional advancement and so on.

The secondary interest becomes objectionable when it is believed to be so large that it may have too much influence on your judgment about the primary interest. Essentially, this is nothing but human bias. Conflict of interest is just one aspect of human bias, and there has been a lot of research from the psychology community showing that conflicts of interest indeed have a serious impact on impartial evaluation of someone's work.

ACM has clearly stated a policy about the kinds of conflicts of interest – COIs – to be avoided for ACM publications[4]. ACM also says that knowingly hiding or falsifying the declaration of COIs is actually a violation of the ACM code of ethics.

For a long time, most database venues have had a COI policy written on their websites. When you submit your paper, you should mark this blah-blah-blah and so on, right?

Yes.

These policies are trying to transform the definition of a conflict of interest into a more clear-cut and actionable definition. Now, the problem lies here: how do you check that authors have indeed reported all their COIs?

Why do you need to check that?

For a long period of time, we didn't have to check it because we were operating based on gentlemen's agreements and trust. But I'm not aware of any human society which is purely based on trust and gentlemen's agreements. Otherwise, we would not need any police!

Sure.

The majority of people are law abiding, and the police are designed for the small minority who are not law abiding.

For a long period of time, our community's handling of conflicts of interest was based on trust. We had our policies, and we believed that the authors will

[2] GBLENDER: Towards Blending Visual Query Formulation and Query Processing in Graph Databases. Changjin Jiu, Sourav S Bhowmick, Xiaokui Xiao, James Cheng, Byron Choi. In SIGMOD, 2010.

QUBLE: Blending Visual Subgraph Query Formulation with Query Processing on Large Networks. Ho Hoang Hung, Sourav S Bhowmick, Ba Quan Truong, Byron Choi, Shuigeng Zhou. In SIGMOD, 2013.

BOOMER: Blending Visual Formulation and Processing of p-Homomorphic Queries on Large Networks. Yinglong Song, Huey Eng Chua, Sourav S Bhowmick, Byron Choi, Shuigeng Zhou. In SIGMOD, 2018.

[3] AURORA: Data-driven Construction of Visual Graph Query Interfaces for Graph Databases. Sourav S Bhowmick, Kai Huang, Huey-Eng Chua, Zifeng Yuan, Byron Choi, Shuigeng Zhou. In SIGMOD, 2020.

CATAPULT: Data-driven Selection of Canned Patterns for Efficient Visual Graph Query Formulation. Huang Kai, Huey Eng Chua, Sourav S Bhowmick, Byron Choi, Shuigeng Zhou. In SIGMOD, 2019.

[4] https://www.acm.org/publications/policies/conflict-of-interest

truthfully and completely declare their COIs. Recently, I wrote a piece of data-driven software called CLOSET[5] that is designed to check if this is really the case. We ran it in a few conferences, initially. In a very recent conference, ICDE 2021, we ran CLOSET post-facto and found a non-negligible number of unreported COIs. So far, in every venue where we have run CLOSET, there is non-empty set of unreported COIs.

> ***You need to try to avoid ethical compromises at all costs, because once you're in, you are in forever.***

What kind of conflicts of interest are you talking about?

Those due to coauthorship or being at the same institution.

In the old days, the computer science research community had small numbers of submissions, small program committees, and they could manually check for COIs: "Hey, this guy should not review that guy's paper" and so on.

But we are in a time where in every data-oriented field, PC sizes are becoming huge. The number of submissions is growing like crazy. It is impossible for a PC chair to manually check whether authors and reviewers have declared their COIs. That's totally out of the realm of possibilities. So, we have our conflict of interest policies, and how do we enforce these policies? We have a huge gap. The implementation can't be manual or based on mutual trust anymore because *the data is showing otherwise*.

So, we need tools. We need data driven tools which will automatically check COIs for our program committee chairs so that they can minimize the bias in the reviews. This will have a downstream effect on the quality of the reviews; on the science which comes out, based on which other people build their science; and essentially, on the healthiness of our scientific community. So it's a very fundamental building block that needs to be put in place.

Now, you'll be amazed to hear this next part. We are the data people, right? And we have a lot of data driven techniques.

Yeah.

On a whole bunch of use cases.

Sure.

If you search for the keyword *data driven* in DBLP, you'll get more than 10,000 hits. We have written more than 10,000 papers on a wide array of issues, but very few on the very scientific framework on which we build our science: *our peer review system*. We data researchers really need to work on this because it has significant downstream impact on many aspects of science.

This is where I got involved, because the data was showing that indeed, there are unreported COIs, and we need to have proper management of this, and proper education. Maybe people do not realize the importance of correctly reporting COIs. And we need to look into possible alternatives for defining COI policies, because our current COI policies are not really capturing existing biases effectively.

What have you been doing to fix those problems?

The CLOSET software that I wrote takes as input the list of submissions and reviewer assignments from any conference management tool, like CMT or EasyChair, and determines whether there are unreported COIs. It detects whether there are program committee members whose review assignments violate the COI policy of a venue, and it does this automatically by analyzing data from several different sources. Then it produces a report for the program committee chairs, saying, "Hey, take a look into these papers. Their review assignments may have COI violations, based on the COI policies of the conference."

The major conference submission review management systems like CMT and EasyChair do have some automated conflict detection functionality now. But I hear from the PC chairs that CLOSET is detecting COIs that have bypassed the conference management system's conflict detection system. We don't know how those algorithms work because their code is not published, but CLOSET is regularly detecting COIs they miss.

While we want to detect undeclared COIs, we also do not want to change the traditional way that authors and reviewers and PC chairs interact with the review management system. We don't want to arm-twist them and ask for a lot of data from them. So, one good thing about CLOSET is that it does not request any additional data from authors and reviewers. It just uses the same kinds of data that authors and reviewers have always provided.

[5] https://personal.ntu.edu.sg/assourav/research/DARE/closet.html

The AI community has taken a different approach, where they ask authors and reviewers for a lot of data, Google Scholar pages and DBLP addresses and all recent collaborators, which can be quite annoying. If I'm a senior author and I have a bunch of collaborators, you're asking me to remember all of them and enter them into the review management system? This is too hard.

Taking too much input from users also increases the chances for the data to be dirty. A PC chair recently told me that they tried to collect Google Scholar and DBLP IDs from reviewers and there were a lot of mistakes. So, if we trust this dirty input data and use it to find COIs, we are bound to have a lot of false negatives.

At the 30,000 feet, the COI detection problem looks deceptively simple: you look up the author's name in, say, DBLP, you look at the co-authors list on that page and if the reviewer's name is there then there's a conflict, and if it's not there then there's no conflict. But once you look deep into the data, you find a lot of issues, and the problem is not that straightforward.

The problem is challenging because, first, the data is dirty. Authors do not always specify their names in the same way that their names are listed in DBLP. Second, even DBLP data is not completely clean, although it is a great data source for us and the cleanest one around. Sometimes DBLP has publications of people assigned to someone who is not actually the correct author, mainly due to homonym problems. Because names are ambiguous, DBLP also has disambiguation pages, where many papers end up because DBLP is not certain which person wrote them. There are more than 270 different computer science researchers named Wei Wang in DBLP!

Maybe someday ORCID IDs will solve the problem, but right now many people don't have an ORCID or don't use it when they write their papers. Certainly they didn't use it on their old papers. Many people have multiple ORCID IDs, too.

Because there are so many data cleaning issues, the problem is much more challenging than you would think. Of course, if we just use a string-matching technique, even regular expression matching and try to look for a name, we will always find some COIs. And that gives us that false security that "hey, I can detect that COI!" But the challenge, the actual issue, is the false negatives. What are we missing? We see that when we run CLOSET and detect some of the COIs which were overlooked by industrial strength review management systems.

You were talking earlier about visual interfaces to data. I understand that you are a visual artist yourself and have even sold your artwork to benefit humanitarian causes. What kind of art do you do?

The topics I choose to paint are often driven by socio-political causes, and one that is close to my heart is refugees. We have millions of refugees worldwide from conflicts and other causes. A few years ago was the peak period of the Syrian crisis, as well as the Myanmar refugee crisis.

One of the refugee crises, the Myanmar crisis, was very close to Singapore. A couple of other painters and I were thinking that Singapore is like a bubble, it's a place like nowhere else. People are very well taken care of by the government here, and people don't worry about the basic things in life. Since there is nothing important to worry about, people instead focus on small issues, relatively speaking, issues that probably other countries will not bother to talk about. Being a refugee is not something you see in Singapore and it does not gather much attention from local people. To educate people here about the refugee problem and to support refugees across the world, my art buddies and I had a series of exhibitions.

In the creative process in art, you start by thinking, looking deep into a problem and start observing things the same way as you do in science. You start with nothing, a blank canvas. You observe what's going on in the same way that we scientists observe our data, and then try to create something out of it. And that's what we also do in science.

The painting above tells the story of a boy named Omran, who was found all covered in dust but still alive, in a place in Syria where the buildings were all destroyed by bombing. In the background of the painting you see the total destruction of the city. The T-shirt he was wearing that day is different from the one in the painting, which shows a particular color combination that is a symbol of peace. It was used in a peace flag back when Iraq was invaded, so this color combination means *give peace a chance*. The painting

is symbolic in that the boy is trying to tell you to give peace a chance, through his T-shirt.

The painting above shows refugees who are fleeing Myanmar by boat. In the painting, even though they're fleeing, you can see that they are together. They have nothing but they are together and trying to help each other, as you can see from how they are trying to pick up other people from the water. Some governments were using helicopters to drop aid packages on the sea for them to pick up, and they are swimming out from the boat to get the aid packages and then going back to the boat.

Up on the right-hand corner, there's a dove, a message of peace. And on the left edge, there's a lady who's looking at the opposite direction to where all the other people are looking. She represents Aung San Suu Kyi, looking away from the problem.

Thank you for showing us those.

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

I don't want our young people to be like Aung San Suu Kyi, looking away from big ethical problems. I recommend that young researchers be ethics centric and ethics first in their approach to research.

But how can you be acting ethically if your advisor is not completely ethical? What should you do?

There will be some people, I hope, in your university who want to support students who are ethically correct. And if that is the case, one approach is to change your supervisor. That is what our students at NTU do if they find that they cannot work with their advisor for various reasons. You need to take that leap of faith and have the courage to remove yourself from any situations like this. You need to try to avoid ethical compromises at all costs, because once you're in, you are in forever. One favor leads to another and then it leads to another and then you have a big problem.

Thank you very much for talking to me today.

It's been a pleasure!

Viktor Leis Speaks Out on Concurrency and Parallelism on Multicore CPUs

Marianne Winslett and Vanessa Braganholo

Viktor Leis

https://www.cs6.tf.fau.de/person/viktor-leis/

Welcome to ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today I have here with me Viktor Leis who won the 2018 ACM SIGMOD Jim Gray Dissertation Award for his thesis entitled Query Processing and Optimization in Modern Database Systems. Viktor is now at the University of University of Erlangen-Nuremberg and his Ph.D. is from the Technical University of Munich, where he worked with Thomas Neumann and Alfons Kemper. So, Viktor, welcome.

Can you tell me what your thesis is about?

Sure! My thesis is in the context of the HyPer system, which is a main memory database system, which we've been building at TU Munich. And in my thesis, specifically, I looked at a couple of different aspects. One big theme is really how do you exploit multicore CPUs. So, a big part of the thesis is how do you do that for concurrency control, for index structures, but also on query processing. How do you parallelize queries, so that you really exploit all these many core CPUs that you have nowadays? And so, in the thesis, you find lots of different techniques that really make it possible to get really good performance and really good speed-ups on modern CPUs.

Nowadays, almost any server CPU has dozens of cores. And so, that's really what I would say is the main focus of my thesis. It also has a chapter on query optimization. But I would say, if there was really one big thing, it's really concurrency and parallelism on multicore CPUs.

So, what was the state of the art like before you did this? Why was this work needed?

So, at that time – and this is also how HyPer started – the initial idea was that in traditional systems you have lots of locks and latches, and this really doesn't scale. And so, the idea was you have to get rid of all of them, and you have to partition your data. And you also have to partition your workload and that obviously can give you very good performance if your workload and your data are actually partitionable. That may work very well for microbenchmarks and for many macro benchmarks that we use in research. But I'm a little bit skeptical: in the real world, a lot of workloads are very messy, very complicated, and it's very hard to find a partitioning that will make this thing actually work. Another way of saying it is that if your workload is partitionable, you can just run five different instances of your database. That would work as well. That's all.

Yeah.

That's, in a way, a little bit cheating. And so, I think 10 years ago, it was not really clear if you could actually achieve very good synchronization without partitioning. And I think now we can say that we can actually do that.

So, what kind of speed-ups do you get now compared to before with your techniques?

So, for the intra-query parallelization: you have one query that scans lots of data, does lots of joins, and so on. Actually, my goal is always if you have a system with 50 CPU cores, then you want a speed up close to 50. Now, you don't always get it, right, but this not totally unrealistic. For most queries, you will actually get that. And to achieve that actually requires a lot of work because what you see at these kinds of speed-ups is that Amdahl's law really kicks in. Because any small part of your query that doesn't scale, that is not parallelized, means you will not get that speed-up of 50.

So, it requires a lot of work, and every single operator needs to be parallelized. A lot of these operators are described in my thesis. And not just joins, but also aggregation, sorting, and window functions. All these operators. And I think there's a lot of work required, both in terms of algorithms and in terms of a lot of low-level techniques.

What kind of impact do you see your thesis having in industry?

> ***[…] pick the right advisor. And they must be someone that you respect […] technically, scientifically […]***

So, my thesis was about the system HyPer. We made a startup, and pretty quickly after my graduation it was acquired by Tableau. And they then put it into their data visualization products. So, if you now click around in Tableau, then it will use HyPer behind the scenes. So, it's actually pretty cool! Particularly, the code of the query parallelization and the query engine that I was talking about is actually now used every day by hundreds of thousands of people. I don't know the exact number, but that's something I am really proud of.

That's really cool. I'm so glad to hear that. Do you have any words of advice for today's graduate students?

Sure. And I should say that, nowadays, you hear a lot of scary stories about how terrible a Ph.D. is. I actually had the best time ever in my Ph.D. But I think that depends mostly on your thesis advisor. So, my main advice, basically, is to pick the right advisor. And they must be someone that you respect, like technically, scientifically, their skills, and also what they're interested in. I think that it's very important that these things are aligned with your interests, and also on a personal level. A Ph.D. takes a long time, and it's also a very personal experience, and so, that is important.

So, I think that's maybe the most important thing. It's probably much more important than whether it's a very fancy school or something in comparison with somebody's who's maybe not in an as fancy university. But if you have a good relationship and you respect the person, that will go a very long way. So, I think that that's a really big thing.

The second piece of advice I could give is – actually, I did my Ph.D. like five years ago, so now I also have a lot of experience on the other side. And so, one thing that I see from a lot of students is, when they have an idea or did something and it's reasonably okay, they immediately want to publish it. They think that's a great thing. I think there's a risk in that because not everything that you can publish – and it turns out you can actually publish everything if you spend a lot of effort there – should be published in the sense that it's much better to try a couple of things, and then pick the best one and then spend that half a year or whatever it takes to polish the paper and so on. I think that's a much better approach.

And I think paradoxically, this can lead to you publishing more work. You will have stronger results that are easier to publish. It will actually make you happy if you believe in the stuff that you are working on. And so, it's pretty easy to do a couple of prototypes. Maybe work on something for a month, and see, "is that just an okay idea or is it actually really good?" It's much better to reiterate and do a couple of prototypes. So, I think that's something more people should do and not just decide on one topic and then fight it through. Because you can do it, but it's just not a lot of fun. So, that would be a second thing I can recommend.

Thank you very much for talking to me today.

Thank you.

Information and Research Science connecting to Digital and Library Science

Report on the 17th Italian Research Conference on Digital Libraries, IRCDL2021

Dennis Dosso
University of Padova
Padua, Italy
dennis.dosso@unipd.it

Stefano Ferilli
University of Bari
Bari, Italy
stefano.ferilli@uniba.it

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

Antonella Poggi
"La Sapienza", University of Rome
Rome, Italy
antonella.poggi@uniroma1.it

Giuseppe Serra
University of Udine
Udine, Italy
giuseppe.serra@uniud.it

Gianmaria Silvello
University of Padova
Padua, Italy
gianmaria.silvello@unipd.it

1. INTRODUCTION

Since 2005 the Italian Research Conference on Digital Libraries is a yearly date for researchers on Digital Libraries and related topics, organized by the Italian Research Community. Over the years, IRCDL has become an essential national forum focused on digital libraries and associated technical, practical, and social issues. IRCDL encompasses the many meanings of the term *digital libraries*, including new forms of information institutions; operational information systems with all manner of digital content; new means of selecting, collecting, organizing, and distributing digital content; and theoretical models of information media, including document genres and electronic publishing.

Digital libraries may be viewed as a new form of information institution or an extension of the services libraries currently provide. Representatives from academia, government, industry, research communities, and research infrastructures, are usually invited to participate in this annual conference. The conference draws from a broad and multidisciplinary research area, including computer science, information science, librarianship, archival science and practice, museum studies and practice, technology, social sciences, cultural heritage and digital humanities.

This year the focus was bridging the broad field of Research and Information Science with the related field of Digital Libraries. Indeed, IRCDL2021 aimed at approaching "Digital libraries" by embracing the field at large, comprehending three critical areas of interest, that are *scholarly communication* (e.g. research data, research software, digital experiments, digital libraries), *e-science/computationally-intense research* (e.g. scientific workflows, Virtual Research Environments, reproducibility) and *library, archive and information science* (e.g. governance, policies, open access, open science). There were a total of 25 submissions, and 17 of these were accepted: 11 full papers, and 6 short papers. The overall aim of IRCDL is to help young researchers to move the first steps in the competitive research world. For this reason, weak papers are shepherded and a thorough feedback is provided to the authors before possibly leading to acceptance.

The conference has been organized in Padua (Italy) at the Department of Information Engineering of the University of Padua. 145 participants attended the conference, coming coming for the most part from Italy, and also from Switzerland, Sweden, Denmark, Russia, Brasil, Germany and the UK. Participants are mainly from academia, but also culture heritage related industry and institutions (e.g., libraries and archives). Due to the COVID-19 emergency, the conference has been entirely online on 18 and 19 February 2021. The proceedings are published in the CEUR-WS Vol-2816 `http://ceur-ws.org/Vol-2816/`. The papers are available in gold open access, and all the videos of the presentations are available on YouTube and accessible via the conference Website: `http://ircdl2021.dei.unipd.it/`

2. CONFERENCE CONTRIBUTIONS

All the contributions were peer-reviewed by three of the twenty-two members of the Program Committee, and eighteen were accepted, out of which seven were short papers. IRCDL 2021 [8] featured one invited speaker and six thematic sessions devoted to the presentation of the accepted research papers.

Invited talk: The swings and roundabouts of a decade of fun and games with Research Objects

Prof. Carole Goble's keynote[1] focused on the concept of Research Objects[2][4]. We are transitioning toward the fourth paradigm of science, a concept based on the idea that computational science constitutes a new set of methods beyond empiricism, theory, and simulation. It requires the integration between tools, technologies, and platforms in shared methodologies and processes, also designing methods for researchers to share their data. It allows the integration of these tools and technologies, creating new research opportunities not available before [15]. In this paradigm research outcomes are more than publications and data, but include software, models, workflows, SOPs, lab protocols, etc. All these products of science should be considered as first class citizens of scholarship, and be treated following the FAIR and Reproducible principles (FAIR+R). The Research Object is a general framework, where research outcomes are related and bundled together, to form a single reusable and citable object in support of reproducibility. RO's metadata describes the attribution, reuse, relational properties of the bundled objects, to ensure their FAIR+Rness. These resources thus become shareable, citable, exchangeable across different platforms, present versioning and snapshots, can be identified with tools such as PIDs and can be registered and deposited on their own in services, thus to be later unpackaged, accessed, activated, and reproduced if appropriate.

Data and Platforms

This section presented new platform to manage and share data, and shed light on important challenges faced by already existing and widely-used data infrastructures. Biasini et al. [7] presented FullBrain[3], a social e-learning platform where students can share and track their knowledge, helping each other in their learning process. Gargiulo et al. [13] noted that the FAIR RDM initiatives in Italy are still based on communities of practice, thus investigate the perception of some leader initiative with respect to good practices, challenges and the strategic vision. Baglioni et al. [3] highlighted the "service misuses" suffered by the ORCID infrastructure, that put in jeopardy its very mission.

Data Access and Monitoring

This section presented new systems and techniques based on data and how its correct use can have meaningful impacts on different aspects of people's everyday life, ranging from learning to security.

Zanichelli et al. [21] noted that digital games could be a valuable tool for enhancing learning and aim to understand how to improve the assessment of learning obtained through them. Avanzi et al. [2] presented *NoBis*, a new service for monitoring the crowding of indoor spaces during the COVID-19 era. Spadi et al. [19] presented the first results of the project RESTORE, whose main goal is the recovery, integration, and accessibility of data and digital objects produced in the last twenty years by the partners of the project, thus building a knowledge base on the history of the town of Prato (Tuscany, Italy).

Information and Knowledge

This section presented new applications and measures to obtain and evaluate information. Irrera and Silvello [16] addressed the problem of finding context information for news articles by extracting entities and relationships from documents. The idea is to provide context to documents and improve learning to rank methods' effectiveness. Lancioni et al. [17] use Generative Adversarial Networks to predict keyphrases, i.e., short text sequences that convey the main semantic meaning of a document. Ferrante et al. [12] extend the AWARE measure with a set of supervised methods, by using TREC collections.

Character Recognition

This section presented new approaches to address problems connected to the automatic recognition of handwritten text to uncover texts still unreadable and provide new and meaningful tools to the experts. Fante and Di Nunzio [10] proposed a mixed qualitative-quantitative approach to OCR error detection and correction to develop a methodology for compiling historical corpora. Heil et al. [14] introduced Handwritten Text Recognition and Document Image Analysis approaches to address the challenges inherent to the original drafts of Astring

[1] https://www.slideshare.net/carolegoble/the-swings-and-roundabouts-of-a-decade-of-fun-and-games-with-research-objects-242974091
[2] http://researchobject.org/
[3] https://fullbrain.org/

Lindren, a Swedish author whose original drafts and manuscripts are edited in the Melin system of shorthands, as of today considered undecipherable. Marinai et al. [18] described a system for the location of simple graphemes in medieval manuscripts based on the Mask R-CNN convolutional neural network. A first approach to provide paleographers with a tool to speed up and refine dating and determining the origin of manuscripts.

Text Analysis

This section explores new potential applications connected to the analysis of documents, opening up new applications and research lines. The work proposed by Bernasconi et al. [6] explored the idea of navigating the semantic relationships among extracted entities to search a text corpus, and the evaluation carried with potential users has shown the feasibility and effectiveness of the approach. Ferilli [11] presented a paper which deals with architectural floorplans, proposing an approach based on formal representation and reasoning for their understanding and interpretation, opening up a viable and promising new line of research. Dosso and Silvello [9] proposed a new problem, called Data Credit Distribution, to annotate data in a database with a value, called credit, representing the impact and importance of this data in the scientific domain.

DL Services

Bernasconi et al. [5] presented the Knowledge Graph they developed within the ERC NOTAE project[4] and showed how navigating such graph can help researchers at finding non trivial implications in data, providing them support in investigating graphic symbols dating back to the Roman State and Post-Roman Kingdoms. Svarre [20] studied labels used by Danish academic library websites that can support user interactions with library websites and their related content, shedding more light on the characteristics and variety of labels used across libraries. Almeida et al. [1] are developing the ROSSIO Thesaurus, whose objective is to support content discovery and management in the ROSSIO Portuguese research infrastructure for arts and humanities, and describe its modeling process and integration in the infrastructure, together with its publication as LOD.

3. CONCLUSION AND PROSPECT

The world of DLs is going through profound transformations, and the research activities and results presented at IRCDL2021 gives a clear indication of how multifaceted Digital Library research is.

A panel of experts was organized to continue the dialogue started in the last years to discuss DL's next future. The purpose is to enlarge DL's scope and open IRCDL to the challenges to come.

The panel opened with an introduction by Gianmaria Silvello, who reported the panel's outcomes at IRCDL 2020 about the future of IRCDL. In 2020 the panel concluded that IRCDL has a central role for the Italian DL community and is a reference point for the young researchers, who, thanks to this conference, can move the first steps in research in a friendly and open environment. Nevertheless, there is the need to update the topics of the conference and to open-up to new emerging fields while maintaining the roots: i) IRCDL is an Italian conference with an international view; thus, it can be organized by international researchers and in English, but it has to maintain its national connotation; ii) cultural heritage, the humanities and library science are essential topics for IRCDL that have to keep a central role in the conference; iii) IRCDL maintains its focus on students and young researchers.

This year's panel discussed the challenges for the DL community in an evolving world and how IRCDL can adapt and be a catalyst for these changes. There were five panelists. The first panelist was Nicola Ferro (University of Padua), who discussed the role of Computer Science and Engineering research in DL. The second was Julian Bogdani (University of Rome - Sapienza), who reported his experience working in the PAThs project [5], that is an ERC interdisciplinary project involving archaeology and computer science. Stefania Gialdroni (University of Roma3) further discussed the intersections between cultural heritage and computing, by reporting her experiences in the field of legal history and depicting the objectives of her future work within the ERC MICOLL project. Luigi Siciliano (Free University of Bozen-Bolzano) dug on digital libraries' role in the ever-changing world of information access; he analyzed prominent vendors and software houses and how they impact the management and access to knowledge. Finally, Donato Malerba (University of Bari) presented the many contact points between Data Science and Digital Libraries, opening up to future collaborations between these two fields.

The next edition of IRCDL will be held in 2022 in Italy. IRCDL2021 consolidated the Italian community presence and widened international participation opening up to authors from all around Europe.

[4] http://www.notae-project.eu/

[5] http://paths.uniroma1.it/

Our goal for IRCDL2022 is to further open up to international involvement to shape DL's future.

Acknowledgments

We desire to thank all those who contributed to the event, especially the advisory board [6] for their advice and support. Special thanks to the members of the program committee whose research experience largely contributed to making this conference an attractive venue and valuable experience. Our sincere gratitude to all participants, invited speakers and authors, whose enthusiasm and vision constitute the soul of this conference.

4. REFERENCES

[1] Almeida, B., Freire, N., and Monteiro, D. The development of the ROSSIO thesaurus: Supporting content discovery and management in a research infrastructure. In Dosso et al. [8], pp. 138–146.

[2] Avanzi, M., Coniglio, R., Cisotto, G., Giordani, M., and Ferro, N. Nobis: A crowd monitoring service against COVID-19. In Dosso et al. [8], pp. 126–137.

[3] Baglioni, M., Mannocci, A., Manghi, P., Atzori, C., Bardi, A., and Bruzzo, S. L. Reflections on the misuses of ORCID ids. In Dosso et al. [8], pp. 117–125.

[4] Bechhofer, S., Buchan, I., De Roure, D., Missier, P., Ainsworth, J., Bhagat, J., Couch, P., Cruickshank, D., Delderfield, M., Dunlop, I., et al. Why linked data is not enough for scientists. In *Future Generation Computer Systems* [8], pp. 599–611.

[5] Bernasconi, E., Boccuzzi, M., Catarci, T., Ceriani, M., Ghignoli, A., Leotta, F., Mecella, M., Monte, A., Sietis, N., Veneruso, S., and Ziran, Z. Exploring the historical context of graphic symbols: the NOTAE knowledge graph and its visual interface. In Dosso et al. [8], pp. 147–154.

[6] Bernasconi, E., Ceriani, M., and Mecella, M. Exploring a text corpus via a knowledge graph. In Dosso et al. [8], pp. 91–102.

[7] Biasini, M., Carmignani, V., Ferro, N., Filianos, P., Maistro, M., and Nunzio, G. M. D. Fullbrain: a social e-learning platform. In Dosso et al. [8], pp. 25–41.

[8] Dosso, D., Ferilli, S., Manghi, P., Poggi, A., Serra, G., and Silvello, G., Eds. *Proceedings of the 17th Italian Research Conference on Digital Libraries, Padua, Italy (virtual event due to the Covid-19 pandemic), February 18-19, 2021* (2021), vol. 2816 of *CEUR Workshop Proceedings*, CEUR-WS.org.

[9] Dosso, D., and Silvello, G. Data credit distribution through lineage. In Dosso et al. [8], pp. 155–161.

[10] Fante, D. D., and di Nunzio, G. M. A quantitative/qualitative approach to OCR error detection and correction in old newspapers for corpus-assisted discourse studies. In Dosso et al. [8], pp. 53–63.

[11] Ferilli, S. Toward automatic floor plan interpretation. In Dosso et al. [8], pp. 13–24.

[12] Ferrante, M., Ferro, N., and Piazzon, L. s-aware: Using crowd judgements in supervised measure-based methods for IR evaluation. In Dosso et al. [8], pp. 162–168.

[13] Gargiulo, P., Galimberti, P., Tammaro, A. M., and Zane, A. FAIR RDM (research data management): Italian initiatives towards EOSC implementation. In Dosso et al. [8], pp. 42–52.

[14] Heil, R., Nauwerck, M., and Hast, A. Shorthand secrets: Deciphering astrid lindgren's stenographed drafts with HTR methods. In Dosso et al. [8], pp. 169–177.

[15] Hey, T., Tansley, S., and Tolle, K. M., Eds. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009.

[16] Irrera, O., and Silvello, G. Background linking: Joining entity linking with learning to rank models. In Dosso et al. [8], pp. 64–77.

[17] Lancioni, G., Mohamed, S. S., Portelli, B., Serra, G., and Tasso, C. Efficient keyphrase generation with gans. In Dosso et al. [8], pp. 1–12.

[18] Marinai, S., Pomaro, G., Raffaelli, C., and Scandiffio, F. Location of simple graphemes in mediaeval manuscripts based on mask R-CNN. In Dosso et al. [8], pp. 103–116.

[19] Spadi, A., Spinelli, F., and Degl'Innocenti, E. RESTORE: smart access to digital heritage and memory. In Dosso et al. [8], pp. 178–185.

[20] Svarre, T. Labelling on academic library websites. In Dosso et al. [8], pp. 186–193.

[21] Zanichelli, F., Encheva, M., Thabet, B., Tammaro, A. M., and Conti, G. Serious games for information literacy: Assessing learning in the NAVIGATE project. In Dosso et al. [8], pp. 78–90.

[6] http://ims.dei.unipd.it/websites/ircdl/home.html