Report from Dagstuhl Seminar 19491

Big Graph Processing Systems

Edited by

Angela Bonifati[1], Alexandru Iosup[2], Sherif Sakr[3], and Hannes Voigt[4]

1 **University Claude Bernard – Lyon, FR,** angela.bonifati@univ-lyon1.fr
2 **VU University Amsterdam, NL,** alexandru.iosup@gmail.com
3 **University of Tartu, EE,** sherif.sakr@ut.ee
4 **Neo4j – Leipzig, DE,** hannes.voigt@neo4j.com

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 19491 "Big Graph Processing Systems". We are just beginning to understand the role graph processing could play in our society. Data is not just getting bigger, but, crucially, also more connected. Exploring, describing, predicting, and explaining real- and digital-world phenomena is increasingly relying on abstractions that can express interconnectedness. Graphs are such an abstraction. They can model naturally the complex relationships, interactions, and interdependencies between objects. However, after initial success, graph processing systems are struggling to cope with the new scale, diversity, and other real-world needs. The Dagstuhl Seminar 19491 aims to addresses the question: How could the next decade look like for graph processing systems?

To identify the opportunities and challenges of graph processing systems over the next decade, we met in December 2019 with circa 40 high-quality and diverse researchers for the Dagstuhl Seminar on Big Graph Processing Systems. A main strength of this seminar is the combination of the data management and large-scale systems communities. The seminar was successful, and addressed in particular topics around graph processing systems: ecosystems, abstractions and other fundamental theory, and performance.

Seminar December 1–6, 2019 – http://www.dagstuhl.de/19491
2012 ACM Subject Classification Computer systems organization → Architectures, Information systems → Data management systems, Software and its engineering → Distributed systems organizing principles
Keywords and phrases Abstractions, Big Data, Big Graph, data management, Ecosystems, graph processing, Performance, systems, Theory
Digital Object Identifier 10.4230/DagRep.9.12.1

1 Executive Summary

Alexandru Iosup (VU University Amsterdam, NL)
Angela Bonifati (University Claude Bernard – Lyon, FR)
Sherif Sakr (University of Tartu, EE)
Hannes Voigt (Neo4j – Leipzig, DE)

In memoriam: This seminar is dedicated to the memory of our co-organizer and friend Sherif Sakr (1979-2020), whose unexpected early departure happened a few months after the seminar. Sherif was a leading scientist in the field of Big Data Technologies. We are grateful to him for the time spent together and the joint work preceding and following the seminar. He will be deeply missed.

The world has become more interconnected than ever. Through an advancing wave of technologies and applications, our society is producing and consuming data at an unprecedented scale and complexity. To model the data, graphs offer a general model and mathematical abstraction, in the simplest form based on arbitrary objects (vertices) connected by relationships (edges), with possibly additional information (properties[1]). Graphs enable already a remarkable range of application domains[2], from industry to science, from society to governance, from education to gaming, but their true potential is just beginning to be unlocked. However, the tremendous increase in the size, complexity, and diversity of the graph-structured data and their applications, and the increasing community using graphs to understand and automate the world around us, raises new challenges for computer science. Under these new circumstances, the potential benefits of graph processing could be canceled by the difficulty to understand, create, develop, and automate graph processing for the masses. Focusing on the interplay between graph data, abstractions, systems, performance engineering, and software engineering, this seminar brings together researchers, developers, and practitioners actively working on this topic, to discuss *timely and relevant* open challenges with a main focus on the following topics: trade-off of design decisions of big graph processing systems, high-level graph programming abstractions and graph query languages, the specific requirements for different application domains for benchmarking and graph engineering purposes, systems and ecosystems for graph processing, the fundamental processes and methods leading to the science, design, and engineering of graph processing.

The seminar focused on the following key topics related to big graph processing systems:

Topic 1. Design Decisions of Big Graph Processing Ecosystems: In modern setups, graph-processing is not a self-sustained, independent activity, but rather part of a larger big-data processing ecosystem. Typical examples include the Giraph's deployment in the Facebook MapReduce ecosystem[3], Powergraph[4] in the GraphLab[5] machine learning and data-mining ecosystem, and GraphX[6] in the Apache Spark ecosystem. In general, more alternatives usually mean harder decisions for choice. In practice, with the wide spectrum of big graph processing systems, with different design decisions, that are currently available, it becomes very challenging to decide by intuition which system is the most adequate for a given application requirements, workload, or the underlying ecosystem. Making such decisions requires significant knowledge about the graph complexity, graph size, world requirements, and even the implementation details of the various available systems. Currently, we lack the fundamental models to understand and quantitatively analyze, estimate, and describe the complexity of big graph processing jobs. In addition, there is no understanding on the relationship between the graph complexity and the computational complexity of big graph processing jobs. Therefore, we need a clear understanding for the impact and the trade-offs of the various decisions (e.g., centralized vs distributed, partitioning strategy, programming model, graph representation model, memory storage vs disk storage) in order to effectively guide the developers of big graph processing applications.

[1] M. Junghanns et al., "Analyzing Extended Property Graphs with Apache Flink," NDA'16
[2] L. da Fontoura Costa et al. (2008) Analyzing and Modeling Real-World Phenomena with Complex Networks: A Survey of Applications, ArXiv Physics and Society, 2008. https://arxiv.org/abs/0711.3199v3 This study identifies tens of application domains for graph processing.
[3] Ching et al., One Trillion Edges: Graph Processing at Facebook-Scale, VLDB '15.
[4] Gonzalez et al., PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs, OSDI '12.
[5] Low et al., Distributed GraphLab: a framework for machine learning and data mining in the cloud, VLDB '12.
[6] Gonzalez et al., GraphX: Graph Processing in a Distributed Dataflow Framework., OSDI '12.

Topic 2. High-Level Graph Processing Abstractions: While imperative programming models, such as *vertex-centric* or *edge-centric* programming models, are popular, they are lacking a high-level exposition to the end user. This way the end user is required more technical programming, which limits the end user productivity in building graph processing pipelines. In contrast, graph query languages build on more high-level, declarative constructs. Query language abstraction give more power to the less technical user and allow for extensive performance optimization by the underlying graph processing system. Current graph query languages, however, lack the power required in many graph analytics use cases. To increase the power of graph processing systems and foster the usage of graph analytics in applications, we need to design high-level graph processing abstractions. It is currently completely open how future declarative graph processing abstractions could look like, which the best level of abstraction is, how abstraction for analytics integrate with existing graph query languages, and we can evaluate new graph processing abstractions regarding utility, simplicity, expressiveness, and optimization potential.

Topic 3. Performance and Scalability Evaluation: Traditionally, performance and scalability are measures of efficiency, contrasting the ability of systems to utilize resources: FLOPS, throughput (e.g., EVPS), or speedup (i.e., compared to either a single-node, or a sequential implementation). Such metrics are difficult to apply for graph processing, especially since performance is non-trivially dependent on *platform*, *algorithm*, and *dataset* (i.e., the PAD triangle[7]). Therefore, many important questions arise: *how to compare the performance of graph-processing systems?, how to define scalability?, should one compare largely different systems, e.g., a distributed, heterogeneous system with a highly-tuned, hand-written sequential implementation?, how to design a framework for reproducible performance evaluation?*. Moreover, running graph-processing workloads in the cloud leverages additional challenges. First, we would like to understand whether the intrinsic cloud elasticity could be harnessed for graph processing. Second, clouds are known to be impacted by large degrees of performance variability due to colocation and virtualization overheads. Studying the impact of cloud performance variability onto graph-processing workloads is another topic of interest. Such performance-related issues are key to identify, design, and build upon widely recognized benchmarks for graph processing.

For each topic, the discussion also considered specific and general applications of graph processing, at various volume, velocity, and other dimensions.

The seminar brought together over 40 diverse and high quality researchers with core expertise from two generally distinct communities, data management and (large-scale) computer systems. The seminar was successful, and addressed in particular topics around graph processing systems: ecosystems, abstractions and other fundamental theory, and performance. To this end, we structured the seminar as follows:

1. Prior to the seminar, the co-organizers have contacted each participant, eliciting commitment for one or several topics, and ideas for key elements of the discussion.
2. During the first day of the seminar, the morning was dedicated to short presentations by each participant, and a long break-out session per topic. The former allowed the participants to better understand each other's core ideas and keywords, to identify synergies and to find experts for keywords not entirely familiar.
3. For the next two days, each morning challenged at least one half the participants with

[7] Guo et al., How well do graph-processing platforms perform? an empirical performance evaluation and analysis, IPDPS '14.

a tutorial given by a leading expert from the *other* community, then proceeded with break-out sessions organized per topic, and ended with a plenary session to share the main ideas. The tutorials were given by Tamer Özsu on "Graph Processing: A Panoramic View and Some Open Problems", on behalf of the data management community, and by Antonino Tumeo on "Big Graph Processing: The System Perspective", on behalf of the systems community. The main results of these two days of intense work were making terminology more uniform across the participants, and the core ideas about challenges (open problems), directions for long-term research, and identification of concrete short-term plans for continuation.

4. During the last day of the seminar, the participants finalized the immediate conclusions of the seminar (see Section "In Conclusion: Challenges and Future Directions for Big Graph Processing Systems"), and agreed on the plans for continuation.

2 Table of Contents

3 Overview of Talks

3.1 Graph Applications Focusing on Legal Cases at the Thomson Reuters Lab

Khaled Ammar (Thomson Reuters Labs, CA)

This talk presented work at Thomson Reuters Lab focusing on legal cases, around the question of "How to identify implied overruling for cited and citing legal cases?"

3.2 G-Core and Path Databases

Renzo Angles (University of Talca – Chile, CL)

Joint work of Renzo Angles, Marcelo Arenas, Pablo Barceló, Peter A. Boncz, George H. L. Fletcher, Claudio Gutierrez, Tobias Lindaaker, Marcus Paradies, Stefan Plantikow, Juan F. Sequeda, Oskar van Rest, Hannes Voigt
Main reference Renzo Angles, Marcelo Arenas, Pablo Barceló, Peter A. Boncz, George H. L. Fletcher, Claudio Gutierrez, Tobias Lindaaker, Marcus Paradies, Stefan Plantikow, Juan F. Sequeda, Oskar van Rest, Hannes Voigt: "G-CORE: A Core for Future Graph Query Languages", in Proc. of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018, pp. 1421–1432, ACM, 2018.
URL https://doi.org/10.1145/3183713.3190654

We present our current research around G-CORE, a proposal of query language for property graph databases. G-CORE was presented in SIGMOD'2018, and currently we are working in its implementation on top of Apache Spark. A novel feature of G-CORE is the use of paths as first class citizens. On this line, our current research concerns the notion of path database and the definition of operations for manipulating paths.

3.3 Native-Graph Native-Relational Query Support: The G+R (GRFusion) Approach

Walid Aref (Purdue University – West Lafayette, US)

Joint work of Mohamed S. Hassan, Tatiana Kuznetsova, Hyunchai Jeong, Walid G. Aref, Mohammad Sadoghi
Main reference Mohamed S. Hassan, Tatiana Kuznetsova, Hyun Chai Jeong, Walid G. Aref, Mohammad Sadoghi: "Extending In-Memory Relational Database Engines with Native Graph Support", in Proc. of the 21th International Conference on Extending Database Technology, EDBT 2018, Vienna, Austria, March 26-29, 2018, pp. 25–36, OpenProceedings.org, 2018.
URL https://doi.org/10.5441/002/edbt.2018.04

Graphs with relational node and edge schemas are popular. It is natural to use a relational system to support such graphs. However, this affects the performance of graph operations. In this talk, I will present the G+R Fusion Approach, where we can (1) seamlessly integrate graph and relational data, (2) natively process graph operations and natively process relational operations, (3) support declarative SQL-based graph and relational query formulation, (4) conceptually model a graph in SQL as a stream of vertexes, edges, paths, or subgraphs, (5) support complex graph operations via hints within SQL, (6) provide an API for programming complex graph operations from within SQL, (7) support graph as an in-memory index inside a relational engine with g2r and r2g pointers to facilitate the navigation between the relational and graph sides, (8) isolate graph topology from graph annotations and form graph index only on the topology, (9) introduce graph operations that co-occur in a relational query evaluation pipeline, e.g., vertexScan, edgeScan, and pathScan, (10) support complex queries that involve combinations of relational and graph operations both in native execution modes. The talk will report on in-progress work in support of Graph Views, SQL-based Graph-to-Graph transformations, and support for graphs with multiple node and edge types.

3.4 Understanding and Accelerating Graph Processing, Storage, and Analytics

Maciej Besta (ETH Zürich, CH)

Joint work of Maciej Besta, Torsten Hoefler, Marc Fischer, Tal Ben-Nun, Johannes de Fine Licht, Simon Weber, Lukas Gianinazzi, Robert Gerstenberger, Andrey Ivanov, Yishai Oltchik, Dimitri Stanojevic
Main reference Maciej Besta, Michal Podstawski, Linus Groner, Edgar Solomonik, Torsten Hoefler: "To Push or To Pull: On Reducing Communication and Synchronization in Graph Computations", in Proc. of the 26th International Symposium on High-Performance Parallel and Distributed Computing, HPDC 2017, Washington, DC, USA, June 26-30, 2017, pp. 93–104, ACM, 2017.
URL https://doi.org/10.1145/3078597.3078616

Combining theory and practice for understanding and accelerating graph processing, storage, and analytics, on all levels of the computing stack.

3.5 On the Linked Data Benchmark Council and Current Research

Peter A. Boncz (CWI – Amsterdam, NL)

Joint work of Peter A. Boncz, Renzo Angles, János Benjamin Antal, Alex Averbuch, Peter A. Boncz, Orri Erling, Andrey Gubichev, Vlad Haprian, Moritz Kaufmann, Josep-Lluís Larriba-Pey, Norbert Martínez-Bazan, József Marton, Marcus Paradies, Minh-Duc Pham, Arnau Prat-Pére
Main reference Renzo Angles, János Benjamin Antal, Alex Averbuch, Peter A. Boncz, Orri Erling, Andrey Gubichev, Vlad Haprian, Moritz Kaufmann, Josep-Lluís Larriba-Pey, Norbert Martínez-Bazan, József Marton, Marcus Paradies, Minh-Duc Pham, Arnau Prat-Pérez, Mirko Spasic, Benjamin A. Steer, Gábor Szárnyas, Jack Waudby: "The LDBC Social Network Benchmark", CoRR, Vol. abs/2001.02299, 2020.
URL https://arxiv.org/abs/2001.02299

This talk presents the past, present, and ongoing activities of the Linked Data Benchmark Council (LDBC), and state-of-the-art research in specializing Worst-Case Optimal Joins for graph processing, on updatable graph storage using Packed Memory Arrays, and the future of graph processing using the Spark framework.

3.6 Graph Metadata Management

Angela Bonifati (University Claude Bernard – Lyon, FR)

Joint work of Angela Bonifati, Peter Furniss, Alastair Green, Russ Harmer, Eugenia Oshurko, Hannes Voigt
Main reference Angela Bonifati, Peter Furniss, Alastair Green, Russ Harmer, Eugenia Oshurko, Hannes Voigt: "Schema Validation and Evolution for Graph Databases", in Proc. of the Conceptual Modeling – 38th International Conference, ER 2019, Salvador, Brazil, November 4-7, 2019, Proceedings, Lecture Notes in Computer Science, Vol. 11788, pp. 448–456, Springer, 2019.
URL https://doi.org/10.1007/978-3-030-33223-5_37

Despite the maturity of commercial graph databases, little consensus has been reached so far on the standardization of data definition languages (DDLs) for property graphs (PG). Discussion on the characteristics of PG schemas is ongoing in many standardization and community groups. Although some basic aspects of a schema are already present in most commercial graph databases, full support is missing allowing to constraint property graphs with more or less flexibility.

In this work, we show how schema validation can be enforced through homomorphisms between PG schemas and PG instances by leveraging a concise schema DDL inspired by Cypher syntax. We also briefly discuss PG schema evolution that relies on graph rewriting operations allowing to consider both prescriptive and descriptive schemas.

3.7 Some Thoughts on Graph Processing Systems

Khuzaima Daudjee (University of Waterloo, CA)

This talk addresses various design choices in graph processing systems, such as: which system issue or bottleneck to address, making data layouts graph-aware, and Scale-up vs. Scale-out. This discussion about graph processing leads to a reconsideration of classic distributed systems ideas.

3.8 Stream Reasoning: graph stream processing + AI

Emanuele Della Valle (Polytechnic University of Milan, IT)

Joint work of Emanuele Della Valle, Stefano Ceri, Frank van Harmelen, Dieter Fense, Daniele Dell'Aglio, Abraham Bernstein
Main reference Daniele Dell'Aglio, Emanuele Della Valle, Frank van Harmelen, Abraham Bernstein: "Stream reasoning: A survey and outlook", Data Sci., Vol. 1(1-2), pp. 59–83, 2017.
URL https://doi.org/10.3233/DS-170006

Stream and complex event processing studies the abstractions, the systems, and the applications to model, process, and manage data characterized by being highly dynamic and unbounded. This type of data is commonly named Data Stream or Event. Graph processing does the same for data characterized by a high variety and complexity. When I conceived Stream Reasoning [1], I placed it in the intersection of those two fields, to study the abstractions, the systems and the applications of inference techniques to streaming RDF data.

In more than 10 years of research [2], researchers active in Stream Reasoning proposed the streaming extensions of the Semantic Web stack as well as of other abstractions studied in Knowledge Representation, Artificial Intelligence, and Robotics. They also developed dedicated systems that find application in several settings, from Smart Cities to Industry 4.0, from the Internet of Things to Social Media analytics. I came to this Dagstuhl seminar on Big Graph Processing Systems to discuss the possibility to further study Stream Reasoning in the context of property graphs and learn from the works done in dynamic and streaming graph processing.

References

1 Emanuele Della Valle, Stefano Ceri, Frank van Harmelen, Dieter Fensel: It's a Streaming World! Reasoning upon Rapidly Changing Information. IEEE Intelligent Systems 24(6): 83-89 (2009) https://doi.org/10.1109/MIS.2009.125

2 Daniele Dell'Aglio, Emanuele Della Valle, Frank van Harmelen, Abraham Bernstein: Stream reasoning: A survey and outlook. Data Sci. 1(1-2): 59-83 (2017) https://doi.org/10.3233/DS-170006

3.9 Graph Query Processing Techniques

Stefania Dumbrava (ENSIIE – Evry, FR)

Joint work of Angela Bonifati, Stefania Dumbrava, Emilio Jesús Gallego Arias
Main reference Angela Bonifati, Stefania Dumbrava, Emilio Jesús Gallego Arias: "Certified Graph View Maintenance with Regular Datalog", Theory Pract. Log. Program., Vol. 18(3-4), pp. 372–389, 2018.
URL https://doi.org/10.1017/S1471068418000224

We have investigated two emerging aspects of graph query processing: 1) approximation over property graphs [SUM19] and 2) formal verification using the Coq proof assistant [TPLP18].

On the first line of research, we use an edge-centric approach to build query-driven quotient property graph summaries. In this context, we focus on the approximate evaluation of counting queries involving recursive paths (already known to be difficult to evaluate even over pure RDF graphs). We address this challenge with an in-database approach, whereby we perform both summarization, based on label-constraint reachability heuristics, as well as computation of relevant statistical information, stored as properties. We then translate queries from the original graph onto the summary. We experimentally assess the compactness of the obtained summaries and the accuracy of answering counting recursive queries on them.

On the second line of research, we used the Coq proof assistant to develop a mechanically-certified engine for incrementally evaluating and maintaining graph views. The language we use is Regular Datalog (RD), a notable fragment of non-recursive Datalog that can express complex navigational queries, with transitive closure as a native operator. To this end, we mechanize an RD-specific evaluation algorithm capable of fine-grained, incremental graph view computation, which we prove sound with respect to the declarative RD semantics. Using the Coq program extraction mechanism, we test an OCaml version of the verified engine on preliminary benchmarks, confirming the feasibility of obtaining a unified, machine-verified, formal framework for dynamic graph query languages and their evaluation engines.

3.10 Interoperability and Integration of Graph-Data Systems

Olaf Hartig (Linköping University, SE)

Being able to store, process, and analyze data about connections and relationships between things is important for scientists and businesses alike. While there exist a plethora of database management systems that specialize in such types of graph-based data, each of these systems is designed for a specific class of query and analysis features. Hence, for an organization to leverage its graph data for a broad range of use cases, it becomes advantageous to employ multiple such systems that complement each other. Unfortunately, most of the systems are incompatible in terms of how exactly they represent the graph data. As a consequence, these systems cannot easily be combined with one another.

The goal of my research is to establish the foundations of solutions that can be used to integrate graph data across across systems employed for maintaining and using such data. My approach to this end is to introduce formal abstractions based on which we can define and analyze concepts that are concerned with heterogeneous forms of graph data. Based on these abstractions, I am working on the following contributions: a) formal mappings between different models commonly used to represent graph data, b) results about fundamental properties of these mappings, c) formal tools to compare potential options for bridging different forms of graph data via the notion of virtual views, d) results shown based on this tools, and e) algorithms and techniques to employ these concepts in practice.

3.11 Big Graph Processing Challenges in Cryptoasset Analytics

Bernhard Haslhofer (AIT – Austrian Institute of Technology – Wien, AT)

Cryptocurrencies such as Bitcoin or Ethereum Token systems are well-known examples of cryptoassets. They build on blockchain technology and form virtual ecosystems in which different types of actors build and share economic relationships expressed in terms of transactions. Cryptoasset analytics aims at developing quantitative methods and tools that contribute to a better understanding of cryptoasset ecosystem. Such techniques are becoming increasingly relevant for a wide spectrum of use cases, ranging from scientific investigations, over compliance and regulation, to law enforcement. This talk will explain why big graphs processing is relevant for cryptoasset analytics, quickly outline the status quo, and point out technical challenges.

3.12 The Graphalytics Ecosystem

Tim Hegeman (VU University Amsterdam, NL)

Joint work of Tim Hegeman, Alexandru Iosup, Alexandru Uta, Wing Lung Ngai, Stijn Heldens, Arnau Prat-Pérez, Thomas Manhardt, Hassan Chafi, Mihai Capota, Narayanan Sundaram, Michael J. Anderson, Ilie Gabriel Tanase, Yinglong Xia, Lifeng Nai, Peter A. Boncz
Main reference Alexandru Iosup, Tim Hegeman, Wing Lung Ngai, Stijn Heldens, Arnau Prat-Pérez, Thomas Manhardt, Hassan Chafi, Mihai Capota, Narayanan Sundaram, Michael J. Anderson, Ilie Gabriel Tanase, Yinglong Xia, Lifeng Nai, Peter A. Boncz: "LDBC Graphalytics: A Benchmark for Large-Scale Graph Analysis on Parallel and Distributed Platforms", PVLDB, Vol. 9(13), pp. 1317–1328, 2016.
URL https://doi.org/10.14778/3007263.3007270

We present here work on the Graphalytics Ecosystem: From Competitions to Performance Analysis. Understanding how well do graph processing systems perform is important given the popularity of graph datasets, of graph processing techniques, and of graph processing systems aiming to process such datasets by combining such techniques. At the core of the Graphalytics Ecosystem is the LDBC Graphalytics benchmark, which in a nutshell is an advanced benchmarking harness, supporting many classes of graph-processing algorithms, with diverse real and synthetic datasets, a diverse set of experiments, and a renewal process to keep the workload relevant. LDBC Graphalytics enables comparison of many platforms, community-driven and industrial, under a set of well-defined metrics and with tools for validating results.

However suitable LDBC Graphalytics is for benchmarking and comparing graph-processing systems, we argue there is more to fully understand their performance. We define a design space of performance analysis tools, where depth of analysis (e.g., type and number of metrics) and breadth of experimentation (e.g., many and diverse experiments across many graph-processing systems) form the main dimensions. In this design space, the Graphalytics Ecosystem provides a *set* of complementary approaches for understanding graph processing performance, with the LDBC Graphalytics benchmark providing low depth and moderate breadth, the performance analysis instruments Grade10 and Granula providing much better depth but only for a narrow set of graph-processing systems, and the Graphalytics Global Competition provides the lowest depth but the broadest span of graph-processing systems.

3.13 Typing Graph Queries

Jan Hidders (Birkbeck, University of London, GB)

We study the problem of determining a query is well-typed given a certain graph schema. The graph schema is based on schemas as defined the Property Graph Schema Working Group. Moreover, we discuss typing regimes that define how to derive the type of a query if it is well-typed.

In addition we formulate an additional notion of schema based dependencies of the form type t1 is a subtype of type t2. We try to derive given a query, what is the weakest schema that guarantees that the query is well-typed.

3.14 Querying Knowledge Graphs on the Web

Katja Hose (Aalborg University, DK)

Numerous approaches and techniques for querying knowledge graphs have been proposed. They differ in their architectures (centralized, federated, cloud, etc.) and the problems they aim to solve (indexing, completeness, join processing, query optimization, provenance, semantic data warehousing, etc.). In particular with respect to knowledge graphs on the Web, there are many challenges that go beyond efficient query processing [1], e.g., encoding and querying metadata, overcoming heterogeneity, and deciding on the underlying paradigms of data storage (triple store vs. graph store) or query language (SPARQL vs. Cipher vs. GraphQL, etc.). But even the most advanced approach cannot deliver any results if the necessary data is (temporarily) not available. Hence, to solve this problem, we have recently developed an approach that builds on P2P systems to keep the data available [2] and that uses prefix-partitioned indexes [3] to efficiently identify relevant data.

References

1 G. Montoya, Hala Skaf-Molli, Katja Hose. *The Odyssey Approach for Optimizing Federated SPARQL Queries.* In Proceedings of the 16th International Semantic Web Conference (ISWC), 2017.

2 Christian Aebeloe, Gabriela Montoya, Katja Hose. *A Decentralized Architecture for Sharing and Querying Semantic Data.* In The Semantic Web – 16th International Conference (ESWC), 2019.

3 Christian Aebeloe, Gabriela Montoya, Katja Hose *Decentralized Indexing over a Network of RDF Peers.* In Proceedings of the 18th International Semantic Web Conference (ISWC), 2019.

3.15 Forecasting Information Diffusion in Online Environments: Lessons from DARPA's SocialSim Project

Adriana Iamnitchi (University of South Florida – Tampa, US)

This talk describes our graph processing experience while designing solutions for forecasting social media activities in various platforms, from GitHub software collaborations to cross-platform disinformation campaigns on Twitter and YouTube. Our experience is that while our raw datasets were huge, the graphs extracted in the end were relatively small, due to objective-specific data filtering (such as time windows of interests) and particularities of information diffusion phenomena (such as cascades of limited size). This observation highlights the need of connecting graph-processing components with (possibly existing, well established) general data processing components for facilitating graph extractions, data curation, etc.

3.16 The Distributed Systems Memex: Graph Processing Elements

Alexandru Iosup (VU University Amsterdam, NL)

Main reference Alexandru Iosup, Alexandru Uta, Laurens Versluis, Georgios Andreadis, Erwin Van Eyk, Tim Hegeman, Sacheendra Talluri, Vincent van Beek, Lucian Toader: "Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems", in Proc. of the 38th IEEE International Conference on Distributed Computing Systems, ICDCS 2018, Vienna, Austria, July 2-6, 2018, pp. 1224–1237, IEEE Computer Society, 2018.
URL https://doi.org/10.1109/ICDCS.2018.00122

In the 1940s, Vannevar Bush defined the concept of the personal *memex* as a person's device for storing and accessing all information and communication involving that person [1]. Bush identified many benefits for archiving large amounts of personal data into the memex, including learning about and eradicating diseases, enabling more creative and thought-related time by eliminating tasks that can be automated, etc. (Bush did not spend much time on the drawbacks, some of which are related to the appearance in the late-2010s of privacy-related regulations, e.g., GDPR.)

Similarly, we have posited in the early 2010s [2] that archiving large amounts of operational traces collected from the distributed systems that currently underpin our society into a Distributed Systems Memex can be beneficial for learning about and eradicating performance and related issues, for enabling more creative designs and extending automation, etc. We have added more recent concerns [3]: the Distributed Systems Memex could also serve as data-source to facilitate reproducible experiments, could preserve quantitative evidence for the future debates related to this class of systems, and could enable retrospective studies about the ethical operation of such systems. We also posit that a Distributed Systems Memex could help with the preservation of original designs and of their use, a valuable heritage.

Adding graph processing to the Distributed Systems Memex is non-trivial: What should the Distributed Systems Memex include? What data? Which types of distributed systems and ecosystems? How can such a Memex be designed, to support the concerns mentioned earlier? What instruments could it use and how could it be implemented overall?

The Distributed Systems Memex can only succeed as a community-driven effort.

References

1 Bush (1945) As we may think. The Atlantic, Jul 1945. [Online] Available: https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/
2 Iosup (2012) Towards logging and preserving the entire history of distributed systems. Is the Future of Preservation Cloudy? (Dagstuhl Seminar 12472), pp. 126–127. [Online] Available: https://doi.org/10.4230/DagRep.2.11.102
3 Iosup et al. (2019) The AtLarge Vision on the Design of Distributed Systems and Ecosystems. ICDCS 2019: 1765-1776. [Online] Available: https://arxiv.org/pdf/1902.05416.pdf

3.17 Representation learning of dynamic graphs

Vasiliki Kalavri (Boston University, US)

This talk introduces the problem of representation learning of dynamic graphs and discuss ideas towards avoiding retraining from scratch when changes occur. While inductive representation learning can generate predictions for unseen vertices, it cannot learn new knowledge from dynamic graphs. We describe a priority-based method for selecting rehearsed samples and show preliminary results on a classification problem.

3.18 Compiling graph algorithms to SQL

Hugo Kapp (Oracle Labs Switzerland – Zürich, CH)

In the past decades, relational databases have been filled up with huge amounts of data. Until now, SQL was seen as expressive enough to perform analytics on this data and answer specific questions. Nowadays, graphs are seen as a simple and intuitive way to express more complex analytics. To bridge the gap between these solutions, users have to export their data from their relational database(s), then import it into an external specialized graph processing engine.

To avoid moving data around, as well as to reduce the effort induced to reflect updates in the original relational data, we propose to perform some of these graph analytics directly on the relational data, inside the RDBMS. This requires the graph analytics workloads to be expressed in what RDBMSs can understand: SQL. The work presented here is focused on translating a procedural DSL for graph algorithms into SQL.

3.19 Analysis of Query Logs

Wim Martens (Universität Bayreuth, DE)

We present some of the results of a deep analysis of SPARQL query logs in collaboration with Angela Bonifati and Thomas Timm. We investigated about 500 million queries coming from Wikidata, DBpedia, geographic data, biological data, museum data, and analyzed a large range of their properties. In a first, shallow study we analyzed the queries size, keyword use, use of projection, etc. In a second, deeper study we analyzed the structure of queries, their (hyper)treewidth, and the structure of property paths (regular expressions) in the queries.

3.20 Towards Graph Processing Standards: The GQL Graph Query Language

Stefan Plantikow (Neo4j – Berlin, DE)

In this talk, we explore the language as the medium for graphs as the universal data model. We focus in particular on the "ISO/IEC WD 39075 – Database Language – GQL" standard, which aims to provide for property-graph databases a language specification that is next-generation, declarative, composable, compatible, modern, and intuitive.

3.21 Querying and Processing Big Property Graphs

Mohamed Ragab (University of Tartu, EE)

This talk introduces new ideas on querying and processing big property-graphs. We focus on the Morpheus framework as "Cypher on Top of Spark", which provides distributed Cypher queries and data integration for a variety of applications in data science. To this framework, we propose to add various graph-aware query optimizations techniques, and experiments leading to in-depth performance analysis.

3.22 BigGraph Projects at UBC

Matei R. Ripeanu (University of British Columbia – Vancouver, CA)

We present here four Big Graph projects developed at UBC, in collaboration with many international partners. The projects are: Infrastructure for Graph Processing on Heterogeneous (CPU+GPUs) Platforms, Exact/ Approximate Pattern Matching for Massive Labelled Graphs, Infrastructure and Algorithms to Support Bi-Temporal Graph Exploration, and Infrastructure and Algorithms to Support On-Line Graph Analytics.

3.23 Graph Processing Systems Research Overview

Semih Salihoglu (University of Waterloo, CA)

We give a broad overview of the systems research graph processing that I work on. GraphflowDB is a graph database we are developing from scratch that rethinks many core components of a native graph database, including query processor, optimizer, physical design, indexes, and triggers. GraphSurge is a graph analytics system designed for applications that need to define multiple views of a base graph and run the same computation on each one. GraphWrangler is a software that is designed to extract graphs out of relational tables with two specific goals: (1) to obtain a graph view over relational tables with a few visual interactions, such as a drag and drops of rows and columns, and (2) to obtain a script that contains a mapping from relational tables to a property graph model, which can be used to perform ETL to move data from a relational system to a graph database.

3.24 Scalable Graph Algorithms and Partitioning

Christian Schulz (Universität Wien, AT)

Our research is mostly concerned with engineering scalable graph algorithms. This talk gives a broad overview and then focuses on one specific area: High Quality Graph Partitioning.

3.25 Querying Property Graphs

Petra Selmer (Neo4j – London, GB)

Property graphs have come of age in recent years. The querying of such graphs has naturally attracted a lot of interest and progressed considerably as a result.

One major development is the inception of GQL (Graph Query Language) as a new ISO/IEC project, with the aim of defining a comprehensive, expressive standardized declarative query language, which will, in the first instance, codify features provided by existing industrial property graph implementations. GQL will be substantially based on Cypher, which is implemented by Neo4j, RedisGraph, AnzoGraph, Memgraph, and SAP HANA Graph, as well as the Cypher for Apache Spark and Cypher for Gremlin projects. Although Cypher will form the substrate of GQL, there are other languages that will influence GQL, such as PGQL, G-CORE and the Property Graph Query Extensions currently under development in the latest version of SQL.

Collaborating with the University of Edinburgh, the formal denotational semantics of Cypher (both reading and updating semantics) have been defined. It is our aim to have in place a similar process for the specification of GQL, whereby formal semantics will inform and provide rigour in tandem with the production of the natural language specification that will be become the GQL Standard.

3.26 Designing and Building Enterprise Knowledge Graphs in Practice and My Research Interest

Juan F. Sequeda (data.world – Austin, US)

Joint work of Juan F. Sequeda, Willard J. Briggs, Daniel P. Miranker, Wayne P. Heideman
Main reference Juan F. Sequeda, Willard J. Briggs, Daniel P. Miranker, Wayne P. Heideman: "A Pay-as-you-go Methodology to Design and Build Enterprise Knowledge Graphs from Relational Databases", in Proc. of the The Semantic Web – ISWC 2019 – 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II, Lecture Notes in Computer Science, Vol. 11779, pp. 526–545, Springer, 2019.
URL https://doi.org/10.1007/978-3-030-30796-7_32

Data Integration has been an active area of computer science research for over two decades. A modern manifestations is as Knowledge Graphs which integrates not just data but also knowledge at scale. Tasks such as Domain modeling and Schema/Ontology Matching are fundamental in the data integration process. Research focus has been on studying the data integration phenomena from a technical point of view (algorithms and systems) with the ultimate goal of automating this task.

In the process of applying scientific results to real world enterprise data integration scenarios to design and build Knowledge Graphs in practice, we have experienced numerous obstacles because the human and social factor is not taken into account. We argue that we need to think outside of a technical box and further study the phenomena of data integration with a human-centric lens: from a socio-technical point of view.

3.27 Building the Uber Graph

Joshua Shinavier (Uber Engineering – Palo Alto, US)

This short talk introduces Uber's knowledge graph and data standardization efforts, as well as the algebraic property graphs (APG) data model. APG is used for Uber's standardized schemas, as an intermediate language for transformations between schemas written in various other languages, and as a means of extending the graph processing idiom to diverse non-graph data sources. The data model has been formalized using category theory and with reference to the concept of algebraic databases, which is mentioned as a topic for additional future work.

3.28 Towards standardized benchmarks for graph processing

Gábor Szárnyas (Budapest Univ. of Technology & Economics, HU)

Defining benchmarks that capture certain types of workloads has been standard practice in database engineering. Such benchmarks offer a number of advantages: (1) they serve as case studies with openly available data sets and queries, (2) they capture a common understanding of what the systems-under-benchmark should be capable of, (3) they serve as a yardstick to compare the performance of industry tools, and (4) they free researchers from the burden of coming up with their own experiments, making their performance evaluations comparable to previous work.

Designing meaningful benchmarks is a major undertaking and necessitates a lengthy discussion to determine points of interest. It also requires creating a sizable software stack, consisting of a data generator, a driver that orchestrates benchmark executions, reference implementations, a reporting framework, etc. In the graph processing space, the Linked Data Benchmark Council (LDBC), established in 2012, has the goal of providing a standard benchmark suite that captures our current understanding of graph processing challenges. The current set of benchmarks consists of Graphalytics (focusing on graph analytics), the Semantic Publishing Benchmark (targeting RDF-based systems) and the Social Network Benchmark that focuses on graph queries (covering OLTP queries in the Interactive workload and OLAP queries in the Business Intelligence workload).

Our recent work was leading the Social Network Benchmark task force with the aim of extending the Business Intelligence workload by introducing delete operations and adding more queries that capture graph queries that are relevant for users but pose challenges for graph database systems.

3.29 Velocity on the Web

Riccardo Tommasini (Polytechnic University of Milan, IT)

Main reference Riccardo Tommasini: "Velocity on the Web – a PhD Symposium", in Proc. of the Companion of The 2019 World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019, pp. 56–60, ACM, 2019.
URL https://doi.org/10.1145/3308560.3314192

The Web is a distributed environment populated by resources and agents that identify, represent, and interact with them. The decentralised nature of Web applications is one of the reasons for the popularity of the Web. Nevertheless, the Web results in an unbounded and noisy environment populated by heterogeneous resources. As part of the Web environment, applications must take resource heterogeneity into account. The Web of Data is the Web extension that addresses this challenge, known as Data Variety, using a stack of semantic technologies that include RDF, SPARQL, and OWL.

Recently, a new generation of Web applications is showing the need for taming Data Velocity, i.e., processing data as soon as they arrive and before it is too late. New protocols are emerging to improve the Web's data infrastructure. Web Sockets and Server-Sent Events respectively enable continuous and reactive data access.

Data velocity is related to the whole data infrastructure, and new abstractions are required, i.e., streams and events that are the fundamental entities of the stream processing. Although seminal work on Stream Reasoning and RDF Stream Processing paved the road for addressing Velocity on the Web, the following research question remains unanswered: Can we identify, represent, and interact with streams and events on the Web?

3.30 HAGGLE and SO(DA)2: Two PNNL Graph System Projects

Antonino Tumeo (Pacific Northwest National Lab. – Richland, US)

This presentation briefly overviews HAGGLE: the Hybrid Attributed Generic Graph Library Environment, PNNL's project for the DARPA HIVE (Hierarchical Identify, Verify and Exploit) program, and SO(DA)^2: Software Defined Architectures for Data Analytics, one of the flagship project in PNNL's Data and Model Convergence Initiative. HAGGLE is designing stack to perform graph analytics on novel graph processors and exascale systems, implementing a Hybrid Data Model (graphs and tables). SO(DA)^2 is implementing a stack for Data Analytics targeting novel runtime reconfigurable architectures.

3.31 Serverless Graph Processing

Alexandru Uta (VU University Amsterdam, NL)

Joint work of Lucian Toader, Alexandru Uta, Ahmed Musaafir, Alexandru Iosup
Main reference Lucian Toader, Alexandru Uta, Ahmed Musaafir, Alexandru Iosup: "Graphless: Toward Serverless Graph Processing", in Proc. of the 18th International Symposium on Parallel and Distributed Computing, ISPDC 2019, Amsterdam, The Netherlands, June 3-7, 2019, pp. 66–73, IEEE, 2019.
URL https://doi.org/10.1109/ISPDC.2019.00012

Serverless computing promises ease-of-use, fine-grained scalability, elasticity and billing, and improved resource utilization via improved workload consolidation of many small functions instead of large monoliths. We investigate the benefits of serverless computing for graph processing workloads and implement the first serverless graph analytics prototype: Graphless. We show that Graphless is able to leverage the fine-grained elasticity that several analytics algorithms exhibit. However, to achieve high performance in a serverless environment, improved communication schemes and low-latency serverless storage.

3.32 Correlating graph properties with graph processing performance

Ana Lucia Varbanescu (University of Amsterdam, NL)

Joint work of Ahmed Musaafir, Alexandru Uta, Henk Dreuning, Ana-Lucia Varbanescu
Main reference Ahmed Musaafir, Alexandru Uta, Henk Dreuning, Ana-Lucia Varbanescu: "A Sampling-Based Tool for Scaling Graph Datasets". ACM/SPEC ICPE 2020 (accepted, in print)

It is well-known that, for many graph processing algorithms, performance is correlated with graph properties. However, there's little progress in analytically defining this correlations, even for well-studied algorithms like BFS or PageRank. In our work, we have attempted three different ways to better understand such correlation in the specific case of parallel graph processing algorithms. In this talk, we summarize our results in these three directions: (1) performance models based on hardware counters, (2) machine learning solutions for optimal algorithm selection, and (3) graph scaling to facilitate in-depth performance analysis. We illustrate the results we achieved in each direction with one relevant example, and list open questions that lie ahead.

3.33 High-Level Graph Processing Abstractions

Hannes Voigt (Neo4j – Leipzig, DE)

In this talk, we analyze the state-of-art in graph processing, and find that it is mainly driven bottom-up from hardware. We ask about better solutions: Are there even better abstractions? Where is the balance between expressivity and convenience (imperative vs. declarative)? How about the integration with query language: how big is the need, the desired degree, etc.? How could we systematically analyze commonality in graph programs? and, last but not least, What are optimization opportunities below an abstraction?

3.34 Graph! A Fundamental Structure for Maps

Hsiang-Yun Wu (TU Wien, AT)

Visualization has been defined as a scientific field to take advantage of human vision and perception in order to amplify human cognition and gain insight in the complex data. Network visualization is one of the essential topics in the field to visually support the comprehensive understanding of the relationship in the underlying Big Data. More specifically, network layout, or namely graph drawing, has been considered as a key factor to understandability and memorability. In this talk, several network visualization techniques have been introduced to untangle nested and complex relationships using graphs, especially focusing on geospatial maps and biological pathway diagrams. Those maps are automatically computed through mimicking the design process of human illustrators. We extract meaningful aesthetic criteria and formulate them into machine computable mathematical equations. Simple examples, such as removing crossings through aligning two edges with minimum pairwise distances and uniformly distributing nodes over then entire map domain through centroid forces, are shown. Finally, the evaluation of usability is presented to qualitatively demonstrate the effectiveness of the proposed approaches.

3.35 AvantGraph: an open-source recursive analytics engine

Nikolay Yakovets (TU Eindhoven, NL)

We present AvantGraph, an upcoming, open-source graph processing system featuring main memory optimizations, recursive analytics, various vectorization and compilation advancements, and guarantees of worst-case optimality.

3.36 A Task-Centric Approach to Revolutionize Big Graph Systems Research

Da Yan (The University of Alabama – Birmingham, US)

The short talk reviews existing works on Big Graph computational systems and Big Data computational systems in general, esp. those of the Hadoop Ecosystem, it indicates the weaknesses in existing research and envisions a new trend of compute-intensive system design that is able to fully utilize CPU cores in a computer cluster. A few Youtube videos on preliminary works are put online for those interested to check out at https://www.youtube.com/channel/UCQyivHtbwTIvwfO7ZWQVEow

3.37 Computer Systems Optimisation in Complex and Large Parameter Space

Eiko Yoneki (University of Cambridge, GB)

Managing efficient configurations is a central challenge for computer systems. I will introduce two recent projects: 1) Structured Bayesian Optimisation (SBO) to optimise systems in complex and high-dimensional parameter space, and 2) RLgraph, our framework for Reinforcement Learning (RL) to bring performance improvements to dynamically evolving tasks such as scheduling or resource management. These frameworks can be used tuning complex, high dimensional, and/or dynamic/combinatorial parameter space in computer systems. The applications that can take advantage of these optimisation frameworks are diverse ranging the hyper parameter tuning of neural networks, device placement of distributed execution engines (e.g. TensorFlow), transformation of computation graphs of Deep Neural Network, compiler optimisation, ASIC design, to database query optimisation.

4 Working groups

4.1 Working Group on Abstractions for Big Graph Processing

Renzo Angles (University of Talca – Chile, CL), Walid Aref (Purdue University – West Lafayette, US), Marcelo Arenas (PUC – Santiago de Chile, CL), Angela Bonifati (University Claude Bernard – Lyon, FR), Emanuele Della Valle (Polytechnic University of Milan, IT), Stefania Dumbrava (ENSIIE – Evry, FR), Olaf Hartig (Linköping University, SE), Jan Hidders (Birkbeck, University of London, GB), Hugo Kapp (Oracle Labs Switzerland – Zürich, CH), Wim Martens (Universität Bayreuth, DE), M. Tamer Özsu (University of Waterloo, CA), Stefan Plantikow (Neo4j – Berlin, DE), Sherif Sakr (University of Tartu, EE), Petra Selmer (Neo4j – London, GB), Juan F. Sequeda (data.world – Austin, US), Joshua Shinavier (Uber Engineering – Palo Alto, US), Hannes Voigt (Neo4j – Leipzig, DE), and Hsiang-Yun Wu (TU Wien, AT)

The purpose of this working group was to identify core requirements for graph processing abstractions. To this end, one has to consider the desiderata of practitioners, developers, and researchers alike and to balance implementational tractability with specification expressiveness. The main debate topics concerned defining a unifying foundation for graph models (the so-called "Dragon" model) and using this as the basis for achieving interoperability. First, we have singled out primitive graph operations that can serve as a comparison baseline and built a lattice of existing graph abstractions. Next, we have discussed constructive definitions of the top ("Dragon") lattice element, based on logics, algebra, type theory, and category theory. Finally, we have looked into centralized (star-shaped) and decentralized (peer-to-peer) approaches to navigating the lattice of abstractions and providing abstract mappings between this overarching specification and particular instances. We have also analysed the interoperability between the relational and graph paradigms, by looking at tipping point use cases.

Key elements of the discussion:

1. Human Data(base) Interaction,
2. Lattice of Abstractions,
3. Type algebras and Category Theory as a basis for a unifying foundation,
4. Logics as basis for a unifying foundation,
5. A graph/operational algebra,
6. Relational Meet Graphs.

4.2 Working Group on Performance of Big Graph Processing Systems

Alexandru Iosup (VU University Amsterdam, NL), Maciej Besta (ETH Zürich, CH), Peter A. Boncz (CWI – Amsterdam, NL), Khuzaima Daudjee (University of Waterloo, CA), Tim Hegeman (VU University Amsterdam, NL), Mohamed Ragab (University of Tartu, EE), Sherif Sakr (University of Tartu, EE), Semih Salihoglu (University of Waterloo, CA), Christian Schulz (Universität Wien, AT), Gábor Szárnyas (Budapest Univ. of Technology &

Economics, HU), Antonino Tumeo (Pacific Northwest National Lab. – Richland, US), Ana Lucia Varbanescu (University of Amsterdam, NL), Nikolay Yakovets (TU Eindhoven, NL), and Eiko Yoneki (University of Cambridge, GB)

The working group focused on the problem of assessing the performance of graph processing systems, i.e. how to design meaningful experiments and avoid reporting misleading results. *First*, we identified some key design decisions that have a significant impact on performance: Models (abstractions): data model, programming model, computational model, Architecture of systems, and Scalability considerations: load balancing, optimization, auto-tuning. These decisions are often intertwined, e.g. one can think of Pregel as computational model, as a system and as a reference architecture. *Second*, we identified some challenges for the systems and database communities w.r.t. the state-of-practice in assessing the performance of graph processing tools. The rest of the summary collects these.

Key elements of the discussion:

1. Methodological aspects of performance experiments,
2. Workloads,
3. Benchmarks, metrics and fine-grained instrumentation,
4. Reference architecture,
5. The Distributed Systems Memex,
6. Specialization vs. portability and interoperability,
7. Interplay between phenomena and complexity,
8. Design and performance space exploration,
9. Techniques to improve performance,
10. The gap between industry and academia/research,
11. Ambitious experiments on big graphs,
12. Vision for the next 5-10 years (+analysis of last 5-10 years).

4.3 Working Group on Ecosystems of Big Graph Processing Systems

Sherif Sakr (University of Tartu, EE), Khaled Ammar (Thomson Reuters Labs, CA), Angela Bonifati (University Claude Bernard – Lyon, FR), Khuzaima Daudjee (University of Waterloo, CA), Bernhard Haslhofer (AIT – Austrian Institute of Technology – Wien, AT), Katja Hose (Aalborg University, DK), Adriana Iamnitchi (University of South Florida – Tampa, US), Vasiliki Kalavri (Boston University, US), M. Tamer Özsu (University of Waterloo, CA), Eric Peukert (Universität Leipzig, DE), Matei R. Ripeanu (University of British Columbia – Vancouver, CA), Riccardo Tommasini (Polytechnic University of Milan, IT), Alexandru Uta (VU University Amsterdam, NL), Da Yan (The University of Alabama – Birmingham, US), and Eiko Yoneki (University of Cambridge, GB)

In modern setups, graph-processing is not a self-sustained, independent activity, but rather part of a larger big-data processing ecosystem with many system alternatives and possible design decisions. We need a clear understanding of the impact and the trade-offs of the various decisions in order to effectively guide the developers of big graph processing applications.

Key elements of the discussion:

1. Workloads of Graph Processing Ecosystems,
2. Overview/Vision/Standards/GQL,
3. Design Dimensions of Ecosystems,
4. (Populated) Reference Architecture,
5. Use Cases for Graph Processing Ecosystems,
6. System Requirements (Scale-up vs. scale-out),
7. Dynamic and Streaming Aspects,
8. Visualization Aspects (gaps between graph processing system and visualization, future technical challenge)

5 Panel discussions

5.1 Big Graph Processing: The System Perspective

Antonino Tumeo (Pacific Northwest National Lab. – Richland, US)

This tutorial introduces the terminology, core concepts, key solved and open challenges, and a roadmap for the future, taking a perspective from the (large-scale) computer systems community. We start by discussing how a big graph processing system is currently layered. We then go through abstractions, programming models, runtime, and custom architectures designs currently employed for Graph Processing. Finally, we discuss some of the ongoing research on Big Graph Processing System at PNNL, highlighting the research challenges, issues, and opportunities.

5.2 Graph Processing: A Panoramic View and Some Open Problems

M. Tamer Özsu (University of Waterloo, CA)

This tutorial introduces the terminology, core concepts, key solved and open challenges, and a roadmap for the future, taking a perspective from the data management community.

Participants

- Khaled Ammar
Thomson Reuters Labs, CA
- Renzo Angles
University of Talca – Chile, CL
- Walid Aref
Purdue University –
West Lafayette, US
- Marcelo Arenas
PUC – Santiago de Chile, CL
- Maciej Besta
ETH Zürich, CH
- Peter A. Boncz
CWI – Amsterdam, NL
- Angela Bonifati
University Claude Bernard –
Lyon, FR
- Khuzaima Daudjee
University of Waterloo, CA
- Emanuele Della Valle
Polytechnic University of
Milan, IT
- Stefania Dumbrava
ENSIIE – Evry, FR
- Olaf Hartig
Linköping University, SE
- Bernhard Haslhofer
AIT – Austrian Institute of
Technology – Wien, AT
- Tim Hegeman
VU University Amsterdam, NL
- Jan Hidders
Birkbeck, University of
London, GB
- Katja Hose
Aalborg University, DK
- Adriana Iamnitchi
University of South Florida –
Tampa, US
- Alexandru Iosup
VU University Amsterdam, NL
- Vasiliki Kalavri
Boston University, US
- Hugo Kapp
Oracle Labs Switzerland –
Zürich, CH
- Wim Martens
Universität Bayreuth, DE
- M. Tamer Özsu
University of Waterloo, CA
- Eric Peukert
Universität Leipzig, DE
- Stefan Plantikow
Neo4j – Berlin, DE
- Mohamed Ragab
University of Tartu, EE
- Matei R. Ripeanu
University of British Columbia –
Vancouver, CA
- Sherif Sakr
University of Tartu, EE
- Semih Salihoglu
University of Waterloo, CA
- Christian Schulz
Universität Wien, AT
- Petra Selmer
Neo4j – London, GB
- Juan F. Sequeda
data.world – Austin, US
- Joshua Shinavier
Uber Engineering –
Palo Alto, US
- Gábor Szárnyas
Budapest Univ. of Technology &
Economics, HU
- Riccardo Tommasini
Polytechnic University of
Milan, IT
- Antonino Tumeo
Pacific Northwest National Lab. –
Richland, US
- Alexandru Uta
VU University Amsterdam, NL
- Ana Lucia Varbanescu
University of Amsterdam, NL
- Hannes Voigt
Neo4j – Leipzig, DE
- Hsiang-Yun Wu
TU Wien, AT
- Nikolay Yakovets
TU Eindhoven, NL
- Da Yan
The University of Alabama –
Birmingham, US
- Eiko Yoneki
University of Cambridge, GB