Breaking the Chains: On Declarative Data Analysis and Data Independence in the Big Data Era

Volker Markl
Technische Universität Berlin
Sekr. E-N 7, Einsteinufer 17
10587 Berlin, Germany
+49 30 314 23555
volker.markl@tu-berlin.de

ABSTRACT

Data management research, systems, and technologies have drastically improved the availability of data analysis capabilities, particularly for non-experts, due in part to *low-entry barriers* and *reduced ownership costs* (e.g., for data management infrastructures and applications). Major reasons for the widespread success of database systems and today's multi-billion dollar data management market include *data independence*, separating physical representation and storage from the actual information, and *declarative languages*, separating the program specification from its intended execution environment. In contrast, today's big data solutions do not offer data independence and declarative specification. As a result, big data technologies are mostly employed in newly-established companies with IT-savvy employees or in large well-established companies with big IT departments. We argue that current big data solutions will continue to fall short of widespread adoption, due to usability problems, despite the fact that in-situ data analytics technologies achieve a good degree of schema independence. In particular, we consider the lack of a declarative specification to be a major roadblock, contributing to the scarcity in available data scientists available and limiting the application of big data to the IT-savvy industries. In particular, data scientists currently have to spend a lot of time on tuning their data analysis programs for specific data characteristics and a specific execution environment. We believe that the research community needs to bring the powerful concepts of declarative specification to current data analysis systems, in order to achieve the broad big data technology adoption and effectively deliver the promise that novel big data technologies offer.

Proceedings of the VLDB Endowment, Vol. 7, No. 13

1. INTRODUCTION

The last decade was marked by the digitalization of virtually all aspects of our daily lives. Today, businesses, government institutions, and science and engineering organizations face an avalanche of digital data on a daily basis. All due in part to the decline in disk storage costs, the ever-increasing popularity of cloud storage services, and the ubiquitous availability of networked devices. At first glance this appears to be favorable for our increasingly networked society. However, in many ways it is a burden. Data is neither information, nor knowledge. Instead, data is of great value once it has been refined and analyzed, in order to address well-formulated questions, concerning problems of interest. It is only then that economic and social benefits can be fully realized.

Modern big data analytics questions are often solved using techniques drawn from varying fields, including graph and network analysis, machine learning, mathematics, statistics, signal processing, and text processing, among others. Currently, data scientists, well versed in scalable data analysis methods, scalable systems programming, and knowledge in an application domain are needed to derive insight from big data. Unfortunately, data scientists with skills in both scalable systems and (potentially domain specific) data analysis methods are few in number. They are expensive and in high-demand. Consequently, this limits the amount of value that can currently be generated from big data for society as a whole. Moreover, despite the ever-increasing number of data science programs (at universities worldwide) and student enrollments, it will still be impossible to educate these so-called "*jack-of-all-trades*," as the required skills are both complex and diverse (as depicted in **Figure 1** and **Figure 2** below).

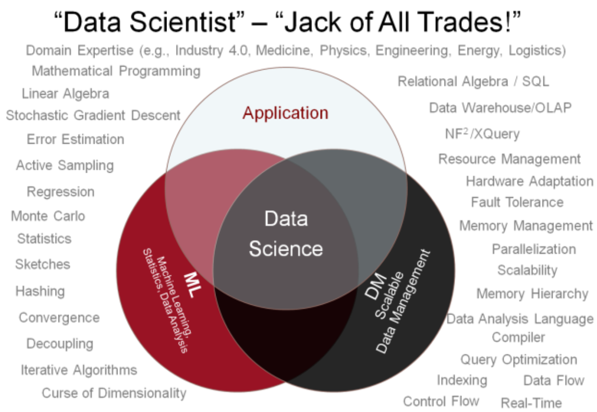

Figure 1. A Data Scientist is a "Jack-of-All-Trades."

Before "big data," the few programmers with MPI expertise, predominantly located in supercomputing centers were sufficient in number. For many decades, software engineers and general users in varying domains did not have to worry about scalability issues in their computing systems, thanks in part to higher-level programming languages, compilers, and database systems. In contrast, today's existing technologies have reached their limits due to big data requirements, which involve data volume, data rate and heterogeneity, and the complexity of the analysis algorithms, which go beyond relational algebra, employing complex user-defined functions, iterations, and distributed state.

In the era of many-core processors, cloud computing, and NoSQL, we must ensure that well-established declarative language concepts (inherent in relational database systems) make their way into big data systems. In order to make this a reality, the research community will need to address the related challenges. For example, (i) designing a programming language specification that does not require systems programming skills, (ii) mapping programs expressed in this programming language to a computing platform of their own choosing, and (iii) executing these in a scalable manner. In particular, this means devising execution strategies that are distributed, parallelized, and support both in-memory technologies and out-of-core execution for data-intensive algorithms.

In order to meet this challenge the *compiler*, *data analysis*, *database systems*, *distributed systems, and machine learning* communities, among others, will have to come together. We will have to develop novel scalable algorithms and systems that are able to organize the data deluge and intelligently distill information to create value. To achieve this, declarative query languages must now be extended to support the declarative specification of varying analysis methods (e.g., anomaly detection, classification, and clustering). This will particularly require making iterations and limited forms of (distributed) state first class citizens of an extended relational algebra.

Figure 2: Deep Analysis is the Name of the Game.

Furthermore, the power of declarative languages, namely, automatic optimization, parallelization, and adaptation of the same program to varying distributed systems and novel hardware architectures (depending on data distribution, data size, data rate, and system load) must be preserved. In this way, we will be able to overcome the current "stone age" in big data analytics. That is, algorithm specifications in systems that do not automatically optimize (e.g., MPI, MapReduce, and Hadoop), imperative languages (e.g., C), object-oriented languages (e.g., Java), and relational-oriented languages (e.g., SQL, XQuery, Pig, Hive, and JAQL) with non-tunable external driver programs, and technical computing systems (e.g., R and MATLAB) that do not scale.

2. A FIRST STEP: STRATOSPHERE AND APACHE FLINK

With grants from the EIT[1] ICT Labs, the DFG[2], IBM, Oracle, Hewlett-Packard, Deutsche Telekom, BMBF[3],

[1] The European Institute of Innovation and Technology

[2] The German National Science Foundation sponsored Research Unit Stratosphere involving TU Berlin, HU Berlin, and the Hasso-Plattner Institute in Potsdam.

[3] The German Ministry of Education and Research

BMWi[4], and the European Commission (under the FP7 funding programme), a large team of researchers has taken its first steps towards building a next generation big data analytics infrastructure called *Stratosphere* [1]. Its primary goal is "[…] to empower data scientists to conduct deep data analysis without the need to concern themselves with parallelization and query optimization issues."

Stratosphere enables massively parallel in-situ data analytics using a programming model based on second order functions [5]. It offers *Java* and *Scala* as programming language frontends, a scripting language called *Meteor*, and a graph processing frontend called *Spargel* with a Pregel-like interface, with other interfaces in development. Key design points include automatic optimization, parallelization, and hardware adaptation of complex data analysis pipelines from laptops to compute clusters. Additionally, native support for iterative data analysis programs, relational operators, and user-defined functions. Through the concepts of bulk and workset iterations, Stratosphere can process information extraction and integration operations together with deep analytics in a single system, subsuming many specialized systems for graph processing or machine learning in a single environment [3]. Released under an Apache 2.0 software license, Stratosphere is an open source-system that runs standalone, natively in compute clusters, or without special installation in Hadoop clusters via YARN (see **Figure 3**).

Stratosphere [9] is a hybrid that combines database technology with map/reduce technology. In this way, Stratosphere retains the processing of complex UDFs, schema on read, complex data types, and the scalability of map/reduce systems while offering the declarativity, independence, and automatic optimization of database technology. Its optimizer can handle advanced data analysis programs that go beyond the relational algebra, by optimizing relational operations jointly with user-defined functions and iterative programs, utilizing varying compiler and database technologies. In this manner, Stratosphere enables data scientists to focus on their respective problem and relieves them from scalable systems programming details.

Most recently, the Apache Software Foundation accepted Stratosphere as one of its newest incubator projects, under the name "*Apache Flink*." [10] Hence, future Stratosphere versions will be released under this new name.

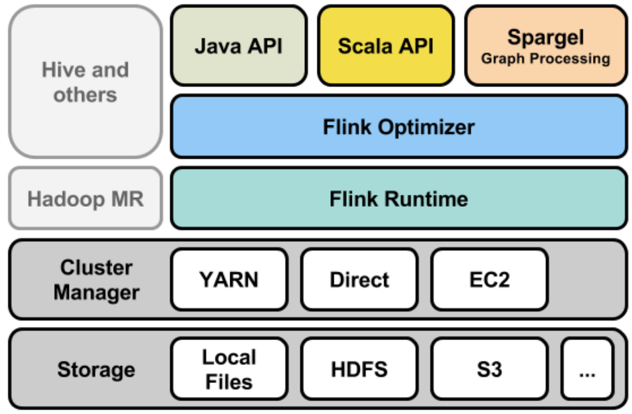

Figure 3. The Stratosphere/Flink Stack

3. THE ROAD AHEAD

Processing iterative, stateful data analysis programs on vast amounts of "data in motion" under low-latency while leveraging a declarative specification and ensuring data independence requires novel methods and techniques both from a systems and an algorithmic point-of-view. As research community, we will have to build on our existing results to considerably advance the state-of-the-art in designing and building systems for optimizing and executing complex data analysis programs on potentially evolving datasets under the constraint of significantly reducing latency (i.e., the time until first analysis results are available). In particular, we see the following major research challenges:

- **declarative specification and automatic deployment of complex data analysis programs:** we need to extend the declarative approach of relational databases to describe, plan, optimize, and execute iterative data analysis programs (DAPs) with complex user defined functions and mutable state;
- **declarative scalable data analysis libraries:** we need to cooperate with the signal processing, machine learning, and the general data analysis community on declarative, scalable algorithms in order to enable deep analysis of big data;
- **continuous, workload-aware optimization and execution of data analysis programs over evolving data:** data analysis usually is a multi-

[4] The German Ministry of Economics and Energy

user scenario, where multiple DAPs run concurrently on the same system, each with the requirement for low-latency answers, and each competing for the same, scarce resources; this scenario requires the system to recognize and leverage synergies during the concurrent execution of long-running or standing queries;

- **adaptive, seamless deployment:** a wider application of data analysis techniques requires that a data analysis system that fits seamlessly into an existing computer architecture and can cope with and automatically optimize for specific properties of the underlying hardware (e..g, [4]);
- **trading-off virtualization:** modern data analysis systems must properly exploit the increasing availability of multi-core machines despite operating in a virtualized environment; achieving this goal requires us to identify crucial resources and make those components transparent to the runtime system, potentially using methods of paravirtualization in contrast to full virtualization; this aspect will be particularly important when accessing huge, distributed states (I/O and network transparency), as well as the efficient execution of CPU-bound operations;
- **first results fast:** low-latency processing of data streams despite high data ingest rates is a key requirement for the analysis of data from sensor networks, internet of things applications, robotics, or long-running simulations; from the systems side, this requires low startup costs for DAPs with respect to both query compilation and execution (which is particularly challenging for many core CPUs or compute clusters/clouds consisting of a large amount of parallel nodes), avoidance or intelligent hedging of blocking/batching operators, and exploitation of state in long-running queries, in particular in conjunction with novel, algorithmic fault-tolerance schemes (e.g, like in [2]);

optimizing access on "raw" data (in situ processing): many data sets may be too large to be shipped; thus processing data at the location, where it is generated, in its native form, will be a key feature of novel data analysis solutions.

Indeed, the research community **has already** been working on several of these challenges in isolation and has built systems beyond Hadoop**,** such as Stratosphere/Flink [9,10], Spark [6], ePIC [7], Asterix [8], among others. These systems represent an inspiring basis for future research, but will need to be adapted and partially redesigned to deal with the high complexity of the operators and data, in order to enable the broad adoption these systems through data independence and declarative specification.

4. ACKNOWLEDGMENTS

Stratosphere requires a large R&D team, which cannot all be mentioned here. However, I would like to thank all members of my research group, as well as the Stratosphere teams at Humboldt University of Berlin, the Hasso Plattner Institute in Potsdam, the EIT ICT Labs, and our colleagues in the Berlin Big Data Center, as well as the broader Apache Flink community for many valuable discussions and for building a system based on our vision.

5. REFERENCES

[1] A. Alexandrov, R. Bergmann, S. Ewen, et al.: "The Stratosphere Platform for Big Data Analytics," VLDB Journal 05/2014.

[2] S. Schelter, S. Ewen, K. Tzoumas, et al.: "All Roads Lead to Rome: Optimistic Recovery for Distributed Iterative Data Processing," CIKM 2013: 1919-1928.

[3] S. Ewen, K. Tzoumas, M. Kaufmann, et al.: "Spinning Fast Iterative Data Flows," PVLDB 5(11): 1268-1279 (2012).

[4] M. Heimel, V. Markl: "A First Step Towards GPU-assisted Query Optimization," ADMS@VLDB 2012: 33-44.

[5] D. Battré, S. Ewen, F. Hueske, et al: "Nephele/PACTs: programming model and execution framework for web-scale analytical processing," SoCC 2010: 119-130.

[6] M. Zaharia, M. Chowdhury, M. J. Franklin, et al: "Spark: cluster computing with working sets," HotCloud (2010).

[7] D. Jiang, G. Chen, B. C. Ooi, K.-L. Tan, S. Wu: "epiC: an Extensible and Scalable System for Processing Big Data," PVLDB 7(7): 541-552 (2014).

[8] S. Alsubaiee, Y. Altowim, H. Altwaijry, et al: "ASTERIX: An Open Source System for Big Data Management and Analysis." PVLDB 5(12): 1898-1901 (2012).

[9] Stratosphere, http://www.stratosphere.eu, last checked Jul 7, 2014

[10] Apache Flink Incubator Project, http://flink.incubator.apache.org/ last checked Jul 7, 2014