Report from the third workshop on Algorithms and Systems for MapReduce and Beyond (BeyondMR'16)

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

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

Christopher Ré
Stanford University, USA
chrismre@cs.stanford.edu

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

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

ABSTRACT

This report summarizes the presentations and discussions of the third workshop on Algorithms and Systems for MapReduce and Beyond (BeyondMR'16). The BeyondMR workshop was held in conjunction with the 2016 SIGMOD conference in San Francisco, California, USA on July 1, 2016. The goal of the workshop was to bring together researchers and practitioners to explore algorithms, computational models, architectures, languages and interfaces for systems that need large-scale parallelization and systems designed to support efficient parallelization and fault tolerance. These include specialized programming and data-management systems based on MapReduce and extensions, graph processing systems, data-intensive workflow and dataflow systems. The program featured two very well attended invited talks by Ion Stoica from AMPLab, University of California Berkeley and Carlos Guestrin from the University of Washington.

1. INTRODUCTION

The third BeyondMR workshop explored algorithms, computational models, architectures, languages and interfaces for systems that need large-scale parallelization and systems designed to support efficient parallelization and fault tolerance. The list of covered topics includes specialized programming and data-management systems based on MapReduce and extensions, graph processing systems, data-intensive workflow and dataflow systems.

After moving from EDBT to SIGMOD, the workshop successfully attracted 19 submission from which the program committee led by Christopher Ré from University of Stanford accepted 5 regular and 5 short papers.

2. REVIEW OF PRESENTED WORK

The proceedings of the workshop were published in the ACM Digital Library [1]. Below we present a short overview of the results followed, in Section 3, by a summary of two highly attended keynotes.

(Short Paper) Bridging the gap: Towards optimization across linear and relational algebra

Andreas Kunft from TU Berlin, Germany presented this paper [9] on behalf of co-authors Alexander Alexandrov, Asterios Katsifodimos and Volker Markl. Data cleaning and preprocessing is typically an initial step of advanced data analysis pipelines. As a theoretical foundation for the first step a relational algebra is used and for the second step a linear algebra. The authors propose to unify those two algebras into a common theoretical foundation. They explore and reason about optimizations across the two algebras in a suitable intermediate language representation. They propose *Lara DSL*, which is embedded in Scala and offers abstract data types for both algebras, i.e., bags and matrices. They also show-case the added benefits of unification and the optimizations that come thereof. A number of holistic optimizations are derived from the unified formal model and implemented under the assumption of a full view of the algorithm code including matrix blocking through joins and row-wise aggregation pushdown.

(Short Paper) Faucet: a user-level, modular technique for flow control in dataflow engines

Andrea Lattuada from the Systems Group, ETH Zürich, Switzerland presented this paper [10] on behalf of co-authors Frank McSherry and Zaheer

Chothia. This short paper introduces *Faucet*, which is a modular control flow approach for organizing distributed dataflow processing with arbitrary topologies including cyclicity. The advantages of Faucet over backpressure techniques are: (i) the implementation only relies on existing progress information exposed by the system and does not require changes to the underlying dataflow system, (ii) it can be applied selectively to certain parts of the dataflow graph, and (iii) it is designed to support a wide variety of use cases, topologies and workloads. The authors have tested their implementation on an example where variability in rates of produced and consumed tuples challenges the flow control techniques employed by systems like Storm, Heron, and Spark. They were able to keep the computation stable and resource bound while introducing at most 20% runtime overhead over an unconstrained implementation.

(Short Paper) Model-Centric Computation Abstractions in Machine Learning Applications

Judy Qiu from Indiana University, Bloomington, USA presented this paper [22] on behalf of co-authors Bingjing Zhang and Peng Bo. This paper considers parallel machine learning as a combination of training data-centric and model parameter-centric processing. It first presents four types of data-centric computation models for distributed machine learning, where they types are characterized by (1) whether the access of the parallel workers to the parameter models is synchronized or not, and if it is how the order of access is determined and (2) whether the workers get access to only the latest model parameters or also to stale model parameters. Several existing systems for distributed machine learning are analyzed and classified according to the presented types of computation models. Subsequently new model-centric abstractions are introduced to improve model update rate and increase model convergence speed. The effectiveness of these abstractions is demonstrated by using Latent Dirichlet Allocation (LDA) as an example, and experimental results show that an efficient parallel model update pipeline can achieve similar or higher model convergence speed compared to existing work.

(Regular Paper) DFA Minimization in Map-Reduce

Gösta Grahne from Concordia University, Montreal, Canada presented this paper [6] on behalf of co-authors Shahab Harrafi, Iraj Hedayati and Ali Moallemi. It features MapReduce implementations of two of the most prominent DFA minimization methods, namely Moore's and Hopcroft's algorithms. Extensive experiments, on various types of DFA's, with up to 217 states, validate that the MapReduce implementation of Hopcroft's algorithm is more efficient, both in terms of running time and communication cost. It was also confirmed that both algorithms are sensitive to skewed input, the Hopcroft's algorithm being intrinsically so.

(Regular Paper) Cross-System NoSQL Data Transformations with NotaQL

Johannes Schildgen from the University of Kaiserslautern presented this paper [16] on behalf of co-authors Thomas Lottermann and Stefan Deßloch. This full paper presents the language NotaQL which allows to concisely express transformations between different NoSQL data formats as are found in wide-column stores, document stores, key-value stores and even CSV files. The language supports a range of input and output formats, as well as different transformation engines for these formats. The language is output-oriented in the sense that the output format determines the structure of the transformation expressions. Finally, the paper presents an implementation of this language based on Apache Spark.

(Regular Paper) On Exploring Efficient Shuffle Design for In-Memory MapReduce

Haronubo Daikoku from University of Tsukuba, Japan presented this paper [4] co-authored with Hideyuki Kawashima and Osamu Tatebe. The authors have studied the efficiency of shuffle phase in MapReduce type systems that run on supercomputer hardware with shared-memory multiprocessor like InfiniBand. There are several design decisions which need to be made in such implementations to adapt MapReduce from commodity hardware communicating over Ethernet to specialized hardware relaying on MPI-based communication. The authors have implemented their own in-memory MapReduce system in C/C++ and used it to compare the efficiency of the data exchange algorithms in the shuffle phase. Specifically they have tested a fully-connected algorithm that mimics standard MapReduce solutions where each reduce process maintains a link to all map processes and a pairwise algorithm where in subsequent steps the processes communicate in pairs. They also have analyzed the effect on shuffle phase of the Remote Direct Memory Access (RDMA) mechanism which enables one machine to read and write data on the local memory of another.

(Short Paper) Toward Elastic Memory Management for Cloud Data Analytics

Jingjing Wang from University of Washington, USA presented this paper [19] co-authored with Magdalena Balazinska. The short paper discusses elastic memory management in modern Big data systems. It starts with demonstrating the negative impact of GC on the execution time of data analytics queries in a modern, Java-based system and shows how changing the heap size directly impacts the execution time. Then, it describes how to modify the JVM to enable dynamic modifications of the application heap layout and thus allow elastic management of its memory utilization. Next, it presents a machine-learning based technique for predicting the GC overhead for an application and whether that application is expected to run out of memory. Finally, an algorithm for dynamic memory management in a Big data analytics system is discussed.

(Regular Paper) Some-Pairs Problems

Jeffrey Ullman from Stanford University, USA presented this paper [17] co-authored with Jonathan Ullman. The paper considers the "some pairs" problem, where we are given two sets X and Y, and wish to detect the presence of pairs (x, y), one from each set, that meet some criterion, e.g., x and y are sufficiently close according to some distance measure. This paper looks at MapReduce algorithms for solving such a problem, in particular looking at the reducer-size vs. replication-rate tradeoff from Sarma et al, VLDB, 2013 [15]. There are two obvious approaches: (1) assume you care about all pairs and don't worry about taking advantage of the fact that you only care about some pairs (2) use one reducer for each of the pairs you care about, and nothing else. The principal result of the paper is a proof that for any X and Y and subset of the pairs that you care about, there is no MapReduce algorithm that has a significantly better replication rate than the better of the two obvious approaches.

(Regular Paper) Tight Bounds on One- and Two-Pass MapReduce Algorithms for Matrix Multiplication

Prakash Ramanan from Wichita State University, Wichita, USA presented this paper [13] co-authored with Ashita Nagar. This paper studies one- and two-pass MapReduce algorithms for multiplying two matrices, and in particular the trade-off between communication cost and replication rate. For multiplying sparse matrices in one pass, it shows tight bounds on qr and wr^2 where q is the reducer size, r the replication rate and w the reducer workload. In fact, the work shows that the bound for qr follows from the bound for wr^2, which means that the latter is the stronger lower bound. Next, the paper considers two-pass algorithms, which have been shown to have less communication cost than one-pass algorithms, given a certain reducer size. For multiplying dense matrices it presents tight bounds on $q_f r_f r_s$ and $w_f r_f^2 r_s$, where the subscripts f and s correspond to the first and second pass, respectively. Using this bound on $q_f r_f r_s$, the paper presents a tight bound on the total communication cost as a function of g_f. The presented lower bounds hold for the two-pass algorithms that perform all the real-number multiplications in the first pass.

(Short Paper) Deterministic Load Balancing for Parallel Joins

Nivetha Singara Vadivelu from University of Wisconsin-Madison, USA presented this short paper [8] co-authored with Paraschos Koutris. This short paper discusses parallel joins and multiway joins where the input data is first distributed over r-dimensional hypercube and then blocks in this cube can be processed independently in parallel. There was already a lot of attention to this problem and efficient solutions were proposed that distribute the tuples by applying a random hash function to achieve with high probability the optimal load within polylogarithmic factor. In the paper the authors explore if it is possible to construct an efficient deterministic algorithm that distributes the tuples such that the load is always as close to the optimal value as possible. They also seek to obtain optimality guarantees under any skew conditions, and not only for the case of no data skew. A general lower bound is proposed for the load, which is based on maximum degrees of each value (or combination of values) in the relation. Then, two fast deterministic algorithms are presented: one that is optimal within a constant factor of the lower bound for one dimensional case and another one that is optimal within a polylogarithmic factor of the lower bound for two dimensional case. The second one extends the first with application of algorithm for vector load balancing problem.

3. SUMMARY OF KEYNOTES

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

(Keynote) Spark: Past, Present, and Future

In his keynote, Ion Stoica from AMPLab, University of California Berkeley gave an overview of the decisions that, apart of timing and luck, lead to the

success of the Spark project [21]. Besides an expressive API that allows to reduce by several times the length of code needed for typical tasks as compared to Hadoop, the main advantage of Spark is its effectiveness. It comes from leveraging hardware and workload trends like the rapid increase in memory capacity, so that working sets in Big Data clusters fit in memory. Further efficiency gains come from using threads rather than JVM processes and dealing with fault recovery with lineage rather than persistent storage. Those ideas lead to significant speed-up in many use cases, especially in interactive processing which is often needed in machine learning applications.

Then prof. Stoica gave an overview of the Spark subprojects: *Spark SQL* [2], *Spark Streaming* [20], *MLlib* [12], *GraphX* [5] and *SparkR* [18] and highlighted recent developments including structured APIs (*Datasets* and *DataFrames*), *project Tungsten* and structured streaming. The first phase of project Tungsten was enabled by DataFrames and results in 5-20× speed up. It exploits cache locality and employs off-heap memory management. The second phase of project Tungsten introduces whole-stage code generation, which removes expensive iterator calls and allows fusing across multiple operators.

The future of Spark brings improvements in performance due to fine grain updates with *IndexedRDDs*, reducing latency with batch scheduling, generality with fine grain task computations, and finally easy of use.

(Keynote) Big Data, Small Cluster: Choosing 'big memory' (RAM, disks, SSDs) over big clusters

Carlos Guestrin from the University of Washington was the second keynote speaker. Although initially he was interested in constructing killer robots, his presentation was about the ideas from the database community that have surfaced in the machine learning community and influence its progress. The first of those ideas is columnar storage and compression of stored data, which allowed for huge speed improvements in projects that Carlos worked on in the past like GraphLab [11]. The second idea is the quantile sketch technique [7, 23] that was adapted to weighted datasets to allow for the development of the scalable end-to-end tree boosting system [3].

Next, he presented an analogy between the current aim in machine learning community to create composites systems and Database Managements System (DBMS) that freed database users from technical decisions like optimizing queries, planing indexes and their usage or using materialized views. Building a machine learning solution nowadays also requires many technical decisions and skills like model selection, parameter selection or implementing distributed execution on a cluster. It would be convenient if a composite solution would be created with build in machine learning algorithms, so that the user could only declare what he needs and the system would analyze the data and make appropriate technical decisions for him.

Finally, the last idea presented by the speaker is related to *provenance*. As machine learning adoption is sometimes slowed down, by lack of *trust* in the results [14], it would be helpful to understand the predictions and get their explanations. This would also allow to avoid the situations where the accuracy percentage is high but the features that are used to achieve this accuracy are only properties of the training set, and in practical scenarios this would not generalize and could lead to incorrect behavior. Furthermore, also in situations where the results are correct, the users would profit from understanding how they were achieved, e.g., that the user could like this new movie because he liked some other or that a patient should be diagnosed with a disease because this is suggested by a given subset of his medical examination results.

4. CONCLUSION

The presentations and keynotes at BeyondMR'16 provided an overview of current developments and emerging issues in the domain of algorithms, computational models, architectures, languages and interfaces for systems that need large-scale parallelization and systems designed to support efficient parallelization and fault tolerance. These proceedings suggest that while MapReduce was replaced by new models, there is an active area of research centered around data-management systems based on MapReduce and extensions, graph processing systems, data-intensive workflow and dataflow systems.

Acknowledgements: We would like to thank the PC members, keynote speakers, authors, local workshop organizers and attendees for making BeyondMR'16 a successful workshop. We also express our great appreciation for the support from Google Inc.

5. REFERENCES

[1] Foto N. Afrati, Jacek Sroka, and Jan Hidders, editors. *Proceedings of the 3rd ACM SIGMOD Workshop on Algorithms and Systems for MapReduce and Beyond, BeyondMR@SIGMOD 2016, San Francisco,*

CA, USA, July 1, 2016. ACM, 2016. http://doi.acm.org/10.1145/2926534.

[2] Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley, Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, and Matei Zaharia. Spark SQL: relational data processing in spark. In Timos K. Sellis, Susan B. Davidson, and Zachary G. Ives, editors, *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 1383–1394. ACM, 2015.

[3] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

[4] Harunobu Daikoku, Hideyuki Kawashima, and Osamu Tatebe. On exploring efficient shuffle design for in-memory mapreduce. In Afrati et al. [1], page 6. http://doi.acm.org/10.1145/2926534.2926538.

[5] Joseph E. Gonzalez, Reynold S. Xin, Ankur Dave, Daniel Crankshaw, Michael J. Franklin, and Ion Stoica. Graphx: Graph processing in a distributed dataflow framework. In Jason Flinn and Hank Levy, editors, *11th USENIX Symposium on Operating Systems Design and Implementation, OSDI '14, Broomfield, CO, USA, October 6-8, 2014.*, pages 599–613. USENIX Association, 2014.

[6] Gösta Grahne, Shahab Harrafi, Iraj Hedayati, and Ali Moallemi. DFA minimization in map-reduce. In Afrati et al. [1], page 4. http://doi.acm.org/10.1145/2926534.2926537.

[7] Michael Greenwald and Sanjeev Khanna. Space-efficient online computation of quantile summaries. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, SIGMOD '01, pages 58–66, New York, NY, USA, 2001. ACM.

[8] Paraschos Koutris and Nivetha Singara Vadivelu. Deterministic load balancing for parallel joins. In Afrati et al. [1], page 10. http://doi.acm.org/10.1145/2926534.2926536.

[9] Andreas Kunft, Alexander Alexandrov, Asterios Katsifodimos, and Volker Markl. Bridging the gap: towards optimization across linear and relational algebra. In Afrati et al. [1], page 1. http://doi.acm.org/10.1145/2926534.2926540.

[10] Andrea Lattuada, Frank McSherry, and Zaheer Chothia. Faucet: a user-level, modular technique for flow control in dataflow engines. In Afrati et al. [1], page 2. http://doi.acm.org/10.1145/2926534.2926544.

[11] Yucheng Low, Joseph Gonzalez, Aapo Kyrola, Danny Bickson, Carlos Guestrin, and Joseph M. Hellerstein. Graphlab: A new framework for parallel machine learning. In Peter Grünwald and Peter Spirtes, editors, *UAI 2010, Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, July 8-11, 2010*, pages 340–349. AUAI Press, 2010.

[12] Xiangrui Meng, Joseph K. Bradley, Burak Yavuz, Evan R. Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, D. B. Tsai, Manish Amde, Sean Owen, Doris Xin, Reynold Xin, Michael J. Franklin, Reza Zadeh, Matei Zaharia, and Ameet Talwalkar. Mllib: Machine learning in apache spark. *CoRR*, abs/1505.06807, 2015.

[13] Prakash Ramanan and Ashita Nagar. Tight bounds on one- and two-pass mapreduce algorithms for matrix multiplication. In Afrati et al. [1], page 9. http://doi.acm.org/10.1145/2926534.2926542.

[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM.

[15] Anish Das Sarma, Foto N. Afrati, Semih Salihoglu, and Jeffrey D. Ullman. Upper and lower bounds on the cost of a map-reduce computation. In *Proceedings of the 39th international conference on Very Large Data Bases*, PVLDB'13, pages 277–288. VLDB Endowment, 2013.

[16] Johannes Schildgen, Thomas Lottermann, and Stefan Deßloch. Cross-system NoSQL data transformations with NotaQL. In Afrati et al. [1], page 5. http://doi.acm.org/10.1145/2926534.2926535.

[17] Jeffrey D. Ullman and Jonathan R. Ullman. Some pairs problems. In Afrati et al. [1], page 8. http://doi.acm.org/10.1145/2926534.2926543.

[18] Shivaram Venkataraman, Zongheng Yang, Davies Liu, Eric Liang, Hossein Falaki, Xiangrui Meng, Reynold Xin, Ali Ghodsi,

Michael J. Franklin, Ion Stoica, and Matei Zaharia. Sparkr: Scaling R programs with spark. In Fatma Özcan, Georgia Koutrika, and Sam Madden, editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 1099–1104. ACM, 2016.

[19] Jingjing Wang and Magdalena Balazinska. Toward elastic memory management for cloud data analytics. In Afrati et al. [1], page 7. `http://doi.acm.org/10.1145/2926534.2926541`.

[20] Matei Zaharia, Tathagata Das, Haoyuan Li, Timothy Hunter, Scott Shenker, and Ion Stoica. Discretized streams: fault-tolerant streaming computation at scale. In Michael Kaminsky and Mike Dahlin, editors, *ACM SIGOPS 24th Symposium on Operating Systems Principles, SOSP '13, Farmington, PA, USA, November 3-6, 2013*, pages 423–438. ACM, 2013.

[21] Matei Zaharia, Reynold S. Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J. Franklin, Ali Ghodsi, Joseph Gonzalez, Scott Shenker, and Ion Stoica. Apache spark: a unified engine for big data processing. *Commun. ACM*, 59(11):56–65, 2016.

[22] Bingjing Zhang, Bo Peng, and Judy Qiu. Model-centric computation abstractions in machine learning applications. In Afrati et al. [1], page 3. `http://doi.acm.org/10.1145/2926534.2926539`.

[23] Qi Zhang and Wei Wang. A fast algorithm for approximate quantiles in high speed data streams. In *Proceedings of the 19th International Conference on Scientific and Statistical Database Management*, SSDBM '07, pages 29–29, Washington, DC, USA, 2007. IEEE Computer Society.