Requirements for Meaningful Big Data Benchmarking

Michael J. Carey
Information Systems Group
Computer Science Department
University of California, Irvine
mjcarey@ics.uci.edu

We started the NSF-sponsored ASTERIX project [2,3,4,5,6] at UC Irvine in Fall 2009. Our goal at the outset was to design and implement a highly scalable platform for information storage, search, and analysis. By combining and extending ideas drawn from semistructured data management [1], parallel database systems [11], and first-generation data-intensive computing platforms (primarily MapReduce [10] and Hadoop [13]), the ASTERIX system was envisioned to be a parallel, semistructured information management platform with the ability to ingest, store, index, query, analyze, and publish very large quantities of semistructured data. ASTERIX is intended to handle use cases ranging all the way from rigid, relation-like data collections, whose types are well understood and invariant, to flexible and more complex data, where little is known a priori and the instances in data collections are highly variant and self-describing. Its data is accessible via both a full query-based API and a simpler key-value interface, and it can access both externally stored data (e.g., in HDFS) as well as its own natively stored, managed, and indexed data. Our eventual goal is to deliver a shareable open-source system that is superior in both offered functionality and performance as compared to today's popular first-generation data-intensive computing platforms and tools. Benchmarking will play an important role in assessing the degree to which we succeed at meeting this goal.

Prior to the ASTERIX project launch, in the summer of 2009, we visited a small handful of our Big Data industrial friends in order to share our initial plans and get feedback on issues they felt mattered and should be included in our research agenda. This industry tour included stops at eBay, Facebook, and Teradata, and it has definitely informed our ASTERIX research planning. One of the key messages that we got at all three stops, albeit paraphrased, was: "Academics mostly focus on running one query as fast as possible. This is ***not*** our world or our systems' workloads. Our clusters all run complex, multi-user workload mixes that include jobs of different sizes and importance. Multi-user performance and effective scheduling and resource management for such workloads is thus a must-examine issue." Benchmarking will clearly play an important role in assessing emerging Big Data technologies, algorithms, and platforms from this important angle.

Big Data benchmarking is in its infancy in this regard, with very little having been done (and virtually none of it done very "well") in terms of multi-user evaluations of such platforms. The first benchmarking effort in the Big Data space was the work of Pavlo *et al* [15], which compared Hadoop with several commercial parallel relational DBMS offerings. Being a first study, it was necessarily rather simple (yet very interesting); it was strictly a single-user study examining alternative technologies for Big Data analytics. Its treatment of internal vs. externally managed data (and the cost of loading, indexing, and querying) was (and remains) somewhat controversial, raising an interesting issue for future efforts in this space. The other best-known benchmark in the Big Data space is probably YCSB, the Yahoo! Cloud-Serving Benchmark [9]. YCSB was a first-study of the performance of NoSQL [8] storage alternatives, studying their behavior for multiuser mixes of get, put, and delete operations. The workloads used in the YCSB work were homogeneous and rather "peculiar". Multiple job classes and important performance issues raised by their presence were not considered. Instead, the YCSB study's workload modeled an open system (in the queueing theory workload sense), with the results being presented as global throughout (arrival rate) *vs.* response time, i.*e.,* in a form that is appropriate only for the analysis of open queueing

systems with homogeneous, single-class workloads.

Moving forward, it is imperative that the Big Data benchmarking community consider multi-class, multi-user workloads and develop, employ, and share appropriate tools and methodologies for evaluating emerging platforms under such workloads. In doing so, the community may need to take a look at other work on the performance evaluation of algorithms where multi-class workloads have been key. For example, the experimental section of [7], an old paper on evaluating multiversion concurrency control techniques, includes a discussion of issues related to the modeling and study of multi-class workloads. At UCI, a graduate student who spent the summer of 2010 "hacking Hadoop" at Facebook conducted a simple multi-user, multi-class study of the resulting Hadoop modifications. The performance aspects of that work, reported in [12], required significant thought, effort, and multiple iterations, and its results provide a good illustration of the importance of considering multi-user workloads and multi-class workloads when comparing scheduling (and related) algorithms. In fact, an interesting side effect of that study is that it called into question the true efficacy of the much-mentioned "Fair Scheduler" component of Hadoop, an optional "improved" scheduler for complex Hadoop workloads [13]. That study's findings were that, although it may be "fair", its overall performance was relatively poor (for our workloads' job mixes) as compared to the stock Hadoop scheduler, which we found somewhat surprising. Also moving forward, there will be a clear requirement for multiple benchmarks aimed at different "Big Data" use cases and workloads, e.g., analytic mixes, real-time key-value mixes, perhaps streaming ("Fast Data") mixes, as well as benchmarks for new specialized platforms such as graph analytics [14] platforms - much like how the various TPC-X benchmarks drove performance progress in the RDBMS community over the years. Clearly, there is much interesting work to be done...!

References

1. S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, San Mateo, 1999.
2. S. Alsubaiee *et al*. ASTERIX: Scalable Warehouse-Style Web Data Integration, *Proc. iiWeb '12 Workshop*, May 2012.
3. *ASTERIX Project Website.* http://asterix.ics.uci.edu/.
4. A. Behm *et al.* ASTERIX: Towards a Scalable, Semistructured Data Platform for Evolving-World Models. *Distributed and Parallel Databases*, 29(3), 2011.
5. V. Borkar *et al*. Hyracks: A flexible and extensible foundation for data-intensive computing, *Proc. ICDE '11 Conf*, April 2011.
6. V. Borkar, M. Carey, and C. Li. Inside "Big Data Management": Ogres, Onions, or Parfaits?. *Proc. EDBT '12 Conf.*, Mar. 2012.
7. M. Carey and W. Muhanna. The Performance of Multiversion Concurrency Control Algorithms. *ACM Trans. on Comp. Sys.* 4(4), Nov. 1986.
8. R. Cattell. Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*, May 2011.
9. B. Cooper, A.Silberstein, E. Tam, R. Ramakrishnan, and R. Sears. 2010. Benchmarking cloud serving systems with YCSB. In *Proc. ACM SoCC '10 Conf.,* June 2010.
10. J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. *Proc. OSDI '04 Conf.,* Dec. 2004.
11. D. DeWitt and J. Gray. Parallel Database Systems: The Future of High Performance Database Systems. *Comm. ACM* 35(6), 1992.
12. R. Grover and M. Carey. Extending Map-Reduce for Efficient Predicate-Based Sampling. *Proc. ICDE' 12 Conf.,* April 2012.
13. *Hadoop Project Website.* http://hadoop.apache.org/.
14. G. Malewicz *et al*. Pregel: A System for Large-Scale Graph Processing. *Proc. ACM SIGMOD' 10 Conf.*, June 2010.
15. A. Pavlo *et al*. A Comparison of Approaches to Large-Scale Data Analysis. *Proc. ACM SIGMOD '09 Conf.*, June 2009.