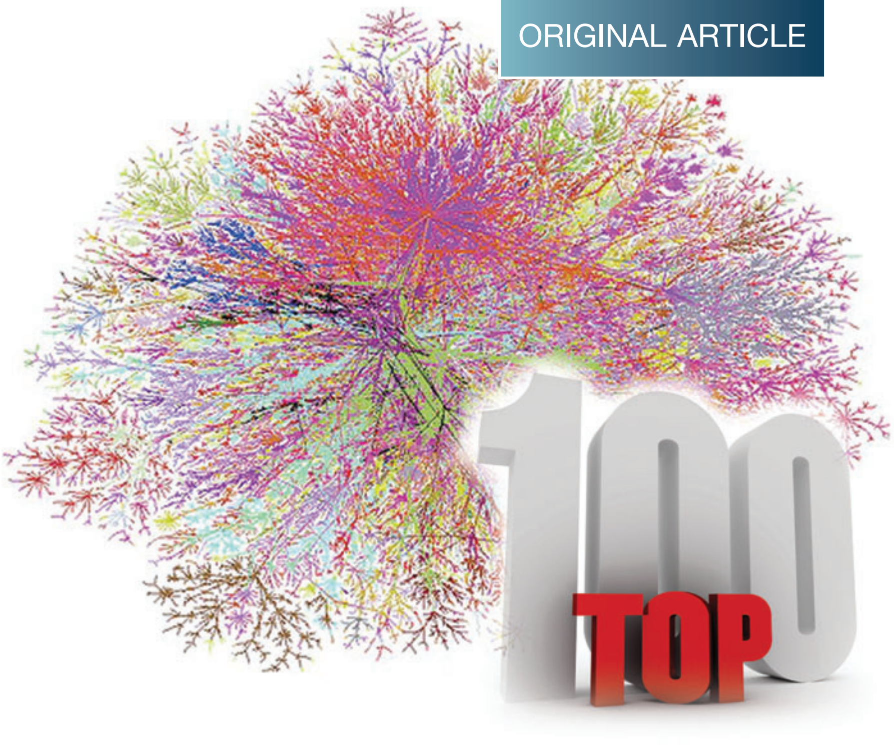
BENCHMARKING BIG DATA SYSTEMS AND THE BIGDATA TOP100 LIST



Chaitanya Baru,[1] *Milind Bhandarkar,*[2]
Raghunath Nambiar,[3] *Meikel Poess,*[4]
and Tilmann Rabl[5]

Abstract

"Big data" has become a major force of innovation across enterprises of all sizes. New platforms with increasingly more features for managing big datasets are being announced almost on a weekly basis. Yet, there is currently a lack of any means of comparability among such platforms. While the performance of traditional database systems is well understood and measured by long-established institutions such as the Transaction Processing Performance Council (TCP), there is neither a clear definition of the performance of big data systems nor a generally agreed upon metric for comparing these systems. In this article, we describe a community-based effort for defining a big data benchmark. Over the past year, a Big Data Benchmarking Community has become established in order to fill this void. The effort focuses on defining an end-to-end application-layer benchmark for measuring the performance of big data applications, with the ability to easily adapt the benchmark specification to evolving challenges in the big data space. This article describes the efforts that have been undertaken thus far toward the definition of a BigData Top100 List. *While highlighting the major technical as well as organizational challenges, through this article, we also solicit community input into this process.*

Introduction

While a mature data-management industry with a robust set of techniques and technologies has been established over the last couple of decades, the emergence of the big data phenomenon over the past few years, with its increased *volume*, *velocity*, and *variety* of data and a requirement for agile development of data-driven applications, has created a new set of challenges. The advent of new techniques and technologies for big data creates the imperative for an industry standard for evaluating such systems. The big data benchmarking activity described in this article was initiated for this very purpose—to provide academia with a way to evaluate new techniques for big data in a realistic setting; industry with a tool to drive development; and customers with a standard way to make informed decisions about big data systems.

Beginning in late 2011, the Center for Large-scale Data Systems Research (CLDS) at the San Diego Supercomputer Center, University of California San Diego, in collaboration with several industry experts, initiated a community activity in big data benchmarking, with the goal of defining *reference benchmarks* that capture the essence of big data application

[1] *San Diego Supercomputer Center; University of California, San Diego; La Jolla, California.*
[2] *Greenplum, EMC, San Mateo, California.*
[3] *Cisco Systems, Inc., San Jose, California.*
[4] *Oracle Corporation, Redwood City, California.*
[5] *University of Toronto, Toronto, Canada.*

DOI: 10.1089/big.2013.1509

scenarios. The goal of this activity is to provide clear objective information to help characterize and understand hardware and system performance and price/performance of big data platforms. A workshop series on big data benchmarking (WBDB) was organized, sponsored by the National Science Foundation (http://clds.sdsc.edu/bdbc/workshops). The new big data benchmark should characterize the new feature sets, large data sizes, large-scale and evolving system configurations, shifting loads, and heterogeneous technologies of big data platforms. The first workshop, held on May 8–9, 2012, in San Jose, developed a number of initial ideas[1] and was followed by subsequent meetings (see http://clds.sdsc.edu/events/nov-02-2012), and the second workshop was held on December 17–18, 2012, in Pune, India (http://clds.sdsc.edu/wbdb2012.in). These meetings substantiated the initial ideas for a big data benchmark, which would include definitions of the data along with a data-generation procedure; a workload representing common big data applications; and a set of metrics, run rules, and full-disclosure reports for fair comparisons of technologies and platforms. These results would then be presented in the form of the *BigData Top100 List*, released on a regular basis at a predefined venue such as at the Strata Conferences.

"THESE MEETINGS SUBSTANTIATED THE INITIAL IDEAS FOR A BIG DATA BENCHMARK, WHICH WOULD INCLUDE DEFINITIONS OF THE DATA ALONG WITH A DATA-GENERATION PROCEDURE."

The BigData Top100 List would pursue a *concurrent benchmarking* model, where one version of the benchmark is implemented while the next revision is concurrently being developed, incorporating more features and feedback from the first round of benchmarking. While this will create different versions of the same benchmark, we believe that the community is sufficiently mature to be able to interpret benchmark results in proper context. Indeed, each release of the benchmark may also be accompanied by a set of issues under design/consideration for the next release, so that the community is made fully aware of the benchmark development activity. Our goal is to pursue this *open* benchmark-development process, soliciting input from the community at large, to be evaluated by a *benchmark steering committee* with representation from industry, academia, and other sectors.

In the final analysis, results from an industry-standard benchmark are only the first—though important—step toward understanding system performance. A user/customer may then run their proprietary benchmarks to complement the open benchmark results.

Characteristics of a Big Data Benchmark

We propose the BigData Top100 List as an *application-level benchmarking* exercise to provide an "end-to-end" view of big data applications. In contrast, *functional benchmarks* focus on specific functions (e.g., TeraSort); *data-genre benchmarks* focus on operations of specific genres of data (e.g., Graph 500); while *micro-benchmarks* focus on lower-level system operations.[2] While TPC benchmarks are also at the application-level, they focus on highly structured (relational) data and are restricted to the functionality strictly provided by Structured Query Language (SQL). We have developed guidelines, as described below, for defining a big data benchmark.

- *Simplicity:* Following the dictum that "Everything should be made as simple as possible, but no simpler," the benchmark should be technically simple to implement and execute. This is challenging, given the tendency of any software project to overload the specification and functionality, often straying from the most critical and relevant aspects.
- *Ease of benchmarking:* The costs of benchmark implementation/execution and any audits should be kept relatively low. The benefits of executing the benchmark should justify its expense—a criterion that is often underestimated during benchmark design.
- *Time to market*: Benchmark versions should be released in a timely fashion in order to keep pace with the rapid market changes in the big data area. A development time of 3 to 4 years, common for industry consortia, would be unacceptable in the big data application space. The benchmark would be outdated and obsolete before it is released!
- *Verifiability of results:* Verification of results is important, but the verification process must not be prohibitively expensive. Thus, to ensure correctness of results while also attempting to control audit costs, the BigData Top100 List will provide for automatic verification procedures along with a peer-review process via a *benchmark steering committee* to ensure verifiability of results.

Benchmark escalation

Benchmark escalation refers to the tendency of benchmark sponsors to assemble ever-larger systems solely in order to obtain a better benchmark result. Since we wish to favor innovative approaches to solving big data challenges rather than simply assembling larger systems, we wish to introduce mechanisms to discourage benchmark escalation. Different methods have been attempted for dealing with this problem, for example, by directly enforcing restrictions on the system size or by indirectly accounting for size via benchmark metrics that reward the efficiency of a system. A simple form of direct restriction, for example, is capping the "total cost" of a system, say, to $100K. However, such restrictions are viewed as arbitrary and—in the case of system cost—difficult to define. Furthermore, they become quickly outdated given the pace of technology. A better approach would be to reward system efficiency, which could be done along various dimensions—cost, energy, data center space, and "processing efficiency" (amount of actual "work" done by the system versus its peak performance). Thus, along with performance, it is also important to report "efficiency," for example, as reported by measuring the performance per dollar, watt, square/cubic foot, or say, peak FLOPS. We believe that such measures of efficiency would foster more innovative system designs as opposed to "raw" scaling up of the hardware systems. We solicit suggestions for approaches to controlling the phenomenon of such benchmark escalation.

Big Data Benchmark Proposals

Big data systems are characterized by their flexibility in processing diverse data genres, such as transaction logs, connection graphs, and natural language text, with algorithms characterized by multiple communication patterns (e.g., scatter-gather, broadcast, multicast, pipelines, and bulk-synchronous). Thus, it would appear that a single benchmark that characterizes a single workload could not be representative of such a multitude of use-cases. However, an informal survey of several use-cases of current big data platforms indicates that most workloads are composed of a common set of stages, which capture the variety of data genres and algorithms commonly used to implement most data-intensive end-to-end workloads. Thus, we propose a workload specification based on stitching together the various stages of processing into an end-to-end *entity-modeling pipeline.*

"A BETTER APPROACH WOULD BE TO REWARD SYSTEM EFFICIENCY, WHICH COULD BE DONE ALONG VARIOUS DIMENSIONS."

A broad range of data-driven industries are engaged in attempting to learn the behavior of entities and the events of interest to them. For example, the online advertising industry is trying to make sense of *user activities* that correlate with the event of interest to them, viz. a click on an online advertisement. The banking industry is trying to predict *customer churn* based on the customer data (demographics, income) and interaction patterns that are available to them. The insurance industry is trying to predict *fraud* based on the data about their customers' activities, while the healthcare industry is trying to predict a patient's propensity to visit the emergency room, and the *need for preventive care*, based on patient data. All of these use-cases involve collecting a variety of datasets about the entities of interest to the organization, and detecting correlations between the outcome of interest and past behavior. The "user modeling" pipeline is, therefore, a typical use-case for current big data workloads and, thus, helps define the workload for our benchmark.

Such a modeling pipeline consists of several stages, either performed on a single platform, or distributed across different platforms, based on each platform's capabilities, cost, operational efficiency, scale, and performance. Each stage is, therefore, described in terms of its functionality rather than in platform-specific terms. Indeed, the notion of "polyglot-persistence" prevalent in the emerging big data platforms[3] points toward such a specification.

A big data analytics pipeline-based workload specification

Step 1: Collect "user" interactions data and ingest them into the big data platform(s). User interaction logs are collected in time-order, closest to the systems that enable these interactions, such as web servers, call centers, or any other medium that allows such interaction. If the "user" is a machine, the syslog/sensor data collector aggregates these interaction events on local storage very near the individual collectors. These "logs" are then ingested in the big data platforms, in the same format as the collector, or with very little transformations, such as timestamp corrections. These logs are ordered according to when they were recorded, i.e., time-stamping, with some granularity.

Step 2: Reorder the logs/events according to the entity of interest, with secondary ordering according to timestamps. Thus, a syslog collection is initially ordered by local timestamps, which is converted to global timestamp and then ordered (or sessionized) by machine identifier. Similarly, users' click/view streams are ordered initially by the website (httpd) logs, but have to be reordered and sessionized according to the end-user, identified by, say, a browser cookie or other user identification information.

Step 3: Join the "fact tables" with various other "dimension tables." This involves parsing the event data (other than timestamp/user identification), and extracting event-specific information that forms the feature in each of the events. This is the step that incorporates the *late binding* feature often encountered in big data applications, since the features to be extracted may be different for different applications. For example, in a news aggregator site, this may involve distilling the URL pointing to a news item to topics in the news, or in the case of machine logs, distilling the specific machine services indicated by the log message.

Step 4: Identify events of interest that one plans to correlate with other events in the same session for each entity. In case of an ad-funded service, this target event is to identify ad-clicks; for datacenter management systems, the target events are abnormalities in the logs, such as machine failure; for Facebook-like systems, the target events are "likes"; in

Twitter-like systems, the target events are re-tweets/favorites; in LinkedIn-like systems, the target events are connections being established; in various subscriber-based organizations, the target event is opening an account, signing onto notification lists, etc.

Step 5: Build a model for favorable/unfavorable target events based on the past session information. Various modeling techniques are employed in this step and, depending upon the sophistication of the modeling team and the platform, increasingly complex models may be built. However, quite often, an ensemble of simple models is the preferred approach.

Step 6: Score the models built in the previous step with the hold-out data. Hold-out data is part of the total dataset available for training models that is not used for training these models but only for validating these models.

Step 7: Assuming the validation step with hold-out data passed, this step is to apply the models to the initial entities, which did not result in the target event. For example, in the news aggregation site, since the target event was to click on the news item, this model-scoring step will be applied to all the users who did not click on any news event that was shown.

"HOWEVER, QUITE OFTEN, AN ENSEMBLE OF SIMPLE MODELS IS THE PREFERRED APPROACH."

Step 8: Publish the model for each user to the online serving system so that the model could be applied to that user's activities in real time.

The benchmark specification will include a workload for each of these eight steps, with several classes, for example, for defining the session period (and therefore number of events per user), number of users, and number of models built.

A TPC DS-based workload specification

An alternative approach to defining the big data workload is based on extending an existing TPC benchmark, viz. TPC-DS,[4] with semi-structured and unstructured data, and correspondingly altering the TPC-DS workload to incorporate queries that target these parts of the database. An initial proposal was presented at the first WBDB workshop.[5] A refined version, which includes implementations for the data generation of semi- and unstructured data, a sizing model, an execution model, and a concrete workload that covers the semi- and unstructured data, was presented at the second WBDB workshop.[6] In the proposal, the TPC-DS warehouse data model is extended by semi-structured web server log data and unstructured item review text. A set of queries is defined that consists of traditional SQL queries, similar to the TPC-DS queries; procedural queries that are not easy to implement directly in SQL, for example, for sentiment analysis; and a mix of both. This proposal also covers a wide range of typical big data challenges.

We are soliciting input from the community between these two approaches to defining the first version of the big data benchmark.

Running the BigData Top100 benchmark

Given the fast-moving nature of the field, it is likely that the execution criteria for the BigData Top100 List will evolve, especially early in the process as we receive and incorporate community input and make progress toward a more steady state. The list would be published for each revision of the benchmark specification, similar to the Sort Benchmark model (www.sortbenchmark.org). We are considering restrictions on the benchmark as follows. First, the benchmark should incorporate cost of the overall systems, for example, the total system cost of hardware, software, and a one-year 24/7 support. The vendors must guarantee the price for every priced component for 1 year from the date of publication. For example, if the total system cost is set to, say, $100K, then the benchmark sponsor must pick a configuration priced at $100K or less. Second, the benchmarks would be run at specific "scale factors," that is, the size of the core dataset, similar to the scale factors in TPC-H and TPC-DS benchmark. To ensure data consistency, a cross-platform data-generation program and scripts would be provided for generating the reference dataset at the given scale factor.

There are four key steps to executing the benchmark:

1. System setup: Configure and install the system under test (SUT). This time is not included in the benchmark metric.
2. Data generation: Generate the dataset that meets the benchmark specification. This time is not included in the benchmark metric.
3. Data load: Load the data into the system. This time is included in the benchmark metric.
4. Execute application workload: Run the specified big data workload consisting of a set of queries and transactions. This time is included in the benchmark metric.

The benchmark metric is often one of the most debated topics in the benchmark development process. The general consensus is to have a simple metric that can be recorded easily and that is also easily understood by the users of the benchmark. A simple metric is *total time*, i.e., the wall-clock time taken to complete Steps 3 and 4 above. The sponsor must run the benchmark three times to guarantee repeatability; the run-to-run variation must be within 2%—a number that is chosen arbitrarily; and the reported total time must be the slowest of the three runs. Results in the BigData

Top100 List would then be ordered by total time, with efficiency reported as a secondary figure/metric.

Thus, in this proposed approach, the system at the top of the BigData Top100 List would be the one that can process the representative big data workload on a dataset of fixed size in the least amount of total time (including initial data load time and application workload execution time) procured on a fixed budget, as specified by the benchmark. The specified system cost could be revised with each revision of the benchmark specification.

The detailed run reports, including full-disclosure reports, would have to be submitted to a steering committee for peer review. The full-disclosure report would be required to include all steps to reproduce the benchmark and a corresponding price quote valid for 1 year from the date of publication. The BigData Top100 List would be maintained by this steering committee on behalf of the community.

Next Steps

Development of a benchmark standard can be a complex and time-consuming process. However, to speed up this process we are planning a series of events and contests that would be open to the community, including academia and industry—vendors as well as customers/users.

- **Contest 1.** Submission of representative data and operations for each step in the Big Data Analytics Pipeline described in the previous section Big Data Benchmark Proposals.
  - Submission deadline: March 31, 2013.
  - Review and selection of best data/operations for the pipeline: by the week of April 1, 2013.
- **Contest 2.** Reference implementation of selected data/operations for pipeline steps.
  - Submission deadline: May 30, 2013.
  - Review of reference implementation: by week of June 1, 2013.
- **Contest 3.** Proposals for metrics, execution rules, audit rules, and reporting rules.
  - Submission deadline: by the Third Workshop on Big Data Benchmarking, July 16–17, 2013, Xi'an, China.
  - Review of input and official release of benchmark specification: August 31, 2013.

Submissions will be accepted via the bigdatatop100.org website. Submissions will be posted at the site, with the permission of the authors. Winning submissions will receive wide exposure and will potentially be incorporated as is, or with modifications, into the formal benchmark specification. Winners will also receive a modest cash award.

Author Disclosure Statement

No competing financial interests exist.

References

1. Baru C., Bhandarkar M., Poess M., et al. Setting the direction for big data benchmark standards. TPC-Technical Conference, VLDB 2012, Istanbul, Turkey.
2. Islam N., Lu X., Rahman M., et al. A micro-benchmark suite for evaluating HDFS operations on modern clusters. Second Workshop on Big Data Benchmarking 2012, Pune, India.
3. Fowler M. Polyglot persistence. Nov. 16, 2011. Available online at http://martinfowler.com/bliki/PolyglotPersistence.html (Last accessed on January 29, 2003).
4. TPC-DS: TPC Benchmark™ DS (TPC-DS): The new decision support benchmark standard. Available online at www.tpc.org/tpcds/default.asp (Last accessed on January 29, 2003).
5. Ghazal A. Big data benchmarking—data model proposal. 2012. First Workshop on Big Data Benchmarking, San Jose, California. Available online at http://clds.sdsc.edu/wbdb2012/program (Last accessed on January 29, 2003).
6. Ghazal A. BigBench. 2012. Second Workshop on Big Data Benchmarking, Pune, India. Available online at www.paralleldatageneration.org/download/wbdb/WBDB2012_IN_05_Ghazal_BigBench.pdf (Last accessed on January 29, 2003).

Address correspondence to:

Chaitanya Baru
San Diego Supercomputer Center
University of California, San Diego
9500 Gilman Drive
La Jolla, CA 92093-0505

E-mail: baru@sdsc.edu