

Lessons learned using the TPCx-BB Express Benchmark* for end-to-end big data clusters

TPCx-BB is a framework that measures the variety, velocity, and volume aspects of end-to-end big data analytics. We propose extending the benchmark to include metrics that evaluate the performance of data ingestion and real-time processing.

ABSTRACT

The TPCx-BB Express Benchmark (TPCx-BB)*¹ is designed to measure the performance of big data analytics systems. The benchmark contains 30 use cases (queries) that simulate big data processing, big data storage, big data analytics, and reporting. Our team used this benchmark to evaluate the performance of software and hardware components for big data clusters. We find the benchmark has good coverage for different data types. The benchmark also provides enough scalability to address challenges of scaling data size and nodes. We have gained key insights into designing big data analytic systems by using TPCx-BB.

We do need more than TPCx-BB to evaluate and design complete, end-to-end big data systems. That's because there is a difference between an analytics system and a real-world, end-to-end system. For example, the data flow of an end-to-end system should include data ingestion.

Data ingestion moves data from where it originates in a system (such as Apache Hadoop*) to where it can be stored and analyzed. Importing that data at a reasonable speed can be challenging for businesses that want to maintain a competitive advantage. However, TPCx-BB was not designed to evaluate the performance of software and hardware for data ingestion. Consider the three dimensions of big data: volume, variety, and velocity. Velocity refers to the high speed of data processing: real time or near real time. Unfortunately, with TPCx-BB, there is a strict limitation on bandwidth and latency for real-time processing.

This paper discusses our experiences and lessons learned using TPCx-BB to evaluate the performance of software and hardware for real-time processing. We then offer advice on how to extend TPCx-BB to evaluate data ingestion and real-time processing. Finally, we share some ideas on how to implement fuller TPCx-BB coverage for end-to-end big data clusters.

Authors

Kebing Wang
Intel Corporation
kebing.wang@intel.com

Bianny Bian
Intel Corporation
bianny.bian@intel.com

Paul Cao
Hewlett Packard Enterprise
paul.cao@hpe.com

Mike Riess
Intel Corporation
mike.riess@intel.com

Table of contents

Abstract 1

Introduction2

Data ingestion2

Big data storage2

Big data analytics2

Benchmarks2

Use cases: Video stream processing and health monitoring.....3

Experience and lessons learned using TPCx-BB3

Challenges in using TPCx-BB3

 Challenge 1: Messaging, stream processing, and data ingestion 4

 Challenge 2: TPCx-BB does not cover real-time analytics5

 Challenge 3: Analytics benchmarks do not evaluate an entire system.....5

Suggestions for extending TPCx-BB5

Extend TPCx-BB to include data ingestion5

Extend TPCx-BB to involve real-time analytics6

Summary7

Introduction

Big data refers to data that is too diverse, fast-changing, or massive for traditional technologies to process efficiently. To solve today’s challenges of the volume, variety, and velocity of big data, new technologies and architectures are continually being invented. This creates a need for better benchmarks — and a need for new ways to use existing benchmarks to analyze the performance of complete big data systems.

In a big data system, data can come from dynamic, disparate, and distributed sources that have different formats, schemas, protocols, speeds, and sizes. Typical data sources include machines, geo-location devices, click streams, files, social feeds, log files, and videos. Figure 1 shows the key components of a typical big data system: data ingestion, data storage, and data analytics.

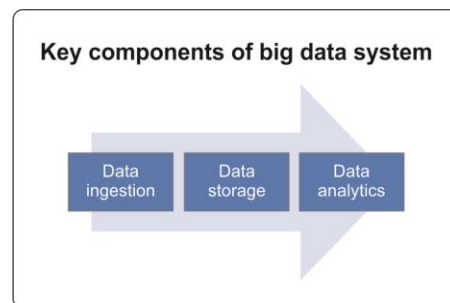


Figure 1. Components of a big data cluster

Data ingestion

This is the process of collecting, filtering, transforming, and reliably moving data to a system where the data can be stored and processed. Data ingestion may be continuous or asynchronous. It may also be real-time or batched — or both, depending on the characteristics of the data source and its destination.

For businesses, importing big data at a reasonable speed can be challenging. Common software stacks currently used in data ingestion include Apache Kafka*, a popular distributed messaging queue that is widely used as a critical software component.

Big data storage

Storage for a big data cluster must be able to handle large amounts of structured/unstructured data. This storage must also be easily scaled to keep up with increasingly large data sets. Big data storage — such as Apache Hadoop Distributed File System* (HDFS) and MongoDB* — must provide the bandwidth necessary to deliver full sets of data to analytic tools.

Big data analytics

Analytics examine large amounts of data to uncover hidden patterns, correlations, and other insights. Typical scenarios for big data analytics are real-time and batch analytics, as well as interactive accesses. Many libraries and tools built on these engines are used for interactive analytics, machine learning, and graph processing.

Benchmarks

Most developers are aware that big data technologies have grown significantly over the past few years, and continue to be widely adopted. What developers need now is a standard benchmark to evaluate and compare the performance of these big data systems. Such a benchmark should cover the entire data flow, as well as address the most common use cases for big data technologies.

One option is the TPCx-BB Express Benchmark (TPCx-BB)*1. TPCx-BB is based on TPC BigBench*2, a framework for end-to-end big data analytics. TPCx-BB measures performance and addresses the variety, velocity, and volume aspects of big data systems that contain structured, semi-structured, and/or unstructured data. The benchmark includes 30 queries to cover different categories of analytics from a business prospective. What TPCx-BB cannot do is evaluate the performance of data ingestion and real-time processing in an end-to-end big data system.

There are several benchmarks related to big data workloads. One is TPC-DI³, a standard data ingestion benchmark. TCP-DI focuses mainly on batch and structured data ingestion, but does not test the performance of data processed in real-time mode or unstructured data ingestion. Other related benchmarks and their limitations for evaluating end-to-end systems are mentioned at the end of this article.

In our experience, it is better to use a single benchmark to evaluate data flow and the entire framework of a big data system. Based on our research into TPCx-BB, we advise extending the TPCx-BB benchmark to include metrics for data ingestion and real-time processing of big data.

Use cases: Video stream processing and health monitoring

For several years, our team has helped developers deploy big data clusters. These clusters are typically designed for two real-world use cases: Continuous video stream processing, and health monitoring. Both use cases are end-to-end solutions.

In the use case for continuous video stream processing (see Figure 2), raw videos are continuously sent to the data

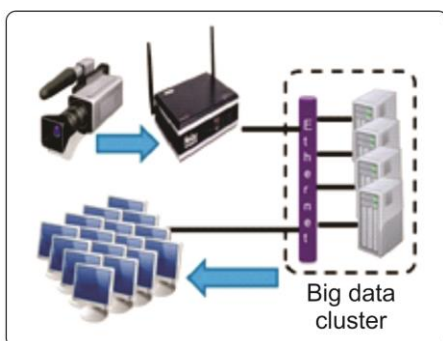


Figure 2. Use case: Video stream processing system

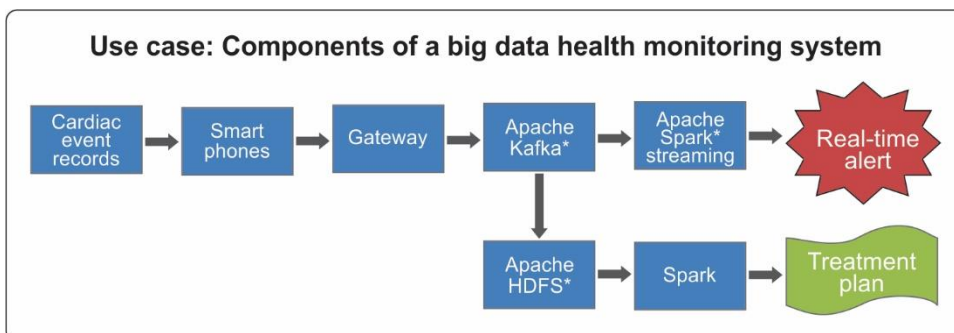


Figure 3. Use case: Health monitoring system

center through a gateway. The big data cluster accepts the stream, encodes the videos, and analyzes the video in real time. The cluster also records some responses and provides some interactive access for history data.

In the use case for health monitoring, we use software stacks for the framework (see Figure 3). Cardiac event records (CERs) collect the cardiac status of patients, and upload cardiac events to the data center. Uploading is done via smart phones and gateways. A Kafka-based cluster then receives the event records and transfers them to an Apache Spark* streaming cluster for real-time analytics.

When a health risk is identified (via those real-time analytics), an alert is immediately generated to inform both patient and physician. These events are also stored in the HDFS, the physician can perform specific batch analytics on that data to help formulate a treatment plan.

In both use cases, developers want to use standard benchmarks to gain insights into the workloads of different software stacks, as well as for cluster deployment, planning, and optimization. In our experience, while TPCx-BB has been useful in characterizing the performance of analytics systems, it also presents several challenges.

Experience and lessons learned using TPCx-BB

To help characterize and deploy big data clusters, we used TPCx-BB to evaluate the performance of big data cluster software and hardware. We chose TPCx-BB because it has good coverage on different data types. It also provides enough scalability to address challenges in data size and node scaling. Using TPCx-BB has helped us gain key insights into designing analytic systems. However, the challenges in using TPCx-BB in practice are especially noticeable when designing an end-to-end system.

Challenges in using TPCx-BB

We noted three key issues that prevented us from being able to use TPCx-BB as a full-coverage benchmark for end-to-end big data clusters.

- TPCx-BB does not test or measure messaging, stream processing, or data ingestion.
- TPCx-BB does not perform real-time analytics.
- Analytics benchmarks (such as TPCx-BB) do not evaluate an entire system.

Table 1. Cluster settings

Component	Description
Node count	1 master + 8 slaves (Hewlett Packard DL380* Gen9)
Processor	Intel® Xeon® processor E5-2699 v3, 2.30GHz
DRAM	256G DDR4-2133, 8 channels
Disk	Intel® Solid State Drive DC S3700 2.0 TB
Network	Intel® Ethernet Network Adapter XXV710 for 25GbE
Software framework	Apache Hadoop* version CDH5.5

Challenge 1:
Messaging, stream processing, and data ingestion

An end-to-end data pipeline includes analytics, but also messaging, streaming data, and data ingestion. TPCx-BB exercises only analytics. The benchmark does not characterize messaging, stream processing, or data ingestion. The workload characteristics of those software stacks are different than the characteristics for which the benchmark was designed (analytics).

To understand how this affects trying to characterize an end-to-end data pipeline, we profiled the benchmark. We tested it for results in scaling up processor frequency, core counts, network bandwidth, and disk bandwidth. Table 1 describes the cluster components we used for this research.

The next four graphs show the results of using TPCx-BB to scale various cluster components. Figures 4 and 5 show the TPCx-BB results for scaling processor frequency and core count for all 30 queries of TPCx-BB. You can see in Figure 4 that nearly all TPCx-BB queries are sensitive to changes in CPU frequency. According to the TPCx-BB benchmark, the scaling efficiency for the power test is 93% from 1.2GHz to 1.8GHz, and 88% from 1.2GHz to 2.3GHz.

Figure 5 shows that many TPCx-BB queries are also sensitive to the processor's core count. Results from TPCx-BB appear to show that the scaling efficiency for this power test is 80% when scaling from 9 cores/18 threads to 18 cores/36 threads. Efficiency is only 50% when scaling from 9 cores/18 threads to 36 cores/72 threads.

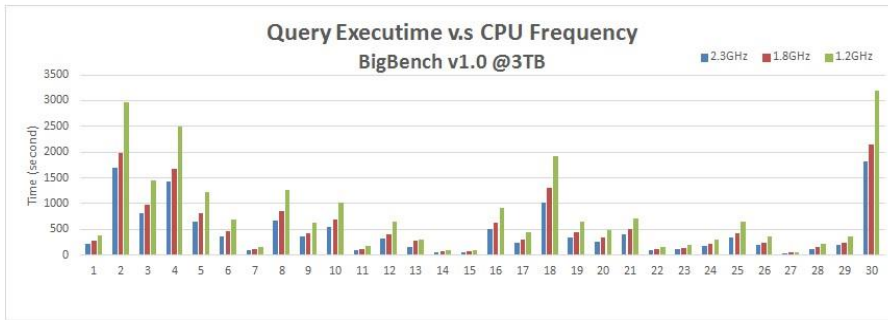


Figure 4. TPCx-BB results when scaling CPU frequency

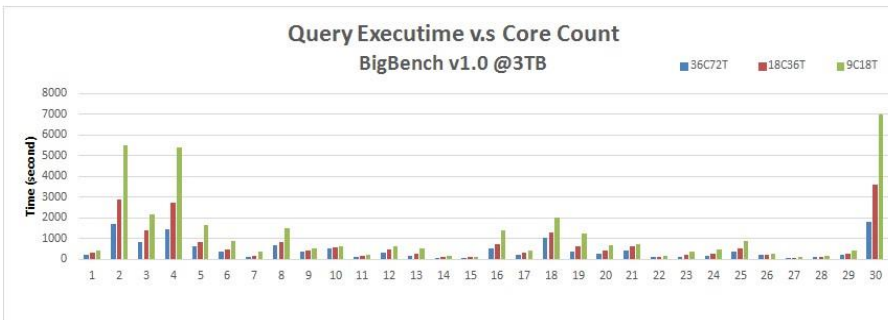


Figure 5. TPCx-BB results when scaling to more processor cores

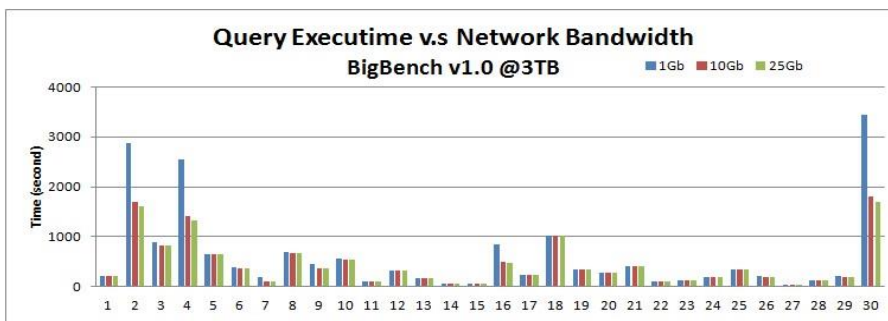


Figure 6. TPCx-BB results when scaling the network

Figures 6 (previous page) and 7 show the TPCx-BB results for scaling disk and network bandwidth. These results are significantly different from the results of scaling processor frequency and core count. As you can see in figures 6 and 7, only a few long execution queries are sensitive to scaling disk and network bandwidth, including queries 2, 3, and 4.

Overall, TPCx-BB reports that the performance of the power test is increased by only 1% when increasing network bandwidth from 10GbE to 25GbE; or when increasing disk bandwidth from 400MB to 2500MB.

In terms of big data ingestion, both network and disk bandwidth are important. Because of this, we used Kafka to identify these workload characteristics. As shown in Figure 8 (next page), updating from a 1GbE network to a 10GbE network increases Kafka throughput by 4.2x. Updating to a 25GbE network increases throughput by 6.9x.

Note that, for this test, the cluster setting is the same as in Table 1 (previous page), except that the number of nodes is 3 (not 9). The size of messages sent to Kafka is 230KB, and there are total of 3 customers for this test's topic of messages.

Based on previous profiling data, TPCx-BB is processor intensive, but makes only a modest impact on network and disk I/O. Unfortunately for end-to-end big data benchmarking, data ingestion is generally both network- and disk I/O- intensive.

Due to the difference of workload characteristics between data ingestion and analytics, TPCx-BB cannot be used as-is to measure a full, end-to-end data pipeline for big data systems.

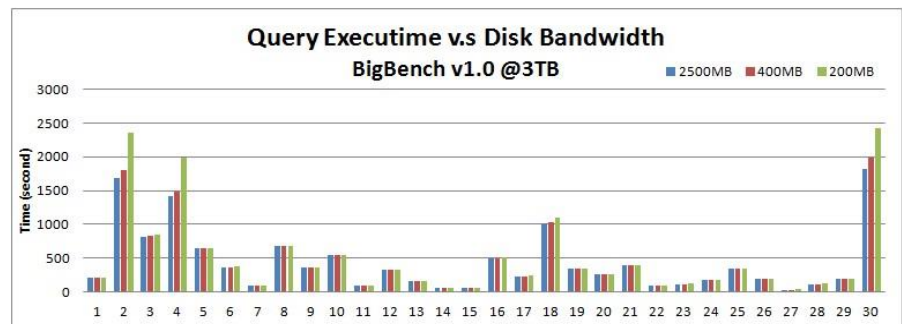


Figure 7. TPCx-BB results when scaling for disk bandwidth

Challenge 2: TPCx-BB does not cover real-time analytics

The second issue in using TPCx-BB has to do with real-time analytics. TPCx-BB is designed to measure batch analytics, not real-time analytics. In real-time processing, latency is a critical metric. In the use case of a health monitoring system, the latency of real-time alerts can have a direct and significant impact on a patient's life. Any delay can put a life at risk. Unfortunately, TPCx-BB doesn't include a way to measure latency.

Challenge 3: Analytics benchmarks do not evaluate an entire system

The third key challenge in using TPCx-BB is that different components of a big data system usually run on the same physical cluster. For example, data ingestion, real-time analytics, and batch analytics often concurrently execute on the same cluster in order to share the resources.

In the use case of the health monitoring system, Spark-based streaming for real-time analytics and Spark for batch analytics co-execute on the same cluster. In this use case, you cannot use an analytics benchmark (such as TPCx-BB) to evaluate the whole system, because different components have different workload characteristics and require different metrics for measurement.

Suggestions for extending TPCx-BB

Based on the lessons we've learned from using TPCx-BB in practical, real-world settings, we advise extending TPCx-BB to provide additional information about the entire data flow of end-to-end big data frameworks. We also offer some ideas on how to implement our recommended extensions.

Extend TPCx-BB to include data ingestion

Data ingestion may be performed in real-time or by batch, depending on the data source and destination warehouse. Load testing with TPCx-BB can be thought of as a simple batch mode of data ingestion. (Remember that the current TPCx-BB benchmark measures only batch data ingestion.)

The underlying benchmark for TPCx-BB — BigBench — is based on a fictitious retailer who sells products to customers via both physical (brick and mortar) retail stores and online retail stores. In the real world, it is likely that data from physical stores is ingested into a big data system via batch mode, while data from online retailers is ingested via real-time mode. Because of this, we suggest dividing the load testing with TPCx-BB into two parts. Part one loads data from the physical stores, as before. The second part of the test is changed to load data from online retail stores via real-time mode.

To do this, we must change the process a bit. As shown in Figure 9, we still use the typical parallel data-generation framework (PDGF)⁴ data generator to generate raw data from both physical and online stores. Also, as per the current TPCx-BB, raw data from the physical stores is still loaded into data storage with optimized storage formats. For example, optimized storage formats include optimized row columnar* (ORC) format and Apache Parquet*.

Now for the changes. In our new model, the raw data from online stores is no longer directly imported. Instead, raw data from online retailers is emitted by a new component: a producer (see Figure 9). This component wraps up and sends the raw data to the data warehouse. The producer can control the input rate for streaming data for ingestion by the system under test. By making the streaming input rate configurable, we can now simulate different real-time data streams. Also, in this new model, the data ingestion component of the system under test can now receive messages, as well as extract, transform, and/or load messages into data storage.

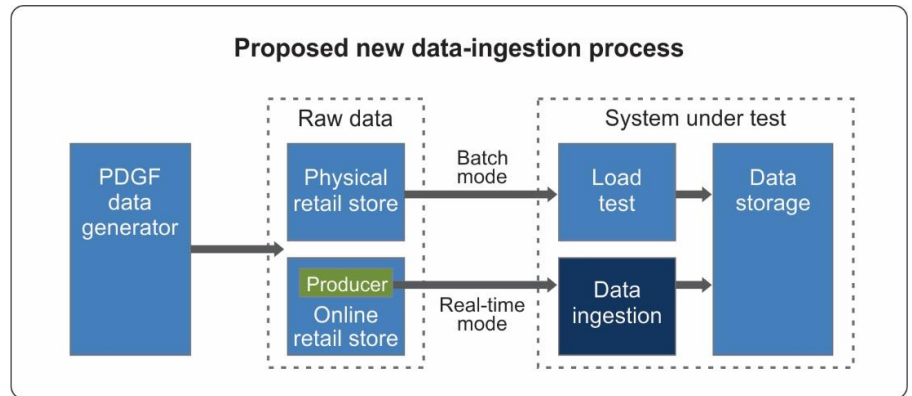


Figure 9. Suggested new data ingestion process includes the new component: producer

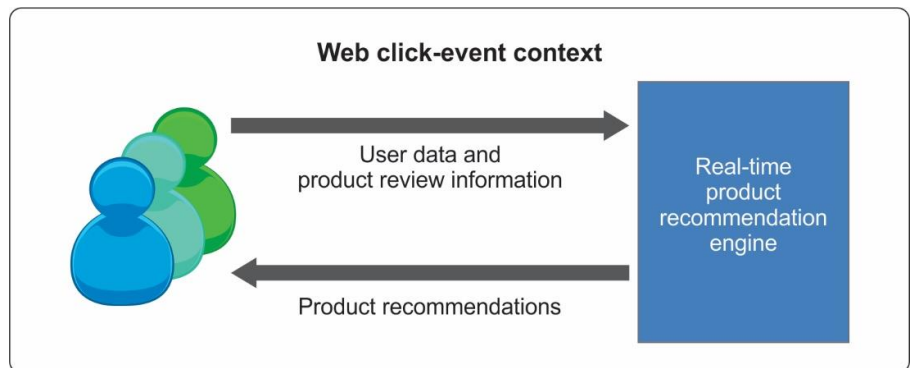


Figure 10. Real-time analytics

Extend TPCx-BB to involve real-time analytics

Our second key suggestion is to extend TPCx-BB to include real-time analytics.

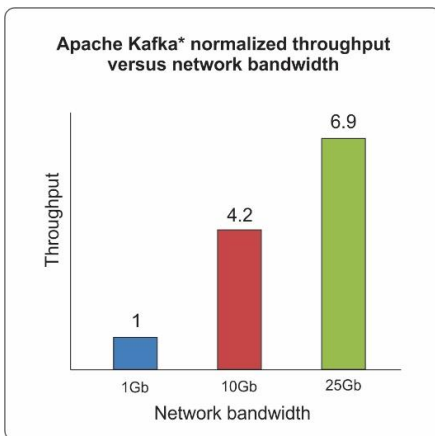


Figure 8. Apache Kafka* network scaling

Currently, all 30 queries of TPCx-BB are offline batch analytics. They do not include real-time data analytics, which is a common use case of big data systems.

One of the challenges here is that metrics for real-time data ingestion should involve both throughput and latency. Throughput measures how many bytes are ingested in a unit of time. Latency is the time it takes for a message emitted by the producer to be stored in the data warehouse.

We can adapt TPCx-BB to cover real-time analytics by adding a real-time product recommendation engine into the benchmark suites (see Figure 10). This can be done since the benchmark already contains a Web click-stream in its dataset. A Web click-stream includes customer profiles and reviews of Web pages.

A real-time product recommendation engine can use this customer and review information to recommend products to customers. Such a recommendation algorithm is an increasingly hot-topic algorithm in machine learning technologies.

For our big data use cases, if we introduce a product recommendation engine into the TPCx-BB benchmark suites, the engine can represent not only real-time data analytics, but also the inference phase of machine learning. Basically, part of the inference phase of the machine learning algorithm is to recommend products based on a customer's profile and product features.

The metric for evaluating real-time analytics should also consider both throughput and latency. Throughput is

Related benchmark work

There are several related benchmarks that provide metrics for real-time analytics. As with TPC-DI³, each benchmark has its strengths and limitations. For our project, these additional benchmarks did not provide the detailed analysis we wanted. This is why we continue to use TPCx-BB, and recommend extending the benchmark to include metrics for the end-to-end data pipeline.

Here we explain some of the advantages and limitations of various related benchmarks. These are benchmarks that could not meet our needs for characterizing the performance of end-to-end big data systems.

TPC-DI. We considered using TPC-DI as an additional benchmark in this study, but found that it could not help evaluate performance for real-time data ingestion. TPC-DI is a data integration benchmark developed by TPC. TPC-DI combines and transforms data extracted from a fictitious brokerage firm's online transaction processing (OTLP) system, along with data from other sources. That data is then loaded into a data

warehouse. However, TPC-DI performs data ingestion only in batch mode. The benchmark's metric includes only throughput, and not latency. Because of this, we could not use TPC-DI to evaluate the performance of real-time data ingestion.

Alexey Medvedev and Alireza Hassani⁵ proposed some benchmarking metrics for a series of experiments. The experiments were designed to evaluate and test the performance of data ingestion and storage of the widely used open source platform, OpenIoT. Medvedev and Hassani provide a detailed analysis of the experimental outcomes. Again, however, the benchmark they propose focuses only on data ingestion and storage performance of IoT platforms. It does not provide full test and metrics for end-to-end systems.

Yahoo Streaming Benchmarks^{*6} is a simple advertisement application. The Yahoo benchmark includes a number of advertising campaigns, and a number of advertisements for each campaign. The benchmark reads various Apache JavaScript* object notation (JSON)

events from Kafka. The benchmark then identifies the relevant events, and stores a windowed count of relevant events per campaign into a Redis Labs* database management system. These steps attempt to probe common operations on data streams. Yahoo Streaming Benchmarks is not an end-to-end benchmark for big data frameworks. It is a benchmark only for evaluating real-time processing.

The Numenta Anomaly Benchmark (NAB)⁷ is a benchmark for detecting streaming anomalies. NAB has two main components: a dataset with labeled, real-world time-series data; and a scoring system designed for streaming data. The NAB repository currently includes ten anomaly detection algorithms. It is a standard open source framework, but it evaluates only real-time anomaly detection algorithms.

With these evaluations, we continue to recommend TPCx-BB as the preferred, most adaptable benchmark to extend in order to characterize end-to-end big data systems.

the number of Web click-events handled per second. Latency is the response time for generating product recommendation information for each click event. For real-time processing, both metrics are important.

Real-time analytics are usually tightly coupled with real-time data ingestion. Because of this, they can be merged with the same data stream emitted by the new producer component. When Web click-events are then imported into the data warehouse, the system can generate product recommendation information at the same time.

Summary

In this paper, we provide some initial ideas about how to extend TPCx-BB to measure real-time data ingestion and analytics. The next step is to implement a proof of concept to evaluate the effectiveness of our ideas. Our experience has shown that although there are several big data benchmarks, TPCx-BB shows the most promise for being adaptable to measure the performance of a full, end-to-end big-data system.

We encourage developers to examine our research on the Intel developer site. We welcome your ideas and comments to help develop this benchmark to provide detailed performance information that includes data ingestion and other components of big data clusters.

For more information about the TPCx-BB benchmark, visit www.tpc.org/tpcx-bb/default.asp/



- 1 TPCx-BB source: <http://www.tpc.org/tpcx-bb/default.asp/>
- 2 BigBench: Towards an Industry Standard Benchmark for Big Data Analytics; Ahmad Ghazal, Tilmann Rabl, Mingqing Hu, Francois Raab, Meikel Poess, Alain Crolotte, Hans-Arno Jacobsen; SIGMOD13, June 2227, 2013, New York, New York, USA.
- 3 TPC-DI source: <http://www.tpc.org/tpcdi/>
- 4 A Data Generator for CloudScale Benchmarking; T. Rabl, M. Frank, H. M. Sergieh, and H. Kosch; TPCTC pages 4156, 2010.
- 5 Data Ingestion and Storage Performance of IoT Platforms: Study of OpenIoT; Alexey Medvedev, Alireza Hassani, Arkady Zaslavsky, Prem Prakash Jayaraman, Maria Indrawan-Santiago, Pari Delir Haghighi, Sea Ling; LNCS, volume 10218.
- 6 Benchmarking Streaming Computation Engines: Storm, Flink and Spark Streaming, Parallel and Distributed Processing Symposium Workshops; Sanket Chintapalli, Derek Dagit, Bobby Evans, Reza Farivar, Thomas Graves, Mark Holderbaugh, Zhuo Liu, Kyle Nusbaum, Kishorkumar Patil, Boyang Jerry Peng and Paul Poulosky; 2016 IEEE International.
- 7 Evaluating Real-Time Anomaly Detection Algorithms – The Numenta Anomaly Benchmark, Machine Learning and Applications (ICMLA); Alexander Lavin, Subutai Ahmad; 2015 IEEE 14th International Conference.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

Benchmark results were obtained prior to implementation of recent software patches and firmware updates intended to address issues referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Information in this document is provided in connection with Intel® products. No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Results have been estimated based on internal Intel analysis and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

Hewlett Packard and the Hewlett Packard logo are trademarks of Hewlett Packard Enterprise in the U.S. and/or other countries.

Copyright © 2017 Intel Corporation. All rights reserved.

Copyright © 2017 Hewlett Packard Enterprise. All rights reserved.

*Other names and brands may be claimed as the property of others.