BIGDATABENCH-MT: A BENCHMARK TOOL FOR 
GENERATING REALISTIC MIXED DATA CENTER 
WORKLOADS

Rui Han, Shulin Zhan, Chenrong Shao, Junwei Wang, 
Jiangtao Xu, Lizy K. John, Lei Wang, Jianfeng Zhan

Institute Of Computing Technology 
Chinese Academy of Sciences 
Beijing, China
english.ict.cas.cn/

Beijing, China

Technical Report No. ICT-April. 8, 2015
Abstract

Long-running service workloads (e.g. web search engine) and short-term data analytic workloads (e.g. Hadoop MapReduce jobs) often co-locate in today’s data centers. Developing realistic benchmarks reflecting such practical scenarios is a key problem to produce trustworthy results when evaluating and comparing data center systems. This requires both actual workloads and their arrivals to follow patterns hidden in real-world traces. However, existing benchmarks either generate actual workloads based on probability models, or replay workload traces using synthetic workloads. To fill this gap, we propose a benchmark tool that is a first step towards generating a mix of realistic service and data analytic workloads on the basis of real workload traces. Our tool includes a combiner that enables the replaying of actual workloads according to the workload traces, and a multi-tenant generator that assists benchmark users to flexibly scale the workloads according to their requirements. Based on this, our demo illustrates the workload customization and generation process using the visual interface of our tool. The proposed tool, called BigDataBench-MT, is a multi-tenant version of our comprehensive benchmark suite BigDataBench and it is publicly available from [http://prof.ict.ac.cn/BigDataBench/multi-tenancyversion/](http://prof.ict.ac.cn/BigDataBench/multi-tenancyversion/)
1 Introduction

In modern cloud data centers, a large number of tenants are consolidated to share a common computing infrastructure and execute a diverse mix of workloads. Benchmarking and understanding these workloads is a key problem for system designers, programmers and researchers to optimize the performance and energy efficiency of data center systems and to promote the development of data center technology. This work focuses on two classes of popular data center workloads [8]:

- **long-running services.** These workloads offer online services such as web search engines and e-commerce sites to end users and the services usually keep running for months and years. The tenants of such workloads are service end users.

- **short-term data analytic jobs.** These workloads process input data of many scales using relatively short periods (e.g. in Google and Facebook data centers, a majority (over 90%) of analytic jobs complete within a few minutes [9, 8]). As data analytic systems such as Hadoop mature, both types of workloads widely co-locate in today’s data centers, hence the pressure to benchmark and understand these mixed workloads rises. The tenants of such workloads are job submitters.

Within this context, we believe that it will be of interest to the data management community and a large user base to generate realistic workloads such that trustworthy benchmarking reflecting the practical data center scenarios can be conducted. Considering the heterogeneity and dynamic nature of data center workloads and their aggregated resource demand and arrival patterns, this requires overcoming two major challenges.

**Benchmarking using actual workloads based on real workload traces.** Data analytic jobs usually have various computation semantics (i.e. implementation logics or source codes) and input data sizes (e.g. ranging from KB to PB), and their behaviors also heavily rely on the underlying software stacks (such as Hadoop or MPI). Hence it is difficult to emulate the behaviors of such highly diverse workloads just using synthetic workloads. Moreover, generating workloads whose arrival patterns follow real-world traces is an equally important aspect of realistic workloads. This is because these traces are the most realistic data sources including both explicit and implicit arrival patterns (e.g. sequences of time stamped requests or jobs).

**Benchmarking using scalable workloads with realistic mixes.** A good benchmark needs to flexibly adjust the scale of workloads to meet the requirements of different benchmarking scenarios. Based on our experience, we noticed that in many cases, obtaining real workload traces is difficult due to confidential issues. The limited trace data also restrict the scalability of benchmark. It is challenging to produce workloads at different scales while still guaranteeing their realistic mix corresponding to real-world scenarios.

In addition to database benchmarks such as the TPC-* series of benchmarks, existing benchmarks designed for batch or analytical systems (e.g. Hadoop, Spark...
and Hive) can be divided into two complementary categories [10]. The first category of benchmarks [1, 11, 12] supports actual workloads but the arrival patterns of these workloads are determined by probability models. By contrast, in the second category of benchmarks, the workload arrival patterns faithfully follow real workload traces, but only synthetic workloads are supported. Specifically, GridMix [4] provides two synthetic workloads: LoadJob that performs I/O operations and SleepJob that sleeps the jobs. SWIM [9] provides four synthetic workloads that emulate the operations of Hadoop jobs to read, write, shuffle and sort data. However, how to generate actual workloads on the basis of real workload traces, is still an open question.

In this report, we propose a benchmark tool that is a first step towards generating realistic mixed data center workloads. This tool, called BigDataBench-MT, is a multi-tenancy version of our open-source project BigDataBench, which is a comprehensive benchmark suite including 14 real-world data sets and 33 actual workloads covering five application domains [3, 13]. The goal of BigDataBench-MT is not only supporting the generation of service and data analytic workloads based on real workload traces, but also providing a multi-tenant framework to enable the scaling up and down of such workloads with guarantee of their realistic mixes. Considering our community may feel interest in using these workloads to evaluate new system designs and implementations, our tool and the corresponding workload traces are publicly available from [http://prof.ict.ac.cn/BigDataBench/multi-tenancyversion/](http://prof.ict.ac.cn/BigDataBench/multi-tenancyversion/).

2 System Overview

Figure 1 shows the framework of our benchmark tool. It consists of three main modules. In the Benchmark User Portal, users can first specify their benchmarking requirements, including the machine type and number to be tested, and the types of workload to use. A set of workload traces following these requirements are then selected. The next step of Combiner of Workloads and Traces is to match the real workload and the selected workload traces, and outputs workload replaying scripts to guide the workload generation. Finally, the Multi-tenant Workload Generator extracts the tenant information from the scripts and constructs a multi-tenant framework to generate a mix of service and data analytic workloads.

In BigDataBench-MT, we employ the Sogou user query logs [7] as the basis to generate the service workload (i.e. the Nutch search engine [2]) and the Google cluster workload trace as the basis to generate data analytic workloads (i.e. Hadoop and Shark workloads). The Sogou trace records logs from 50 days and it includes over 9 million users and 43 million queries. The Google trace records logs from 29 days and 12,492 machines and it includes over 5K users, 40K workload types, 1000K jobs and 144 million tasks. As a preprocessing step, we converted both traces into Impala databases (full version) and MySQL database (24-hour version) to facilitate the customization of benchmarking scenarios. In the following subsections, we describe the last two modules of our tool.
2.1 Combiner of Workload and Traces

The goal of the combiner is to extract the request/job arrival patterns from real workload traces and combine them with actual workloads. The combiner applies differentiated combination techniques to the service and data analytic workloads because their workload generations have different features.

**Service workloads.** The generation of a service workload is determined by three factors: the request submitting time, the sequence of requests and the content of each request query. Take the web search engine for example, the combiner implements a request submitting service that automatically derives these factors from the Sogou trace and uses them to determine the request submission process.

**Data analytic workloads.** The generation of a data analytic workload is determined by four factors: the job submitting time, the workload type (i.e. the computation semantics and the software stack) and the input data (i.e. the data source and size). The current workload traces show the information of job submitting time but only provide anonymous jobs whose workload types and/or input data are unknown. Hence the basic idea of the combiner is to derive the workload characteristics of both actual and anonymous jobs and then match jobs whose workload characteristics are sufficiently similar. Table 1 lists the metrics used to represent workload characteristics, which reflect both jobs’ performance (execution time and resource usage) and micro-architectural behaviors (CPI and MAI).

Figure 2 shows the process of matching actual and anonymous data analytic jobs and it consists of two parallel sub-processes. First, the actual jobs with different input data sizes are tested and their metrics of workload characteristics are collected. In
Table 1: Metrics to represent workload characteristics of data analytic jobs

| Metric       | Description                          |
|--------------|--------------------------------------|
| Execution time | Measured in seconds                  |
| CPU usage    | Total CPU time per second            |
| Total memory size | Measured in GB                     |
| CPI          | Cycles per instruction               |
| MAI          | The number of memory accesses per instruction |

BigDataBench-MT, we provide auto-running scripts to collect performance metrics and hardware performance counters (Perf [6] and Oprofile [5] for Linux 2.6+ based systems) to obtain micro-architectural metrics. Using the testing results as samples, the combiner trains the multivariate regression model to describe the relationship between an actual job (including both its workload type and input size as the independent variables) and its workload characteristic metrics (one metric is a dependent variable). Second, the combiner views each anonymous job as an entity and the five workload characteristic metrics as its attributes, and employs the Bayesian Information Criterion (BIC)-based k-means clustering algorithm [14] to cluster anonymous jobs in the trace into different clusters.

Based on the constructed regression models and clusters, the combiner further matches each cluster to one actual job with a specific input data. In the matching, the coefficient of variation (CV) measure, defined as the ratio of the standard deviation $\sigma$ to the mean $\mu$, is used to describe the dispersion of jobs in the same cluster. The combiner iteratively tests actual jobs of different workload types and input sizes, and matches an actual job with a cluster under two conditions: (i) the CV of the cluster is smaller than a specified threshold (e.g. 0.5), which indicates the anonymous jobs in this cluster are closely similar to each other; (ii) the change in this CV is smaller than a threshold (e.g. 0.1) after the actual job is added to the cluster. This means the workload characteristics of the added job are sufficiently similar to those of the anonymous jobs in the cluster. If multiple matched actual jobs are found for one cluster, the combiner selects the job resulting the smallest CV change. Finally, the combiner produces workload replaying scripts as the output.

Note that BigDataBench-MT provides two ways of using the above combiner. First, it directly provides some workload replaying scripts, which are the combination results of representative actual workloads (e.g. Hadoop Sort and WordCount) and the Google workload trace. Second, it also supports benchmark users to directly use the above combination technique to match their own data analytic jobs with Google anonymous jobs.
2.2 Multi-tenant Workload Generator

Based on workload replaying scripts, the workload generator applies a multi-tenant mechanism to generate a mix of workloads using two steps. First, the generator extracts the tenant information from the scripts. For the service and data analytic workloads, this tenant information represents the number of concurrent end users and submitters of analytic jobs, respectively. Second, the generator creates a client for each tenant and emulates the scenarios that a number of end users/job submitters concurrently submit requests/jobs to the system. This multi-tenant framework allows the flexible adjustment of workload scales with guarantee of their realistic mixes. For example, benchmark users can double or halve the size of concurrent tenants, after which the distributions of requests/jobs submitted by these tenants still correspond to those in real workload traces.

3 Demonstration Description

3.1 Chosen Workloads and Workload Traces

In our demonstration, benchmark users want to evaluate their data center systems using a mix of service and data analytic workloads. The Nutch web search engine \[2\] is used as the example service workload and four Hadoop workloads are used as the example data analytic workloads. The chosen Hadoop workloads have a variety of workload characteristics: WordCount and Naïve Bayes classification are typical CPU-intensive workloads with integer and float point calculations; Sort is the typical I/O-intensive workload and PageIndex is the workload having similar demands for CPU and I/O resources. Both the data generators \[15\] and workloads in the demo can be obtained from BigDataBench \[3\].

Our demo uses a 24-hour user query logs from Sogou, which include 1,724,264 queries from 519,876 end users, as the basis to generate realistic search engine service; and uses a 24-hour cluster workload trace from Google, which includes 37,842
anonymous jobs from 2,261 job submitters, as the basis to generate realistic Hadoop jobs.

3.2 System Demonstration

BigDataBench-MT provides a visual interface in the Benchmark User Portal to help benchmark users make appropriate benchmarking decisions. This portal provides users necessary information, allows them input benchmarking requirements and executes system evaluations on their behalf. The whole process consists of three steps, as shown in Figures 3, 4 and 5, respectively.

Step 1. Specification of tested machines and workloads. The first step of the demo presents an overview of workload traces (i.e. Sogou and Google traces) and the data center status, including the six types of machines, their machine number and configurations, and the user, job and task statistics in these machines. This information assists benchmark users to select the type and number of machines to be evaluated, and the workloads they want to use. Suppose users select Type Four of the machines with 2 process cores and 4GB memory and 100 machines to be tested, the workload traces belonging to these machines are extracted and forwarded to the next step.

Step 2. Selection of benchmarking period and scale. At this step, users have the option to select the period and scale for their benchmarking scenarios. To facilitate this selection, BigDataBench-MT shows the statistic information of both the service workload (including its number of requests and end users per second) and the data analytic workloads (including their number of jobs and average CPU, memory resource usages) at each of the 24 hours. Suppose users select the benchmarking period of 12:00 to 13:00 and the scale factor is 1 (that is, no scaling is needed). The workload traces belonging to this period are selected for step 3.
Step 3. Generation of mixed workloads. After both workloads and traces have been selected, the final step employs the combiner described in Section 2.1 to generate workload replaying scripts for both the service and data analytic workloads, and sends these scripts as feedback to users. In the matching of actual Hadoop jobs with anonymous ones, we tested each Hadoop workload type using 20 different input sizes to build the regression models. Based on the replaying scripts, benchmark users can press the ”Generate mixed workload” button to trigger the multi-tenant workload generator, in which each tenant is an independent workload generator and multiple tenants generate a mix of realistic workloads.

4 Future Work

There are multiple avenues for extending the functionality of our benchmark tool. A first step will be to support more actual workloads. Given that there are 33 actual workloads in the BigDataBench and many workloads (e.g. Bayes classification and WordCount) have three versions of implementations (Hadoop, Spark and MPI), adding more workloads to BigDataBench-MT will be helpful to support wider benchmarking scenarios. We also plan to extend our multi-tenant workload generator to support different classes of tenants and allow users to apply different priority disci-
plines in workload generation.

References

[1] Amplab benchmark. https://amplab.cs.berkeley.edu/benchmark/
[2] Apache Nutch Search. http://nutch.apache.org/
[3] BigDataBench. http://prof.ict.ac.cn/BigDataBench/
[4] GridMix. http://hadoop.apache.org/docs/stable1/gridmix.html
[5] Oprofile. http://oprofile.sourceforge.net/
[6] Perf. https://perf.wiki.kernel.org/
[7] Sogou user query logs. http://www.sogou.com/labs/dl/q-e.html
[8] C. Reiss, A. Tumanov, et al. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In SoCC’12, pages 7-19. ACM, 2012.
[9] Y. Chen, S. Alsphaugh, and R. Katz. Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads. In Proceedings of the VLDB Endowment, vol. 5, no. 12, pages 1802-1813. ACM, 2012.
[10] R. Han, X. Lu, and J. Xu. On big data benchmarking. In BPOE’14, pages 3-18. Springer, 2014.
[11] A. Ghazal, M. Hu, et al. Bigbench: Towards an industry standard benchmark for big data analytics. In SIGMOD’13, pages 1197-1208. ACM, 2013.
[12] T. G. Armstrong, V. Ponnekanti, et al. Linkbench: a database benchmark based on the facebook social graph. In SIGMOD’13, pages 1185-1196. ACM, 2013.
[13] L. Wang, J. Zhan, et al. Bigdatabench: A big data benchmark suite from internet services. In HPCA’14. 2014.
[14] D. Pelleg, A. W. Moore, et al. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In ICML’00, pages 727-734. 2000.
[15] Z. Ming, C. Luo, et al. Bdgs: A scalable big data generator suite in big data benchmarking. In WBDB’13, pages 138–154. Springer, 2013.