Front. Comput. Sci., 2012, 6(4): 347–362
DOI 10.1007/s11704-012-2118-7

RESEARCH ARTICLE

CloudRank-D: benchmarking and ranking cloud computing systems for data processing applications

Chunjie LUO[1], Jianfeng ZHAN (✉)[1], Zhen JIA[1], Lei WANG[1], Gang LU[1], Lixin ZHANG[1], Cheng-Zhong XU[2,3], Ninghui SUN[1]

1 State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100019, China
2 Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, USA
3 Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

Abstract With the explosive growth of information, more and more organizations are deploying private cloud systems or renting public cloud systems to process big data. However, there is no existing benchmark suite for evaluating cloud performance on the whole system level. To the best of our knowledge, this paper proposes the first benchmark suite *CloudRank-D* to benchmark and rank cloud computing systems that are shared for running big data applications. We analyze the limitations of previous metrics, e.g., floating point operations, for evaluating a cloud computing system, and propose two simple metrics: *data processed per second* and *data processed per Joule* as two complementary metrics for evaluating cloud computing systems. We detail the design of CloudRank-D that considers representative applications, diversity of data characteristics, and dynamic behaviors of both applications and system software platforms. Through experiments, we demonstrate the advantages of our proposed metrics. In several case studies, we evaluate two small-scale deployments of cloud computing systems using CloudRank-D.

Keywords data center systems, clouds, big data applications, benchmarks, evaluation metrics

Received March 28, 2012; accepted June 15, 2012

E-mail: zhanjianfeng@ict.ac.cn

1 Introduction

We live in a world where information is growing explosively. To store and process massive data, more and more organizations around the world have started to rent public clouds or build internal datacenters. Internal datacenters are not open to the general public and tend to be shared among different departments or users for running big data applications. In this context, these internal datacenters are often referred to as private clouds [1] or warehouse-scale computers [2]. According to a recent Hadoop usage report [3], more than 150 companies have built their own Hadoop cluster systems whose machines range in scale from several nodes to thousands of nodes. Since MapReduce (the programming model of Hadoop) is not an one-size-fits-all solution [4], and many companies or institutes have developed their own software infrastructure, e.g., Dryad developed by Microsoft, it is likely that there are more private cloud systems than are in the Hadoop usage report [3]. Table 1 reports the deployments of private cloud systems in different companies[1)]. The data of renting public clouds for big data applications are not publicly available.

The concept of datacenter-as-a-computer [2] brings huge opportunities and challenges to the design of computer hardware and architecture. Different from traditional high

[1)] Google's data are cited from http://en.wikipedia.org/wiki/Google_platform. Facebook and Yahoo!'s data are taken from http://wiki.apache.org/hadoop/PoweredBy. Microsoft's data are cited from [5]. Taobao's data are taken from Hadoop in China 2010. Ebay, Baidu, Renren, and Tencent's data are taken from Hadoop in China 2011.

Table 1 Private cloud systems deployed in different companies

Company	Machine scale	Data scale	Software infrastructure
Google	450 000	20–100 PB	Google MapReduce, GFS, BigTable
Facebook	1 100	12 PB	Apache Hadoop Ecosystem
	300	3 PB	
Yahoo!	Total 40 000 (multi-cluster, biggest 4 500)	–	
Ebay	1 000 s	10 s PB	
Baidu	Total 16 000+ (10 clusters, biggest 3 000)	127.2 PB	
Renren	200	700 TB	
	30	–	
Taobao	1 100	9.3 PB	
Microsoft	1 800	–	Dryad
Tencent	–	–	Typhoon

performance computing (HPC) systems, cloud systems running data processing applications have three unique features.

First, data analysis applications account for a large proportion of the workload in private clouds. Facebook [6] reported that apart from ad hoc data analysis and business intelligence dashboards created by analysts across the company, many Facebook features, which range from simple reporting applications like insights for the Facebook advertisers, to more advanced features such as friend recommendation, are also based on analyzing large data sets. According to statistical analysis in [3], most companies deploy private cloud systems for data analysis applications, and the usage percentages of applications in private clouds are roughly shown in Fig. 1, from which we can observe that apart from web crawling and data storage, data analysis applications count more than 68%.

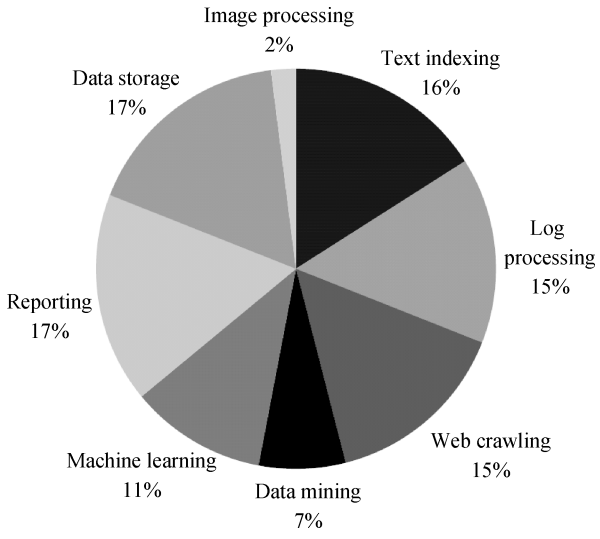


Fig. 1 Usage percentages of applications in private clouds reported in [3]

Second, in comparison with HPC systems that are mainly concerned with performance in terms of floating point operations (FLOPS), cloud systems running big data applications pay more attentions to data processing capabilities, which depends on not only the computing speed of processors but also the efficiency of accessing memory, disk, and network I/O. As a result, the evaluation metric of HPC systems, e.g., FLOPS [7], is not suitable for evaluating cloud systems running big data applications.

Third, cloud systems running big data applications have *unified system software platform*, e.g., Hadoop [8], to provide a distributed programming framework, distributed file systems, manage distributed resources, schedule jobs, and handle failures. So the unified system software platform and its dynamic behavior characteristics are an integral part of the workload. In other words, in characterizing workloads, we need to reflect the characteristics of not only users' jobs but also the unified system software platform.

Benchmarking is a quantitative foundation of computer system and architecture research. Benchmarks are used to experimentally determine the benefits of new designs [9]. The focus of this paper is to propose new metrics and a benchmark suite for evaluating a cloud system running big data applications *on the whole system level* in stead of a component or subsystem level, e.g., processor or I/O subsystems.

There are many benchmarks for evaluation of processors [9,10], HPC [7], web servers [11], database servers [12]. These traditional benchmarks are not necessarily suitable for evaluating cloud systems because of the differences between cloud systems and traditional computer systems. GridMix [13], HiBench [14], and Berkeley WL Suite [15] are benchmarks designed for evaluating the Hadoop framework. They do not aim to evaluate cloud systems on the whole system level. Proposed for architecture research, CloudSuite [16] is a benchmark suite for emerging scale-out cloud applications, however CloudSuite only includes one data analysis application: the Bayesian classification algorithm, which cannot represent the diversity of data analysis applications. Moreover, in previous work [15,16], no metric is proposed for evaluating cloud systems on the whole system level. Our previous work [17] systematically identifies three main categories of throughput oriented workloads in the cloud: services, data processing applications, and interactive real-time applications, and coin a new term high volume throughput computing (HVC) to describe them. We also release a web search engine benchmark [18], which is also an important workload in cloud systems. More details of existing benchmarks will be discussed in Section 4.6.

In this paper, we systematically analyze the limitations of the existing metrics. For example, for a cloud system running Hadoop applications, the metric of *the number of completed tasks per unit time*, is decided by the user configurations instead of the problem scale, while the other metric, (i.e., I/O rates [17]) only characterizes a subsystem capability. So they are not suitable for evaluating a cloud system running big data applications on the whole system level. We propose two simple metrics related to processing data: *data processed per second* and *data processed per Joule*. There are two complementary metrics to evaluate cloud systems running data analysis applications. The metric of data processed per second is defined as the total amount of data inputs of all jobs divided by the total running time from the submission time of the first job to the finished time of the last job, while the metric of data processed per Joule is defined as the total amount of data inputs of all jobs divided by the total energy consumed during the duration from the submission time of the first job to the finished time of the last job. From the definitions of our metrics, we can see that the measured performance metrics of the system under test depend on not only the computing speed of processors (data-centric computation), but also the efficiency of accessing memory, disk, and network I/O. Our analysis in Section 3 and experiments in Section 6.3 show our metrics are more reasonable than either the number of tasks in [14] or the number of jobs in [13].

We present a new benchmark suite, CloudRank-D, to benchmark cloud systems[2)]. Our benchmark suite provides a set of 13 representative data analysis tools, including basic operations for data analysis, classification, clustering, recommendation, sequence learning, association rule mining applications, and data warehouse operations, *a representative unified system software platform* that manages resources and provides a programming framework, and *a flexible approach to customizing their dynamic behavior characteristics*, which includes job configurations, job submission patterns, scheduling policies, and system configurations. In the CloudRank-D design, we consider diversity of data characteristics of workloads, including data semantics, data models, data sizes, and characteristics of data-centric computation, e.g., the ratio of the size of data input to that of data output. On the basis of the characteristics of data-centric computation, we classify benchmarks in CloudRank-D into four basic categories: *transformation*, *aggregation*, *summary*, *expansion*, and one derived category: *hybrid*, the details of which can be found at Section 4.2. Users can customize the benchmarks suite through choosing the full workloads of CloudRank-D or one specific basic category of workloads: transformation, aggregation, summary, and expansion according to their business requirements. Through deploying and running the customized benchmark suite, a user can report the performance metrics of the system under test. There are three purposes for using our benchmark suite. First, the metrics quantitatively measure performance of different systems, and especially measure by how much one system outperforms another. Second, the metrics can guide the optimization of the system under test. Third, according to the metrics, we can rank different systems.

The main contributions of this paper are:

1. We analyze the limitations of previous metrics in evaluating cloud systems running data processing applications, and propose two simple metrics: data processed per second and data processed per Joule. There are complementary metrics to evaluate a cloud system on a whole system level.
2. We propose a new benchmark suite CloudRank-D for evaluating cloud systems running data processing applications. ColudRank-D includes a suite of representative data analysis applications, a representative unified system software platform, and a flexible approach to customize dynamic behavior characteristics. We present a new design methodology that considers diversity of data characteristics of workloads, including data semantics, data models, data sizes, and characteristics of data-centric computation.
3. We present a methodology of evaluating and ranking cloud systems running data processing applications, and demonstrate how to use our benchmark suite—CloudRank-D to evaluate and rank cloud systems running big data applications.

Section 2 introduces the Hadoop background. In Section 3, we analyze the limitation of existing metrics, and propose two metrics to evaluate cloud systems on the whole system level. We introduce the design methodology of CloudRank-D in Section 4. Section 5 presents the usage methodology of CloudRank-D. In Section 6, we use our benchmark suite to evaluate two small-scale deployments of cloud systems. Finally, we draw conclusions in Section 7.

2 Background

This section presents the Hadoop background.

[2)] As an open source project, the CloudRank-D benchmark suite is available on our web site: http://prof.ict.ac.cn/CloudRank/.

2.1 Hadoop overview

The Apache Hadoop [8] software library is a framework that allows for distributed processing of large data sets across clusters of computers using the MapReduce programming model. In the MapReduce model, an application may consist of several dependent jobs, and a job is divided into map tasks and reduce tasks. Map tasks take key-value pairs as input to produce intermediate key-value pairs. After these are shuffled to appropriate nodes, the intermediate data is processed by reduce tasks to generate the final output key-value pairs. The input and output data are stored in Hadoop distributed file system (HDFS), which creates multiple replicas of data blocks and distributes them on compute nodes to enable reliable, extremely rapid computations.

2.2 Hadoop configurations

Hadoop has two kinds of configurations: job-specific configurations, and system-wide configurations. In other words, different jobs can have specific configurations, while system-side configurations can be set for all jobs by the system managers.

2.3 Hadoop schedulers

The scheduler in Hadoop is a pluggable component, and system managers can customize the scheduler according their requirements. There are three schedulers in Hadoop 0.20.2. By default Hadoop uses FIFO. The fair scheduler [19,20], developed by Facebook, provides fast response time for small jobs and QoS for production jobs. The capacity scheduler [21], which supports several similar features of the fair scheduler, was developed by Yahoo!. More schedulers are proposed by researchers and developers to meet their own specific requirements [22–26].

3 Metrics

In this section, we discuss why previous metrics, floating point operations and I/O operations, are not appropriate for whole system cloud benchmarking. Then, we propose our metrics. In Section 6.3, we also demonstrate the advantages of our proposed metrics over the other ones through experiments.

3.1 Floating point operations

HPC benchmarks such as Linpack use this metric to evaluate the performance of target machines. However, there are three reasons that this metric is not reasonable for evaluating cloud systems. First, data analysis applications have few floating point operations: on the average less than 3.2% among the total instructions, the details of which can be found in Section 6.3. Second, even though there are floating point operations-intensive applications, the number of floating point operations only reflects the computing power of the CPU, while cloud systems running big data applications pay more attentions to the ability of processing data, which is related to not only the computing speed of CPU, but also the efficiency of accessing memory, disk, and network I/O. Third, different from science and engineering computing, most of which are CPU bound, cloud systems have both CPU and I/O bound applications.

3.2 I/O or data rate

The I/O rate or data rate [17] is an important metric characterizing data analysis applications. However, many data analysis applications involve large amounts of computation, and hence it is inappropriate for us to adopt a metric measuring only an I/O subsystem to evaluate cloud systems on the whole system level.

3.3 Number of completed jobs

In a batch queuing system, the metric of the number of the completed jobs [27] is widely used to measure the throughput of the system. GridMix [13] also adopts this metric. For data analysis applications, even for the same code, each job may take different sizes of data input. We consider two cases: one job whose input data size is 100 GB finished in ten minutes, while ten jobs with the same code, the size of whose input data is 10 GB, respectively, also finished in 10 min. In the two cases, the system processes the same amount of data in the same time. However, if we use the number of completed jobs as the metric, the former case has a score ten times that of the latter case, which is unfair.

3.4 Number of completed tasks

HiBench [14] adopts this metric. We take Hadoop as an example to explain why this metric is not appropriate. (The same analysis can be equally performed on other programming models). A job is divided into map tasks and reduce tasks in Hadoop. Each map task takes a data split as input, and so the number of map tasks is decided by the number of data splits, which correspond to the number of data blocks. A data block is a basic unit of storage in HDFS, and its size is defined by the system managers. That is to say, the same data may be processed by a varied number of map tasks if

the data block size is set to a different value. The number of map tasks is irrelevant to the problem scale and is not reasonable in measuring the capability of cloud systems. In addition, the number of reduce tasks is also not decided by the problem scale since it is configured by users. For example, in real-world jobs, the number of reduce tasks are set to a large number to avoid intermediate data flowing through a single reduce task. As a result, the number of tasks is not related to the problem scale and cannot be used to evaluate the capability of a cloud system.

3.5 Our proposed metrics: data processed per second and data processed per Joule

For a cloud system, to measure the ability of processing data, we propose *data processed per second* and *data per Joule* as two complementary metrics to evaluate cloud systems. There are three reasons: first, the two metrics are intuitive and straightforward to reflect the throughput and energy efficiency of processing data, since cloud systems are deployed to process data. Second, the metric of data processed per second not only reflects the I/O capability of *accessing* data but also the CPU ability of *processing* data. Moreover, different from the metrics of task or job numbers, it accurately reflects the problem scale, irrespective of the configuration set by users.

Before proceeding to discuss how to calculate the metrics, we define three variables. S_{total} mean the total data input, T_{total} mean the total running time from the submission time of the first job till the completion of the final job. E_{total} means the total energy consumed from the submission time of the first job till the completion of the final job. Please note that energy consumed only includes the power consumption of the servers, including disks and network switches, excluding the power consumption of the power supply system and cooling infrastructure. We propose two approaches to calculate these metrics as follows:

$$\text{DPS} = \frac{S_{\text{total}}}{T_{\text{total}}}. \quad (1)$$

DPS is the for total data (of all jobs) *processed per second*. DPJ is the total data *processed per Joule*.

$$\text{DPJ} = \frac{S_{\text{total}}}{E_{\text{total}}}. \quad (2)$$

4 CloudRank-D design

Benchmarking is the fundamental to the evaluation of cloud systems. However, to be relevant, a benchmark suite needs to satisfy a number of properties as follows: first, the applications in the suite should consider a target class of machines [9], that is a cloud system running big data applications. Second, the workloads in the benchmark suite should be diverse enough to exhibit the range of behavior of the target applications [15]. Since service providers may deploy different applications, it is important for them to customize their chosen benchmarks to be relevant to their applications [17].

To cover workloads with diverse characteristics, we think a benchmark suite should consider:

1. A suite of representative data analysis applications.
2. The diversity of data characteristics.
3. The dynamic behavior characteristics of applications, such as the submission pattern.
4. A representative unified system software platform that manages resources and provides a programming framework.
5. The dynamic behavior characteristics of the unified system software platform, including scheduling policies and system configurations.

4.1 Representative applications

There are two approaches to build a benchmark application [9]: the top-down method and bottom-up method. The top-down method is to select a suite of representative programs, e.g., HiBench [14], MineBench [28]. The bottom-up method is to synthesize the diverse range of characteristics of workloads, e.g., WL Suite [15]. Since the best programs to use for benchmarking are real applications that the user employs regularly or simply applications that are typical [29], we use the first method to build our benchmark. We choose not only basic operations that are the basis of other algorithms, but also state-of-the-art algorithms that are widely used in both research and industrial communities as representative programs.

Our benchmark suite includes basic operations for data analysis, classification, clustering, recommendation, sequence learning, association rule mining, and data warehouse operations. Please note that in our benchmark suite, we also include *a real applications*, *ProfSearch* (http://prof.ict.ac.cn/), including the support vector machine algorithm, which distinguishes researchers' home pages from other pages, and the hidden Markov model used for Chinese

word segmentation[3].

The applications that we choose are not only representative but also *deterministic*. Since certain iterative algorithms may be non-deterministic, such as the *k*-means in our benchmark suite, we use a control (in the Hadoop implementation) that controls the initial center and the number of iterations so as to guarantee that the program behavior is deterministic.

The basic operations for data analysis include sorting, counting, regular expression matching, which are selected from the application examples of the Hadoop distribution package. Many other researchers also use these programs to evaluate their work [30].

The most popular data mining algorithms are included into our benchmark suite. For example, naive Bayes, and support vector machine are widely used in classification algorithms, and *k*-means is a popular clustering algorithm. They are used in the most influential data mining algorithms in the research community [31], and we single out them as representative applications for our benchmark suite. In Internet service companies, finding users' interests and recommending information to users are very valuable business, and hence we include an item based collaborative filtering algorithm [32] into our benchmark suite. Association rule mining is another important data mining application, which is employed in many areas including market basket analysis, web usage mining, intrusion detection, and bio-informatics [33]. We also include the frequent pattern growth algorithm, a well known association rule mining algorithm, in our benchmark suite.

Sequence learning applying a hidden Markov model is widely used in temporal pattern recognition such as speech, handwriting, gesture recognition, part-of-speech tagging, and bio-informatics. Our benchmarks contain the hidden markov model, which is implemented in a tool for Chinese part-of-speech tagging.

Data are often stored in data warehouses before mining, so data warehouse operations are significant workloads in cloud systems. And the Hive-bench [34] is the benchmarks for the data warehouse [35], which is based on Hadoop. As a result, we adopt several Hive operations in the Hive-bench as a part of our benchmark.

The details of the representative programs in CloudRank-D can be found in Table 2.

Table 2 Representative programs in CloudRank-D

	No.	Applications
Basic operations	1	Sort
	2	Word count
	3	Grep
Classification	4	Naive bayes
	5	Support vector machine
Clustering	6	*k*-means
Recommendation	7	Item based collaborative filtering
Association rule mining	8	Frequent pattern growth
Sequence learning	9	Hidden Markov
Data warehouse operations	10	Grep select
	11	Rankings select
	12	Uservisits aggregation
	13	Uservisits-rankings join

4.2 Diversity of data characteristics

Our benchmarks consider not only representative applications, but also the diversity of data characteristics. The data characteristics contain four aspects: data semantics, data models, data sizes, and categories of data-centric computation.

If there are two data sets for classification. One data set about news categorized in politics, entertainment, sports the other data set about email categorized in normal and spam emails. We say the two data sets have different data semantics. The data model describes the model of data organization, which includes *structured*, e.g., data in a relational database, *semi-structured*, e.g., an XML file, and *unstructured*, e.g., a text file.

On the basis of the work of Chen et al. [15], we classify data-centric computation into four basic categories: transformation, aggregation, summary, expansion, and one derived category: hybrid. Table 3 shows the characteristics of different categories of data-centric computation.

Table 3 Categories of data-centric computation

Category	Number of jobs	Characteristic
Transformation	$\geqslant 1$	Input ~ Output
Expansion	$\geqslant 1$	Input < Output
Aggregation	$\geqslant 1$	Input > Output
Summary	$\geqslant 1$	Input $\gg$ Output
Hybrid	> 1	At least two jobs are classified into different categories mentioned above

- A transformation application includes one or more jobs, and for each job the input sizes of jobs are approximately equal to their output sizes. In our design, when the ratio of the data output to the data input is greater than or equal to 0.9 and less than 1.1, we classify an application into the transformation category.

[3] Developed by our group, ProfSearch has been online since September, 2011. We released CloudRank-D as an open source project on our web site: http://prof.ict.ac.cn/CloudRank.

- An aggregation application includes one or more jobs, and for each job the input size is more than the output size. In our design, when the ratio of the data output to the data input is greater than or equal to 0.01 and less than 0.9, an application is classified into the aggregation category.
- A summary application contains one or more jobs, and for each job the input size is far more than its output size. In our design, when the ratio of the data output to the data input is less than 0.01, an application belongs to the summary category.
- An expansion application contain one or more jobs, and for each job the input size is less than its output size. In our design, when the ration of the data output to the data input is greater than or equal to 1.1, we classify an application into the expansion category.
- A hybrid application includes several different jobs, and at least two jobs are classified into two of the different categories mentioned above.

Table 4 presents the data characteristics of each benchmark in CloudRank-D in terms of categories of data-centric computation, data sizes, data semantics, and data models.

4.3 Dynamic behavior characteristics of applications

The submission pattern of applications can follow different probability distribution models with different average application submission intervals. According to [36], the job inter-arrival intervals from the Facebook trace roughly meet with an exponential distribution with a mean interval of 14 s.

4.4 Representative unified system software platform

The target of our benchmark suite is to evaluate cloud systems. We believe that in characterizing workloads, we should characterize not only user job behaviors but also unified system software platform behaviors. The behaviors of the unified system software platform are important factors that affect performance. We use Hadoop as a representative unified system software, and our benchmark suite considers the behavior

Table 4 Data characteristics of each benchmark in CloudRank-D

No.	Application	Category	Data size	Data semantics	Data models
1	Sort	Transformation	1.13 MB 1 GB 5.01 GB	Automatically generated	Unstructured
2	Word count	Aggregation	1.24 MB 1.03 GB 5.13 GB		
3	Grep	Summary	1.24 MB 1.03 GB 5.13 GB		
4	Naive bayes	Hybrid	15 MB 4.289 GB	20news Wikipedia	Unstructured
5	Support vector machine	Summary	8.21 MB 50 MB	ProfSearch	Unstructured
6	*k*-means	Transformation	146 KB 4.48 MB 15.77 MB	Sougou corpus	Unstructured
7	Item based collaborative filtering	Hybrid	1 KB 11 MB 123 MB	Ratings on movies	Semi-structured
8	Frequent pattern growth	Hybrid	4 MB 30 MB 34 MB 1.413 GB	Retail market basket data Click-stream data of a on-line news portal Traffic accident data Collection of web html documents	Semi-structured
9	Hidden Markov model	Expansion	8.21 MB 50 MB	ProfSearch	Unstructured
10	Grep select	Summary	486 MB	Automatically generated table	Structured
11	Ranking select	Summary	1.57 GB		
12	Uservisits aggregation	Aggregation	1.31 GB		
13	Uservisits-rankings join	Hybrid	(1.57+1.31) GB		

characteristics of Hadoop, for example scheduling policies and system configurations related resource management or fault tolerance.

4.5 Dynamic behavior characteristics of the unified system software platform

4.5.1 Scheduling policies

The main purpose of workloads is to characterize real applications. Different from traditional computing systems, cloud systems shared by data analysis applications are more complex. Since a cloud system is shared among different users or different departments submitting different jobs, and hence the scheduling policy is very important to performance, so we must consider the effect of different scheduling policies in benchmarking cloud systems.

4.5.2 System configurations

To some extents, Hadoop can be considered as the operating system of a distributed cluster because it manages the distributed resources, provides distributed file systems, schedules jobs, and handles faults. In addition to scheduling, there are many other system-related parameters that can be configured. These configurations that affect the behavior of the unified system software platform should not be ignored, e.g., the slot number, fault handling policies, and file system configurations.

4.6 Differences between CloudRank-D and other benchmark suites

MineBench [28] contains 15 representative data mining algorithms from various categories of workloads, which contain classification, clustering, association rule mining, and optimization methods. However, the applications in MineBench are not distributed, only running on single-node computers, which are significantly different from cloud systems. Compared with our CloudRank-D, MineBench does not contain basic operations, data warehouse operations, or recommendation.

GridMix [13] is a benchmark suite for Hadoop clusters. As shown in Table 5, GridMix only includes basic operations, such as sorting, reference selecting, and word counting. It can not represent diverse data analysis workloads, and hence it is not appropriate to for evaluation of cloud systems.

HiBench [14] consists of a set of Hadoop programs to evaluate and characterize the Hadoop framework. It contains micro benchmarks, web search, machine learning and HDFS benchmarks. However, HiBench does not consider varied characteristics of workloads on a shared data centers, e.g., data characteristics, job submission patterns, and scheduling policies. In addition, except for classification and clustering, HiBench does not contain recommendation, sequence learning, and association rule mining, which are very important machine learning and data mining algorithms and widely used in Internet service companies. Besides, it does not contain data warehouse operations, which are typical workloads in cloud systems running data analysis applications.

Berkeley WL Suite widely investigates MapReduce workloads. It does not, however, propose metrics for evaluating a cloud system on the whole system level. Moreover, WL Suite uses proxy code instead of real applications, and hence the reported performance metric may be biased. In characterizing workloads, it indeed considers data characteristics, number of jobs, their arrival patterns, and computation performed by

Table 5 Comparison of different benchmarks suites

		MineBench [28]	GridMix [13]	HiBench [14]	WL suite [15]	CloudSuite [16]	CloudRank-D
Representative applications	Basic operations	No	Yes	Yes	Yes	No	Yes
	Classification	Yes	No	Yes	No	Yes	Yes
	Clustering	Yes	No	Yes	No	No	Yes
	Recommendation	No	No	No	No	No	Yes
	Sequence learning	Yes	No	No	No	No	Yes
	Association rule mining	Yes	No	No	No	No	Yes
	Data warehouse operations	No	No	No	No	No	Yes
Workloads description	Submission pattern	No	No	No	Yes	No	Yes
	Scheduling policies	No	No	No	No	No	Yes
	System configurations	No	No	No	No	No	Yes
	Data models	No	No	No	No	No	Yes
	Data semantics	No	No	No	No	No	Yes
	Multiple data inputs	Yes	Yes	No	Yes	No	Yes
	Category of data-centric computation	No	No	No	Yes	No	Yes

the jobs. However, the effects of scheduling policies and system configurations are ignored, which are important for a shared cloud system.

CloudSuite [16] is a benchmark suite to characterize scale-out cloud applications. Its target is to provide benchmarks for architecture research instead of evaluating a cloud system on the whole system level. In addition, it only includes one data analysis application: classification.

Table 5 and Table 6 summarize the differences of CloudRank-D from other benchmark suites.

Table 6 Different targets and metrics among benchmark suites

	Targets	Metric
MineBench [28]	Data mining algorithm on single-node computers	No
GridMix [13]	Hadoop framework	Number of jobs and running time
HiBench [14]	Hadoop framework	Job running time, the number of tasks completed per minute
WL suite [15]	Hadoop framework	No
CloudSuite [16]	Architecture research	No
CloudRank-D	Evaluating cloud systems on the whole system level	Data processed per second and data processed per joule

5 How to use CloudRank-D

There are three scenarios of using our benchmark suite: first, using CloudRank-D, a user can quantitatively measure the performance metrics of different cloud systems, and especially measure by how much one system outperforms another. For example, a user can measure two clusters with different configurations, and then find the best one for data processing applications. Second, CloudRank-D can guide the optimization of a system under test. For example, the user can tune different configurations of node operation systems, such as the page size or socket buffer size, or scheduling policies. Third, according to the reported performance metrics of CloudRank-D, we can rank different systems.

Figure 2 shows the methodology of evaluating and ranking cloud systems. First, users customize the CloudRank-D benchmark suite by choosing the full workloads or one specific basic category: transformation, aggregation, summary,

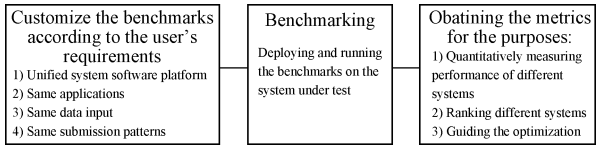


Fig. 2 CloudRank-D usage methodology

and expansion according to their business requirements. If a user runs mixed workloads or does not fully understand their workloads, he or she can choose to run the full benchmark suite. In evaluating different cloud systems, we deploy the same benchmark suite that includes the unified system software stack of the same configurations and the representative workloads that have the same data inputs.

Second, users run the same benchmark suite according to the specified submission orders following the specified probability model with the specified average submission interval. We use a tuple (No. of a workload, the size of data input) to describe a workload, and a series of tuples to describe the submission orders of each workload.

Finally, users obtain the quantitative metrics of CloudRank-D for the system under test. After obtaining the performance metrics, we calculate the metrics according to Eqs. (1) and (2), which is the basis for our ranking and choosing systems.

6 Evaluation

In this section, first, we introduce the testbeds configurations in Section 6.1. In Section 6.2, we present the CloudRank-D configurations. In Section 6.3, through experiments, we demonstrate that our metrics are more reasonable than two other metrics. In Section 6.4, we evaluate and rank two deployments of cloud systems running full workloads and customized workloads. Meanwhile, we also investigate the effects of different job submission intervals and scheduling policies.

6.1 Testbeds

The testbeds are two clusters: Cluster one and Cluster two, which are composed of Xeon and Atom processors, respectively. Cluster one contains seven homogeneous slave nodes and one master node. Each node consists of one Intel Xeon quad-core processor and 4 GB of memory, and runs the CentOS 5.5 operating system. Cluster two contains seven homogeneous slave nodes and one master node. Each node consists of one Intel Atom dual-core processor and 2 GB of memory, and runs the CentOS 5.5 operating system. Table 7 and Table 8 list the important configuration parameters of Xeon and

Table 7 Configuration details of a Xeon processor

CPU type		Intel CPU core	
Intel ® Xeon		4cores@1.6 GHz	
TLB	L1 I/D Cache	L2 Cache	Memory
256 entries	32 KB	4 096 KB	4 GB

Atom nodes. Figure 3 shows the system deployment of Cluster one and Cluster two.

Table 8 Configuration details of an Atom processor

CPU type		Intel CPU core	
Intel ® Atom D510		2cores@1.66 GHz	
TLB	L1 I/D Cache	L2 Cache	Memory
256 entries	24 KB	512 KB	2 GB

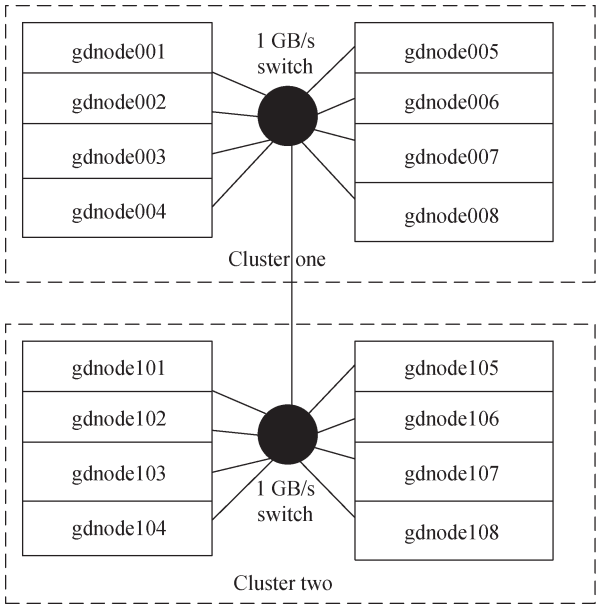


Fig. 3 Testbeds configurations

6.2 CloudRank-D configurations

The details of the benchmark suite are as follows:

- **Software stack** We deploy the same unified system software stack: Hadoop (version 0.20.2), Hive (version 0.6.0), and Mahout (version 0.6). We adopt the fair scheduler and set five queues by default to simulate the situation of multiple users. The number of map slots and reduce slots are 4 and 2, respectively. Other system configurations are left to default values from Hadoop 0.20.2.
- **Benchmarks and their data inputs** The details of benchmarks and their data inputs can be seen in Table 4.
- **Job submission patterns** We submit the benchmarks according to the exponential distribution with a specified mean of 14 s. For each run of evaluating different cloud systems, the submitted orders of different workload are defined and deterministic.

6.3 Why our metrics are more reasonable than other metrics?

Figures 4 shows the ratio of floating point operations to total instructions for each workload. We can see that all applications have only a few floating point operations (less than 3.2%), so the metric of PLOPS is not reasonable in evaluating cloud systems running data processing applications.

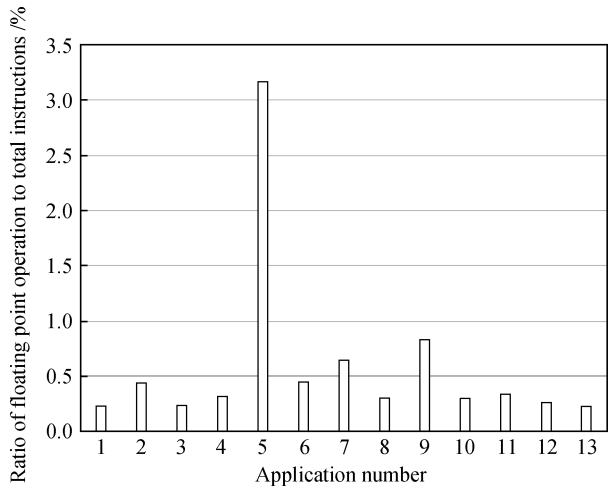


Fig. 4 Ratio of floating point operations to total instructions of each workload in CloudRank-D

We construct a set of different experiments to show our metrics are more reasonable than other metrics.

For workload No. 2 in Table 2, word count, we generated randomly two data sets which are text files: a single 40 GB data set and five 8 GB data sets. We run three different experiments. In Test 1, we run a total of five jobs, and each job processes 8 GB data. According to the default Hadoop configuration, each job has 160 map tasks and one reduce task. Please note that the number of map tasks is decided by the number of data blocks by default. So the total number of map tasks is 805, and the total data input is 40 GB. The five jobs are submitted at the same time.

In Test 2, we run one job on the 40 GB data set. By default, the job has 720 map tasks, which is decided by the number of data blocks, and we set the configuration of *mapred.reduce.tasks* to five to make the reduce task the same as that in Test 1.

In Test 3, we also run one job on the same data set. However, we set the configuration of *mapred.max.split.size* to a half of the block size so that the number of map tasks is changed to 1 360. We also set the configuration of *mapred.reduce.tasks* to five. The detail is shown in Table 9.

In three experiments, the same program (with different jobs) processes the same amount of data input on the same

Table 9 Different configurations and performance results of three tests running the same program and the same amount of total data inputs

	Job number	Map task number	Reduce task number	Total task number	Total data size/GB	Total time	Total power
Test 1	5	800	5	805	40	1 203	1 265 956 Joule
Test 2	1	720	5	725	40	1 092	1 266 165 Joule
Test 3	1	1 360	5	1 365	40	1 242	1 354 946 Joule

system under the test, so the reported metrics that measure the performance of the system under test should not significantly vary among three experiments. However, as we can see in Fig. 5, if we choose the number of completed jobs per second to be the metric, which is used in GridMix [13], the reported metric in Tests 2 and 3 is just 0.20 times of that of Test 1. In terms of the number of the completed tasks per second [14], the result of Test 3 is 1.6 times of that of Tests 2 and 3 (which is shown in Fig. 6). If we choose our proposed metrics: the data processed per second, the reported performance metric in Fig. 7 changes slightly (less than 5%), and hence this metric is more reasonable than the other two metrics.

We also report performance metrics of the number of the completed jobs per Joule, the number of the completed tasks per Joule, and data processed per joule in Figs. 8–10, respectively. From the three figures, we can observe that the number of completed jobs per joule and the number of the completed tasks per Joule are unrelated to the problem scales, while our

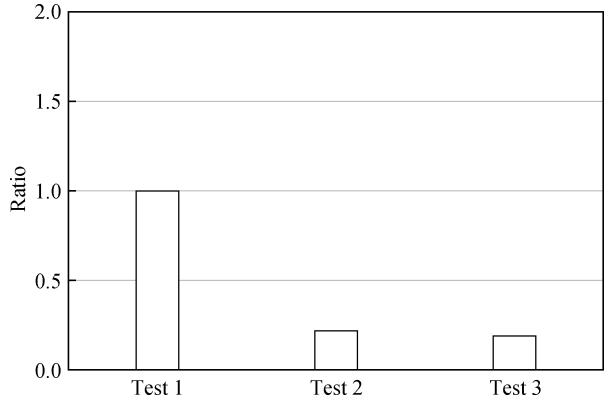


Fig. 5 Ratios of metrics of different tests to that of Test 1 in terms of the number of the completed jobs per second

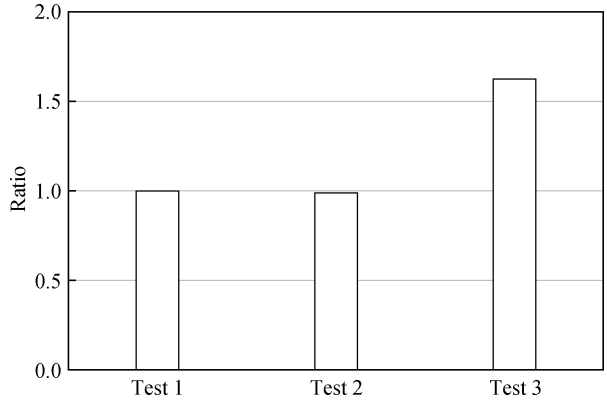


Fig. 6 Ratios of metrics of different tests to that of Test 1 in terms of the number of the completed tasks per second

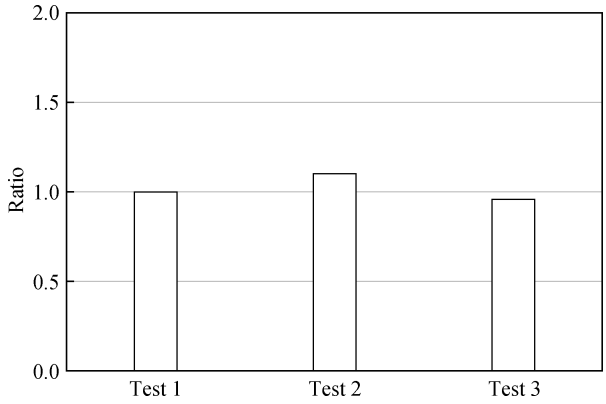


Fig. 7 Ratios of metrics of different tests to that of Test 1 in terms of our metric: data processed per second

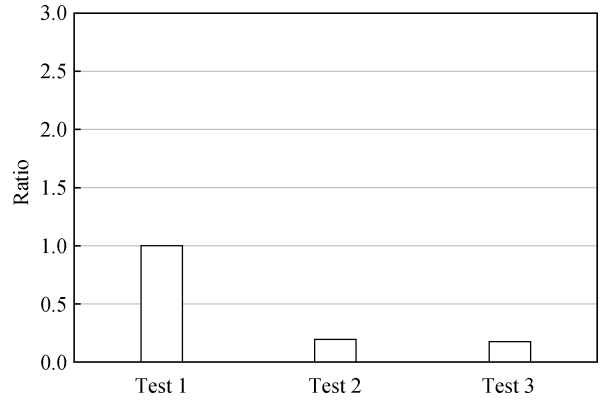


Fig. 8 Ratios of metrics of different tests to that of Test 1 in terms of the number of the completed jobs per Joule

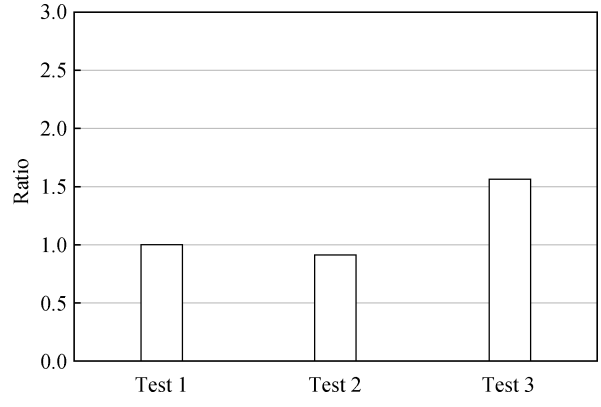


Fig. 9 Ratios of metrics of different tests to that of Test 1 in terms of the number of the completed tasks per Joule

metric data processed per joule can reflect the problem scale, so our metric measuring energy efficiency is more reasonable than the other two metrics.

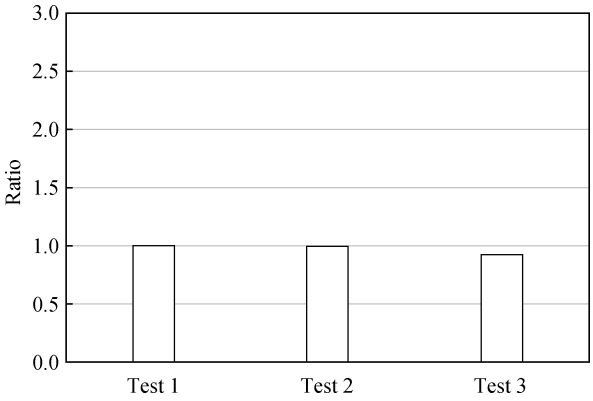


Fig. 10 Ratios of metrics of different tests to that of Test 1 in terms of our metric: data processed per Joule

6.4 Evaluating and ranking two cloud systems using CloudRank-D

In this section, we use CloudRank-D to evaluate and rank two cloud systems that consis of Xeon and Atom processors. We also investigate the effects of job submission intervals, scheduling policies, and different categories of workload on the reported metrics of the systems under test.

6.4.1 Running full workloads

We run the full workloads described in Table 4 on: Cluster one and Cluster two. For the full workloads, the submission orders are defined as follows:

((6, 146 KB), (4, 15 MB), (7, 11 MB), (8, 34 MB), (2, 1.03 GB), (3, 1.24 MB), (5, 50 MB), (13, 2.88 GB), (6, 4.48 MB), (4, 4.289 GB), (1, 1 GB), (7, 123 MB), (8, 30 MB), (12, 1.31 GB), (1, 1.13 MB), (5, 8.21 MB), (11, 1.57 GB), (6, 15.77 MB), (3, 1.03 GB), (2, 5.13 GB), (7, 1 KB), (8, 4 MB), (8, 1.413 GB), (2, 1.24 MB), (1, 5.01 GB), (9, 50 MB), (3, 5.13 GB), (5, 8.21 MB), (10, 486 MB)).

We submit the workloads of CloudRank-D according to the exponential distribution with a mean submission interval of 14 s. We use the fair scheduler and build five queues to simulate the situation of multiple users.

Figure 11 shows that the running time of CloudRank-D on Cluster one is nearly half of that on Cluster two, while the power consumption of Cluster one is 2.6 times of that of Cluster two. Figure 12 shows the reported performance metrics of two clusters. We can see that in terms of the data processed per second, the efficiency of Cluster one (Xeon processors) is two times of that of Cluster two, while in terms of the data processed per Joule, the energy efficiency of Cluster two (Atom processors) is 2.4 times of that of Cluster one.

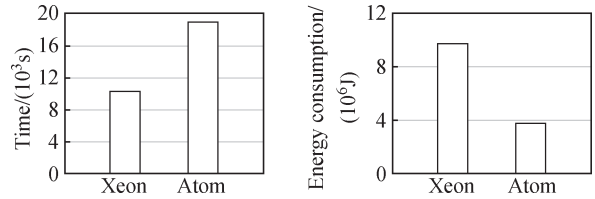


Fig. 11 Running time and energy consumption of running CloudRank-D on Cluster one (Xeon processors) and Cluster two (Atom processors)

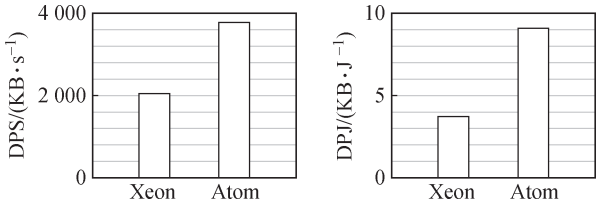


Fig. 12 Performance of Cluster one (Xeon processors) and Cluster two (Atom processors) in terms of two metrics

6.4.2 Effect of job submission intervals

On Cluster one, we submit the full workloads specified in Section 6.4.1 with the mean submission interval of 14 s, 70 s, 140 s, 210 s, and 700 s, respectively. From Fig. 13, we can observe that when the average submission interval is 70 s, peak performance metrics are reported, which have two reasons as follows: when the submission interval is smaller than 70 s, more workloads are submitted to the Hadoop system, and resource competition among the workloads degrade performance. When the submission interval is larger than 70 s, however, the resource utilization is low so that the reported performance is lower than reported when the interval is 70 s.

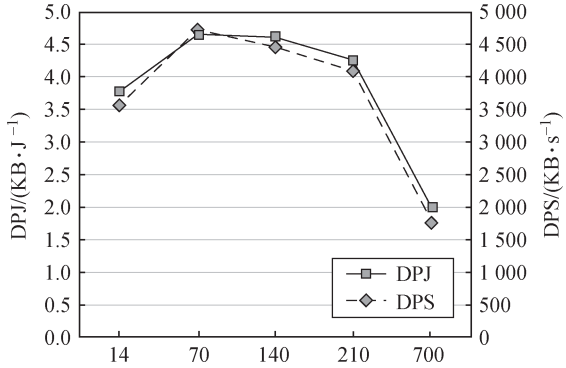


Fig. 13 Effects of different submission intervals on the reported performance metrics of Cluster one

6.4.3 Effect of different scheduling policies

We investigate the effects of different scheduling policies on the performance of a cloud system. Figure 14 shows the ex-

perimental results when choosing between the Fair scheduler and the FIFO scheduler. We submit the applications according to the exponential distribution with the mean submission interval of 14 s. Both the data processed per second (DPS) and the data processed per Joule (DPJ) using the fair scheduler are 1.1 times that of using the FIFO scheduler. The reason is that when using the fair scheduler jobs with different resource demands avoid competition for resources since they share machines and run concurrently. For example, when a cpu-intensive job is co-running with an I/O intensive job, the interference will be slighters than the case that two I/O intensive jobs are co-running. These experimental results show we can optimize the performance of the system under test through tuning the configuration of the system software.

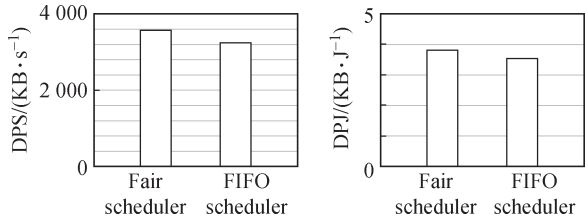


Fig. 14 Effects of different scheduling policies on the reported performance metrics of Cluster one

6.4.4 Evaluating and ranking two cloud systems using different categories of workloads

We investigate the effect of different categories of workloads on the reported performance metrics. We choose three categories of workloads: *hybrid workloads* a combination of workloads belonging to the hybrid category, *summary workloads* which are a combination of workloads belonging to the summary category, and *partial workloads* that are a mix of workloads belonging to four categories of workloads: aggregation, hybrid, summary, and transformation. The submission orders of hybrid workloads are defined as ((4, 15 MB), (13, 2.88 GB), (7, 11 MB), (8, 4 MB)). The submission orders of summary workloads are defined as ((3, 1.03 GB), (11, 1.57 GB), (10, 486 MB), (5, 8.21 MB)). The submission orders of partial workloads are defined as ((12, 1.31 GB), (2, 1.03 GB), (2, 1.24 GB), (2, 5.13 GB), (4, 15 MB), (13, 2.88 GB), (7, 11 MB), (8, 4 MB), (6, 146 KB), (6, 4.48 MB), (6, 15.77 MB), (3, 1.03 GB), (11, 1.57 GB), (10, 486 MB), (5, 8.21 MB), (1, 1 GB), (1, 1.13 MB), (1, 5.01 GB)). The submission intervals follow an exponential distribution with the mean submission interval of 14 s. We run three workloads on Cluster one, respectively.

The performance metrics are reported in Figs. 15 and 16. We can see that running the summary workloads has higher performance in terms of both data processed per second and data processed per joule than that of running the hybrid workloads and the partial workloads. This is because the summary workloads have less intermediate data than that of hybrid workloads. For example, grep (No.3), belonging to the summary workloads, has only 1 120 bytes of shuffle data, and support vector machine (No.5), belonging to the summary workloads, has no shuffle data, while naive bayes (No.4) has 9 350 453 bytes of shuffle data, so running the summary workloads has fewer disk and network accesses. In terms of the number of disk and network accesses, the partial workloads are between the summary workloads and hybrid workloads, so the performance metrics of the partial workloads are also between that of the summary workloads and the hybrid workloads.

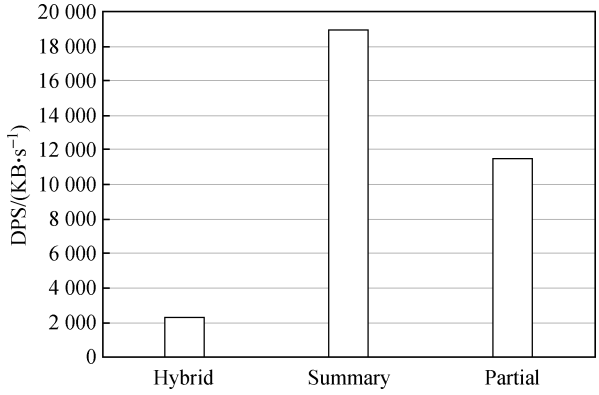


Fig. 15 Performance of the Xeon cluster in terms of DPS using different categories of workloads

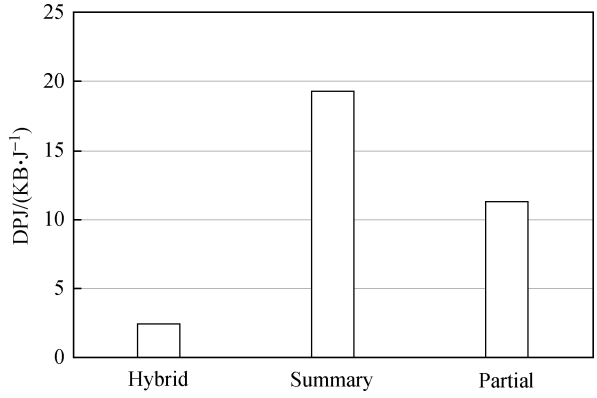


Fig. 16 Performance of the Xeon cluster in terms of DPJ using different categories of workloads

From the above experiments, we can observe that different categories of workloads have non-negligible effects on the performance metrics of the system under test, and hence users should customize the benchmark suites according to their requirements.

7 Conclusion and future work

Benchmarking is the quantitative foundation of computer system and architecture research. However, previous benchmarks are not reasonable for evaluating cloud systems running big data applications because of the lack of diversity of data analysis workloads.

In this paper, we analyzed the limitations of previous metrics in evaluating cloud systems running data processing applications, and proposed two simple metrics: data processed per second and data processed per Joule as two complementary metrics to evaluate a cloud system on the whole system level. Both our experiments and analysis show our proposed metrics outperform other metrics.

We presented a new benchmark suite CloudRank-D for evaluating cloud systems, including a suite of representative data analysis applications, a representative unified system software platform, and a flexible approach to customize their dynamic behavior characteristics. In the CloudRank-D design, we presented a new design methodology that considers diversity of data characteristics of workloads in terms of data semantics, data models, data sizes, and characteristics of data-centric computation.

We presented a methodology of evaluating and ranking cloud systems, and demonstrated how to use our benchmark, CloudRank-D, to evaluate and rank two small-scale deployments of cloud systems.

In future work, we will perform the following research efforts. First, we will further investigate the scalability of CloudRank-D in terms of both the scale of the system under test and the size of data inputs. Second, we will explore the effects of more system configurations, including hardware updates. Third, we plan to release a top 500 clouds list in cooperation with other partners from both academic and industrial areas.

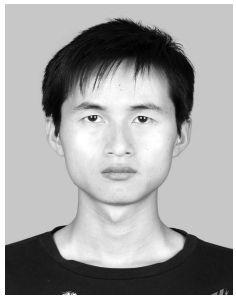
Acknowledgements We are very grateful to anonymous reviewers. This work was supported by the National Basic Research Program of China (973 Program) (2011CB302500), the National Natural Science Foundation of China (Grant No. 60933003), and the Strategic Priority Research Program of the Chinese Academy of Sciences (XDA06010401).

References

1. Armbrust M, Fox A, Griffith R, Joseph A, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. Above the clouds: a Berkeley view of cloud computing. Deptartment Electrical Engineering and Compututer Sciences, University of California, Berkeley, Report UCB/EECS, 2009, 28
2. Barroso L, Hölzle U. The datacenter as a computer: an introduction to the design of warehouse-scale machines. Synthesis Lectures on Computer Architecture, 2009, 4(1): 1–108
3. http://wiki.apache.org/hadoop/PoweredBy
4. Wang P, Meng D, Han J, Zhan J, Tu B, Shi X, Wan L. Transformer: a new paradigm for building data-parallel programming models. IEEE Micro, 2010, 30(4): 55–64
5. Isard M, Budiu M, Yu Y, Birrell A, Fetterly D. Dryad: distributed data-parallel programs from sequential building blocks. ACM SIGOPS Operating Systems Review, 2007, 41(3): 59–72
6. Thusoo A, Shao Z, Anthony S, Borthakur D, Jain N, Sen Sarma J, Murthy R, Liu H. Data warehousing and analytics infrastructure at Facebook. In: Proceedings of the 2010 International Conference on Management of Data. 2010, 1013–1020
7. Dongarra J, Luszczek P, Petitet A. The linpack benchmark: past, present and future. Concurrency and Computation: Practice and Experience, 2003, 15(9): 803–820
8. http://hadoop.apache.org
9. Bienia C. Benchmarking modern multiprocessors. PhD thesis. Princeton University, 2011
10. http://www.spec.org/cpu2006
11. http://www.spec.org/web2005
12. http://www.tpc.org/information/benchmarks.asp
13. http://hadoop.apache.org/mapreduce/docs/current/gridmix.html
14. Huang S, Huang J, Dai J, Xie T, Huang B. The hibench benchmark suite: characterization of the mapreduce-based data analysis. In: Proceedings of the 26th IEEE International Conference on Data Engineering Workshops, ICDEW'10. 2010, 41–51
15. Chen Y, Ganapathi A, Griffith R, Katz R. The case for evaluating mapreduce performance using workload suites. In: Proceedings of the IEEE 19th International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems, MASCOTS'11. 2011, 390–399
16. Ferdman M, Adileh A, Kocberber O, Volos S, Alisafaee M, Jevdjic D, Kaynak C, Popescu A, Ailamaki A, Falsafi B. Clearing the clouds: a study of emerging scale-out workloads on modern hardware. In: Proceedings of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems. 2012, 37–48
17. Zhan J, Zhang L, Sun N, Wang L, Jia Z, Luo C. High volume throughput computing: identifying and characterizing throughput oriented workloads in data centers. In: Proceedings of the 2012 Workshop on Large-Scale Parallel Processing. 2012
18. Xi H, Zhan J, Jia Z, Hong X, Wang L, Zhang L, Sun N, Lu G. Characterization of real workloads of web search engines. In: Proceedings of the 2011 IEEE International Symposium on Workload Characterization, IISWC'11. 2011, 15–25
19. http://hadoop.apache.org/common/docs/r0.20.2/fair_scheduler.html
20. Zaharia M, Borthakur D, Sarma J, Elmeleegy K, Shenker S, Stoica I. Job scheduling for multi-user mapreduce clusters. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-55, 2009
21. http://hadoop.apache.org/common/docs/r0.20.2/capacity_scheduler.

html

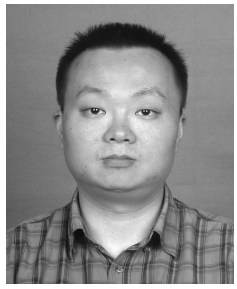
22. Rasooli A, Down D. An adaptive scheduling algorithm for dynamic heterogeneous Hadoop systems. In: Proceedings of the 2011 Conference of the Center for Advanced Studies on Collaborative Research. 2011, 30–44
23. Sandholm T, Lai K. Dynamic proportional share scheduling in Hadoop. In: Job Scheduling Strategies for Parallel Processing. 2010, 110–131
24. Wolf J, Rajan D, Hildrum K, Khandekar R, Kumar V, Parekh S, Wu K, Balmin A. Flex: a slot allocation scheduling optimizer for mapreduce workloads. Middleware 2010, 2010, 1–20
25. Lee G, Chun B, Katz R. Heterogeneity-aware resource allocation and scheduling in the cloud. In: Proceedings of the 3rd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud'11. 2011
26. Yong M, Garegrat N, Mohan S. Towards a resource aware scheduler in hadoop. In: Proceedings of the 2009 IEEE International Conference on Web Services. 2009, 102–109
27. Wang L, Zhan J, Shi W, Yi L. In cloud, can scientific communities benefit from the economies of scale? IEEE Transactions on Parallel and Distributed Systems, 2012, 23(2): 296–303
28. Narayanan R, Ozisikyilmaz B, Zambreno J, Memik G, Choudhary A. Minebench: a benchmark suite for data mining workloads. In: Proceedings of the 2006 IEEE International Symposium on Workload Characterization. 2006, 182–188
29. Patterson D, Hennessy J. Computer organization and design: the hardware/software interface. Morgan Kaufmann, 2009
30. Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Communications of the ACM, 2008, 51(1): 107–113
31. Wu X, Kumar V, Ross Quinlan J, Ghosh J, Yang Q, Motoda H, McLachlan G, Ng A, Liu B, Yu P, Zhou Z-H, Steinbach M, Hand D, Steinberg D. Top 10 algorithms in data mining. Knowledge and Information Systems, 2008, 14(1): 1–37
32. Linden G, Smith B, York J. Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing, 2003, 7(1): 76–80
33. http://en.wikipedia.org/wiki/Association_rule_learning
34. https://issues.apache.org/jira/browse/HIVE-396
35. http://hive.apache.org/
36. Zaharia M, Borthakur D, Sen Sarma J, Elmeleegy K, Shenker S, Stoica I. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In: Proceedings of the 5th European Conference on Computer Systems. 2010, 265–278



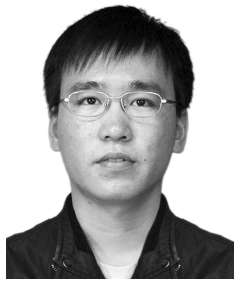
Chunjie Luo is a Master's student at the Institute of Computing Technology, Chinese Academy of Sciences. His research interest focuses on data center computing. He received his BS in 2009 from Huazhong University of Science and Technology in China.

Jianfeng Zhan received his PhD in Computer Engineering from the Chinese Academy of Sciences, Beijing, China, in 2002. He is currently an associate professor of computer science with Institute of Computing Technology, Chinese Academy of Sciences. He was a recipient of the Second-class Chinese National Technology Promotion Prize in 2006, and the Distinguished Achievement Award of the Chinese Academy of Sciences in 2005.

Zhen Jia is a PhD candidate in Computer Science at the Institute of Computing Technology, Chinese Academy of Sciences. His research focuses on parallel and distributed systems, benchmarks, and data center workload characterization. He received his BS in 2010 from Dalian University of Technology in China.



Lei Wang received the his MS in Computer Engineering from the Chinese Academy of Sciences, Beijing, China, in 2006. He is currently a senior engineer with the Institute of Computing Technology, Chinese Academy of Sciences. His current research interests include resource management of cloud systems.



Gang Lu received his Bachelor's degree in 2010 from Huazhong University of Science and Technology in China, in computer science. He is a PhD candidate at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include distributed and parallel systems.

Lixin Zhang is a professor and vice general engineer of Systems at the Institute of Computing Technology, Chinese Academy of

Sciences. He is the director of the Advanced Computer Systems Laboratory. His main research areas include computer architecture, data center computing, high performance computing, advanced memory systems, and workload characterization. Dr. Zhang received his BS in computer science from Fudan University in 1993 and his PhD in computer science from the University of Utah in 2001. He was previously a research staff member at IBM Austin Research Lab and a Master Inventor of IBM.

Cheng-Zhong Xu received his PhD from the University of HongKong in 1993. He is currently a tenured professor of Wayne State University and the director of the Center for Cloud Computing in Shenzhen Institute of Advanced Technology of CAS. His research interest are in parallel and distributed systems, and cloud computing. He has published more than 180 papers in journals and conferences. He serves on a number of journal editorial boards, including IEEE TPDS and JPDC. He was a recipient of the Faculty Research Award, Career Development Chair Award, and the President's Award for Excellence in Teaching of WSU. He was also a recipient of the "Outstanding Oversea Scholar" award of NSFC.

Ninghui Sun is a professor and the director of Information Computing Technology (ICT) of the Chinese Academy of Sciences (CAS). He graduated at Peking University in 1989, and received his MS and PhD degrees from ICT of CAS in 1992 and 1999, respectively. Prof. Sun is the architect and main designer of the Dawning2000, Dawning3000, Dawning4000, Dawning5000, and Dawning6000 high performance computers.