Case Study on Improving Hadoop MapReduce Performance using SSD

(Vincent, Olivier BROT Sik Hee FUNG Pavitra NAIK Wen YANG)

1. Abstract

The performance of solid state drive (SSD) is better than traditional hard disk (HDD) in many aspects. However, in terms of cost-per-bit, the cost of SSD is still higher than traditional HDD. Improving the performance of Hadoop framework by the total replacement of HDD by SSD is definitely not cost-effective. Our study attempts to find out the options in using SSD in boosting Hadoop performance and to evaluate the performance gain under various types of workload. A use case of using SSD [12] demonstrates improvement in performance and cost-effectiveness by store intermediate data on SSD. However, [12] only provides performance measurements under Terasort and DFSIO which is quite limited. To respond to this finding, we study big data benchmark tools. A good big data benchmark tool should have taken into account the 4V properties [19]. Bigdatabench [14] may be the best choice to evaluate big data system architectures. A few issues in the experiment design and analysis are also discussed.

2. SSD versus HDD

2.1 General description of SSDs and HDDs

Solid-state drives (SSDs) are recent fast storage media based on flash memory, while hard-disk drives (HDDs) are the standard storage media using rotational magnetic disks. In general, it is commonly believed that SSDs are faster and more durable than traditional HDDs. Unlike HDDs, SDDs do not have mechanic structures (rotating platters, motor). This allows faster seek times and high IOPs (input output operations per second). Also, SSDs are preferred in many industries because they are more portable than HDDs. Table 2.1 below shows the major differences between SSDs and HDDs.

	SSDs	**HDDS**
Seek time	Low latency with small seeking time, typical value: 0.1ms. [8]	High latency, due to rotating disk, typical value: 12ms. [7]
Data speed	Up to 1.3GBps in sequential read and write, depending on types and manufacture.	About 120MBps in sequential read and write.
Heat, noise, and power consumption	Less power consumption, no heat nor noise.	More power consumption to rotate the platters, generating heat and noise.
Defragmentation	No defragmentation necessary	Need periodical defragmentation

Table 2.1, differences between SSDs and HDDs

Owing to the decreasing price [1], more and more industries are replacing their HDDs by SSDs. Usage of SSDs in data warehouse delivers low-latency as well as accelerates parallel query greatly in big data analysis [6]. Given the growth of data warehouses and the decrease of SSDs cost, the quest of the best SSD implementation is becoming critical.

2.2 Using SSDs in MapReduce and Hadoop Distributed File System

MapReduce [3] and HDFS (The Hadoop Distributed File System) [4] are two important modules of the Apache™ Hadoop® project. MapReduce is a powerful framework meant for analyzing large set of data. HDFS, one kind of distributed file system, yet different to others, is custom designed for commodity hardware. MapReduce processes big data set based on HDFS.

The elementary hardware strategy is to prefer HDDs over SSDs for economical reason. Since HDFS provides high fault-tolerant to fit low-cost hardware, replacing all HDDs with SSDs in HDFS is not such a good idea since HDDs still have many merits over SSDs such as bigger volumes. Hence, to get the best cost-effective of using SSDs, combining SSDs and HDDs in HDFS is a natural solution. To get the right combination, we need a close examination on the MapReduce working flow as described in Figure 2.1.

MapReduce splits raw input data set into many small partitions and distributes these partitions to HDFS on many parallel machines. Each small partition is processed by a single map task on a local machine. The shuffle stage gathers the output of those many map tasks and send them to the reduce stage with different keys in <key, value> list. Thus, sorting and classifying are often involved in this stage. The reduce stage optimizes the input from shuffle on each different key and outputs the result to disk.

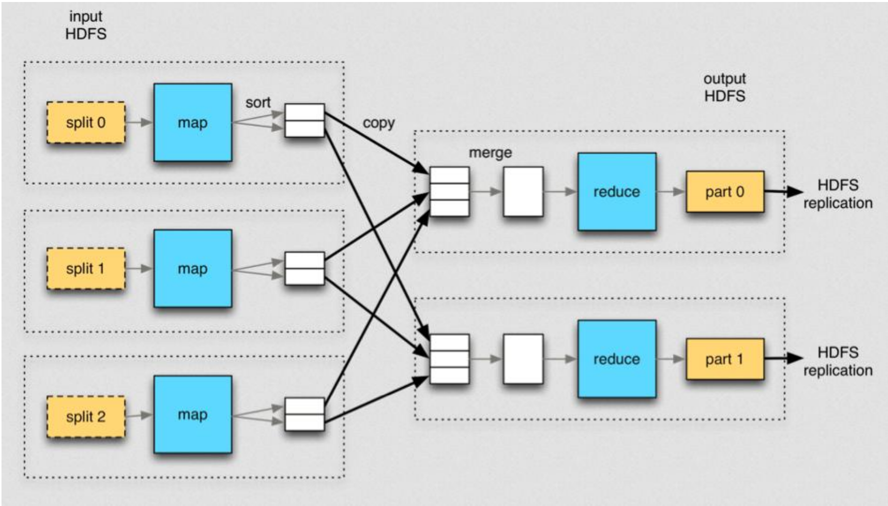

Figure 2.1: MapReduce Dataflow. Source: [2]

By understanding the work flow of MapReduce, we can distinguish two kinds of data access in Table 2.2. [9]

Data Access	Description
Large, sequential access	• Read the raw input data • Split the input into small partitions • Write result
Small, random access	• Shuffle intermediate data

Table 2.2, different kinds of data access in MapReduce [9]

At the beginning of a MapReduce job, HDFS loads the entire data set and split it into many partitions. There is often a large amount of sequential data access involved in this stage. Given the typical reading bandwidth of respectively 481 MB/s and 140 MB/s for SSDs and HDDs [10], we combine four HDDs together and achieve a total bandwidth of 140*4 =560MB/s, which is roughly equal to the bandwidth of one SSDs. Hence, assuming that the price of HDDs is four times smaller than SSDs, the price-performance ratio is roughly the same.

For the intermediate data which is generated by map stage, HDFS must perform a lot of random I/O. Merge sort and temporary data writing are often involved in this stage. Due to these sorting tasks and random accesses, the low latency of SSDs helps a lot accelerating this process.

As a result, for large amount of sequential data access, HDDs and SSDs may perform more or less similarly. However, in the shuffle stage, the low latency of SSDs improves the performance especially in heavy shuffle job. From this result, we can conclude that if best performance is the main target, using SSDs only is the better choice.

2.3 Intel suggestions [11]

By comparing the performance on HDFS with different storage types and different combinations of them, Intel suggests that many factors should be taken into account, such as hardware design of single machine, bandwidth in network and the internal configuration of HDFS [10].

- Disk Bandwidth Limitation

The figure 2.2 below illustrates a motherboard design where the disk slots connect to two SATA controllers [10]. Those upper level hardware controllers will affect the total bandwidth available.

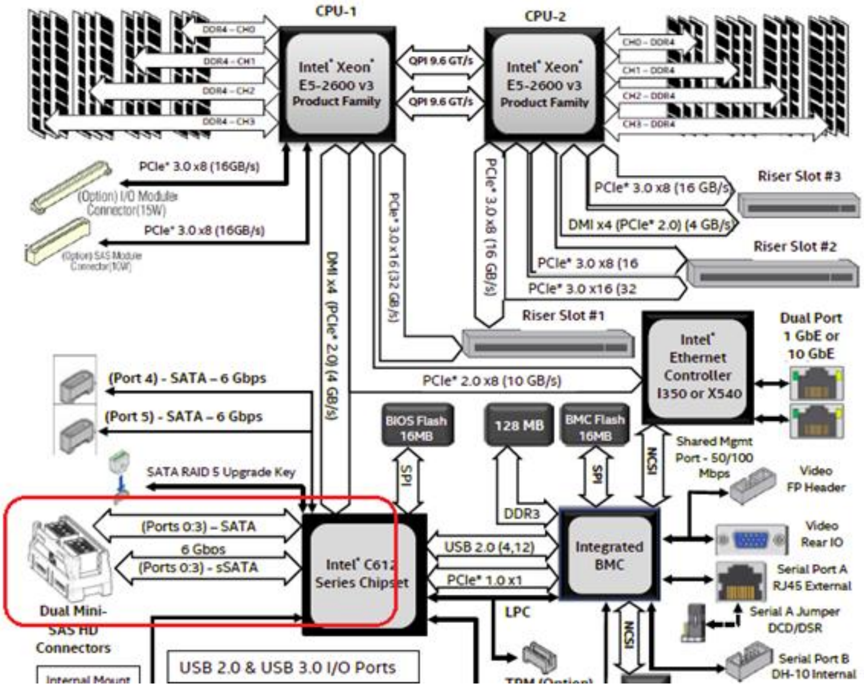

Figure 2.2: Hardware design of PC motherboard Source: [10]

In this configuration, the maximum bandwidth available for a single machine is 2*6 Gbps =1536 MB/s. Table 2.3 below shows the bandwidth of different possible combinations of SSDs and HDDs, given a writing bandwidth of respectively 447 MB/s and 127 MB/s for SSDs and HDDs.

Combination	Bandwidth	Maximum Bandwidth
7 SSDs	447*7 = 3129 MB/s	1536 MB/s
7 HDDs	127*7 = 889 MB/s	1536 MB/s
2 SSDs and 5 HDDs	447*2 + 127*5 = 1529 MB/s	1536 MB/s

Table 2.3 Bandwidth for different combination of SSDs and HDDs

We know that our total output bandwidth of HDDs and SSDs due to the SATA controllers is 1536 MB/s. In this case, a configuration with 7 SSDs will not bring much more than one that uses 2 SSDs and 5 HDDs because it is affected by the SATA limitation. The second and the third configuration are not. The third configuration will be an appropriate choice here since the total bandwidth matches the upper SATA controller.

A good option to avoid such design limitation and gain more bandwidth on a machine would be to use more HBA cards.

- Network bandwidth between nodes

Network bandwidth limits the data transmission speed between different machines. It is an important issue here since HDFS is a distributed file system. Hence, the total bandwidth must be carefully checked when introducing SSDs. If the network bandwidth becomes the bottleneck, the performance we gained by using SSDs may be lost. For example, if we have a HDFS network throughput of 11 Gbps, we do not want to use the typical 10Gbps network cable as default. Similarly, the bandwidth of internal switches and routers also needs scrutiny.

- Internal configuration of HDFS

Owing to the differences we mentioned earlier between SSDs and HDDs, HDFS should not use them in the same way. In other words, using the system default and treating them as equal will not help us get the best performance. It is obvious that we want to send to SSDs tasks involving many random accesses since it has higher IOPs than HDDs.

The synchronize methods is an important factor that needs our close attention. Since the same operation is much slower on HDDs than SSDs, there is a big chance that the SSDs will spend a lot of time waiting the lock from HDDs. This may upset the performance of SSDs.

In sum, we need to carefully check the configurations to avoid limitation derived from HDFS setting when combining SSDs and HDDs.

3. An implementation Use Case [12]

We have seen previously that Hadoop design was initially created ignoring SSDs and considering HDDs as its only storage device. Therefore, it naturally gives an advantage to HDDs.

However, the below section proves that we still can leverage the SSDs benefits. We will detail an experiment [11] whose goal was to determine what is the best way to use SSDs with the Hadoop MapReduce Framework. We will first expose some tests that have been done using different combinations of media storage; we will then analyze their results and do some recommendations based on the Performance/Cost ratio.

- Media storage versus network

This DFSIO test measures HDFS performance, compared to the network bandwidth and the media storage. We compare cluster configurations with 1 GBit vs 20 Gbit Ethernet links, combined to 3 different storage devices: 1 HDD, 2 HDDs (RAID) and 1 SSD.

The figure 2.2 shows that the read performance is not affected by Network bandwidth. We could have anticipated that result: indeed, the map tasks read data from a local storage. Then, the SSD shows a throughput significantly greater with SSD than HDD, HDD (Raid) being between them.

If we now look at the write performance, we see that the 1 Gbit Ethernet network strongly affects the performance of every storage devices, being significantly lower than their transfer rates. The throughput is a lot better with the 20 Gbit Ethernet network for all storage devices. The SSD gives better performance than HDD, especially when the size of file increases.

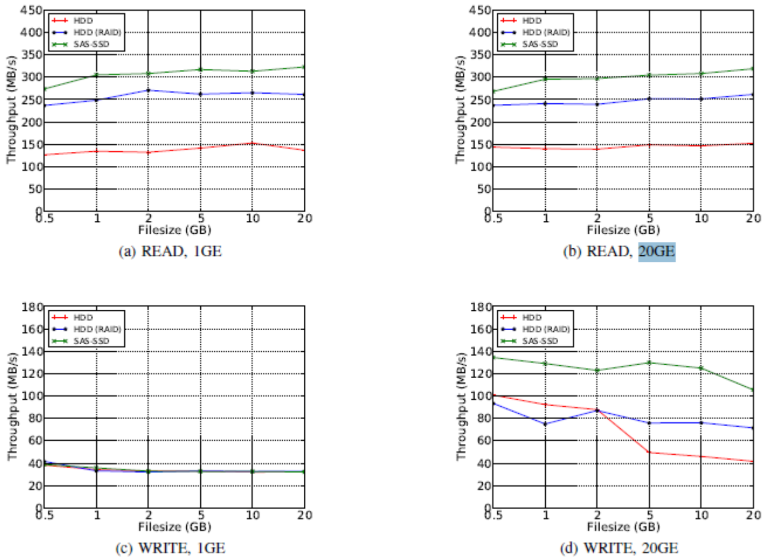

Figure 3.2 DFSIO Throughput vs Network Link Bandwidth and Storage Device Source: [12]

- DFSIO execution time

This DFSIO test in Figure 2.3 shows the I/O throughput and the I/O utilization given different numbers of data node read tasks.

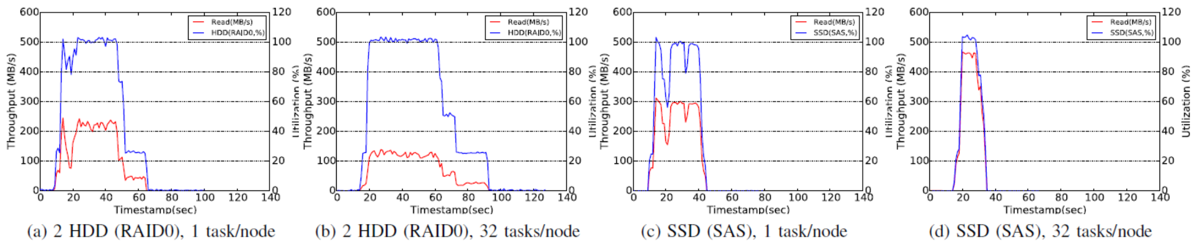

Figure 3.3 Multiple Task I/O Utilization (READ) Source: [12]

We note here that SSDs give better performance than HDDs even in the event of 1 task/node. However, SSDs become particularly interesting vs HDDs when the number of tasks per node increases: as expected, SSDs perform very well at random reads whereas HDDs are affected by their rotational latencies.

- Cost Effectiveness

Now that we know that we can greatly increase the shuffle phase execution time by using SSDs or more DRAM, we just have to add the Cost variable to find what is the best cluster configuration.

The figure 4 tells us that the best option is to use SSD as intermediate data storage in term of efficiency. An alternative is to stay with HDDs and increase the amount of DRAM. Of course, if we are looking at the performance only with no economical constraint, replacing all HDDs by SSDs is the solution you want to put in place.

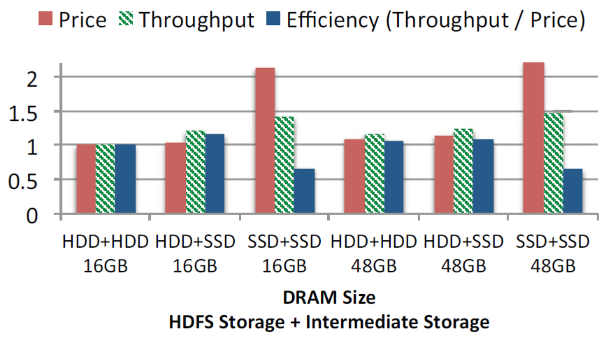

Figure 3.4 Cost effectives Source: [12]

4. Benchmarking Hadoop Big Data System

4.1 What is a Benchmark?

In Big data, benchmarks are important tools which can be used to measure, evaluate and compare big data systems and architectures in terms of performance, cost effectiveness etc. It can involve running many programs or perform operations to obtain performance of a system.

As a benchmark suite, we can say below two requirements are essential [13].

- Number of benchmarks should be as small as possible- This will ensure suite takes lesser time to evaluate a system. Benchmark itself shouldn't take more time.
- Benchmarks to be as diverse as possible- Multiple programs not just Word Count or Sort.

4.2 Key Qualities to be considered while benchmarking Big Data

Big data depends on below key qualities and it is called 4V [14].

Volume - Amount of data big data to handle. May be Petabytes. While some researchers started to design benchmarks for big data but with artificial data sets. So, it was unclear whether these benchmarks can precisely evaluate big data system performance. Understanding is that only real data can tell a real system behavior and work load characteristics and hence real-world data is preferred in big data benchmarks.

Variety - Capability of big data to process variety of data- structured, semi-structured or unstructured data and different sources like graph data.

Velocity - Ability of a big data system to stay unaffected through daily refreshes, referred to as extraction, transformation and load (ETL).

Veracity - This is added by IBM data scientists. It means that raw data characteristics must be preserved in processing or synthesizing the data.

4.3 Different Big Data Benchmarking tools:

Below are the different big data benchmarking tools we have come across during the survey and our focus is more on Hadoop benchmarks.

- BigDataBench [13], [14]:

This suite is part of ITC Bench which is presented by Institute of Computing Technology. BigDataBench was mainly for large-scale systems and architecture researches and also for big data applications characterizing. Each benchmark in BigDataBench is equal to a single big application.

- GridMix [16]

Specially designed for Apache Hadoop MapReduce, which includes only micro benchmarks (Sort, WordCount etc.)

for text data. It is always dominated by large data sort job. Unfortunately, it cannot properly evaluate the Hadoop framework due to the limitations in their representativeness and diversity.

- TeraSort or GraySort [15]

These are considered as micro benchmark. It sorts a large number of 100-byte records doing considerable amount of computation, networking, and storage I/O. This is limited to measuring only a specific property of the Hadoop. These cannot be used where complex computations take place for Example-Machine Learning.

- Cloudsuite

It supports a variety of data sources and types which are only designed to test applications running in cloud architecture, DBMS and MapReduce Hadoop [17]. Latest release is based on real world software stacks and represent real world setups.

- YCSB [14]

It is developed by Yahoo! as a cloud serving benchmark to evaluate NoSQL data stores [16].
It is a flexible benchmark and can be used by multiple users. It has two tiers. A performance tier (testing latency) and a scalability tier. YCSB is mainly for simple online service workloads (Cloud OLTP).

- TPC-DS [15]

TPC-DS is TPC's latest decision support benchmark. It covers below three major disciplines in the life-cycle of a relational decision support benchmark.

i) loading the initial database
ii) executing queries in both single and multi-user modes
iii) refreshing the database

TPC-DS handles some aspects of big data like volume and velocity. But lacks key components of big data like semi-structured and unstructured data and their associated analytics.

- BigBench [15]

It is the recent effort towards designing big data benchmarks.
BigBench focuses on big data offline analytics, thus adopting TPC-DS as the basis and adding a top new data types like semi-/un-structured data, as well as non-relational workloads. Although it has a complete coverage of data types, its object under test is DBMS and MapReduce systems leading to partial coverage of software stacks.

- HiBench [16]

It was proposed by Intel Company. It is a new, realistic and comprehensive benchmark suite for Hadoop which is must to properly evaluate and characterize the Hadoop framework. It covers incomplete data types and software stacks. It consists of a set of Hadoop programs including both read-world applications (large scale Data-Machine Learning Bayesian, K-means), Page Rank and synthetic mircobenchmarks.

- MRBench [18]

MRBench are based on TPC-H already audited benchmark which focuses on processing business oriented queries (22-ad-hoc complex queries) and concurrent data modifications. It allows users to configure environmental parameters (node count, the number of (Map/Reduce) tasks). Also, provides a flexibility to evaluate the performance of MapReduce systems while varying environmental parameters. Experiment concludes that MRBench can benchmark systems for jobs processing huge data and requiring relatively small processing power. So, it can be a possible approach to benchmark the MapReduce system. However, still requires many experiments and updates required.

5. Implementation issues in [12]

5.1 Evaluating Effectiveness of a New Architecture

The study by Moon et al. Sangwhan Moon [12] made use of Terasort benchmark results to demonstrate the performance improvement that SSD brought in their new architecture. However, the use of just one workload to conclude the improvement in effectiveness of a new architecture is a bit far from comprehensive.

From the perspectives of engineers and researchers, the improvement in performance and energy efficiency will be their primary concern. From commercial perspectives, price-performance effectiveness and the applicability to various application domains [19] [13] [[16]] [14] [15] will be their primary considerations. Thus, the effectiveness of new computer architecture should be evaluated across many dimensions and perspectives. Consequently, a series of experiments should be designed to obtain various measurements and metrics to describe the effectiveness of the new architecture from various perspectives quantitatively.

The following discussions will focus on performance improvement, although energy efficiency is also an important aspect to be studied. Hopefully, the assessment methodology employed for performance evaluation will be applicable to energy efficiency evaluation.

5.2 Applicability to Application Domain – Selection of Benchmark

The developments of big data systems and applications have been growing rapidly in recently years and will continue over the next decade. These systems cover many industrial and public service areas such as search engines, social networks, E-commerce sites and multimedia, as well as a variety of scientific research areas such as bioinformatics, environment, meteorology, and complex simulations of physics. Different application domains have different workload characteristics. Thus, the improvement that a SSD-enhanced architecture may bring will vary across domains. For example, sort workload is mostly I/O bound but wordcount workload is mostly CPU bound. The performance improvement brought by SSD to sort workload is likely to be more significant than that for wordcount. The applicability across application domains can be assessed using big data benchmark suites.

To produce meaningful evaluation results, the experiments must fulfill many requirements or considerations [19] [13] [[16]] [14] [15]. First, the workloads to be employed should be representative of a wide range of application domains (e.g. search engines, e-commerce, social networks, machine learning, bioinformatics, meteorology, etc.) without/with minimal redundancy. A spectrum of structured, semi-structured and un-structured data should be including. Varying volume of data input should be catered in the experiments as it has non-negligible impact on micro-architecture events [14]. There are a few big data benchmarking suites that support these requirements to different extend.

Big data benchmark aims to generate application-specific workloads and tests capable of processing big data with the 4V properties - volume, velocity, variety, and veracity [14][19]. Not all benchmark suites support all the 4V properties. Also, different benchmark suites may not have implementations for specific software stacks (Table 5.1 and Table 5.2 provide list of benchmarking suites and supported software stacks [19]). BigDataBench provides a comprehensive set of workloads for the evaluation of SSD-enhanced Hadoop MapReduce software stack [14].

Benchmark efforts	Workloads		Software stacks
	Type	Operations	
Sort [11]	Offline analytics	Sort	Hadoop
DFSIO [12]	Offline analytics	Generate, read, write, append, and remove data for MapReduce jobs	Hadoop
MRBench [13]	Online services	MapReduce jobs transformed from 22 TPC-H queries	Hadoop
GridMix [14]	Online services	Sort, sampling a large dataset	Hadoop
PigMix [15]	Online services	12 data queries	Hadoop
SWIM [16]	Offline analytics	Synthetic MapReduce jobs of reading, writing, shuffling and sorting data	Hadoop
HiBench [9]	Offline analytics	Sort, WordCount, TeraSort, PageRank, K-means, Bayes classification, Index	Hadoop and Hive
CALDA [17]	Online services	Load, scan, select, aggregate and join data, count URL links	Hadoop and DBMSs
AMPLab benchmark [18]	Online services	Part of CALDA workloads (scan, aggregate and join) and PageRank	Redshift, Hive, Shark, Impala and Tez
BigBench [10]	Online services	Database operations (select, create and drop tables)	Hadoop and DBMSs
	Offline analytics	K-means, classification	
LinkBench [19]	Online services	Database operations such as select, insert, update,and delete; association range queries and count queries	DBMS
TPC-DS [20]	Online services	Data loading, queries and maintenance	DBMS
BG benchmark [21]	Online services	Reading and updating databases	DBMS and NoSQL systems
YCSB [22]	Online services	OLTP (read, write, scan, update)	NoSQL systems
CloudSuite [23]	Online services	YCSB workloads	NoSQL systems, Hadoop, GraphLab
	Offline analytics	Text classification, WordCount	
BigDataBench [8]	Online services	Database operations (read, write, scan)	Hadoop, DBMSs, NoSQL systems, Hive, Impala, Hbase, MPI, Shark, Libc, and other real-time analytics systems
	Offline analytics	1. *Micro Benchmarks* (sort, grep, WordCount, CFS); 2. *Search engine workloads* (index, PageRank); 3. *Social network workloads* (connected components (CC), K-means and BFS); 4. *E-commerce site workloads* (Relational database queries (select, aggregate and join), collaborative filtering (CF) and Naive Bayes; 5. *Multimedia analytics workloads* (BasicMPEG, SIFT, DBN, Speech Recognition, Ray Tracing, Image Segmentation, Face Detection); 6. *Bioinformatics workloads* (SAND and BLAST)	

Table 5.1. Comparison of implemented workloads and supported software stacks in existing big data benchmarks. [19]

Benchmark efforts	Data Volume	Data Velocity	Data Variety		Data Veracity
			Data types	Data sources	
Sort [11]	Scalable	Un-controllable	Unstructured data	Texts	Un-considered
DFSIO [12]	Scalable	Un-controllable	Unstructured data	Texts	Un-considered
MRBench [13]	Scalable	Un-controllable	Structured data	Tables	Un-considered
GridMix [14]	Scalable	Un-controllable	Unstructured data	Texts	Un-considered
PigMix [15]	Scalable	Un-controllable	Unstructured data	Texts	Un-considered
SWIM [16]	Scalable	Un-controllable	Unstructured data	Texts	Un-considered
Hibench [9]	Partially scalable	Un-controllable	Unstructured data	Texts	Un-considered
CALDA [17]	Scalable	Un-controllable	Structured and unstructured data	Tables, texts	Un-considered
AMPLab benchmark [18]	Scalable	Un-controllable	Structured and unstructured data	Tables, texts	Un-considered
BigBench [10]	Scalable	Semi-controllable	Structured, semi-structured and unstructured data	Tables, web logs and texts	Partially Considered
LinkBench [19]	Partially scalable	Semi-controllable	Semi-structured data	Graphs	Partially Considered
TPC-DS [20]	Scalable	Semi-controllable	Structured data	Tables	Partially Considered
BG benchmark [21]	Scalable	Semi-controllable	Structured data	Tables	Un-considered
YCSB [22]	Scalable	Un-controllable	Structured data	Tables	Un-considered
CloudSuite [23]	Partially scalable	Semi-controllable	Structured, semi-structured and unstructured data	Tables, resumes, graphs and texts	Partially Considered
BigDataBench [8]	Scalable	Semi-controllable	Structured, semi-structured and unstructured data	Tables, resumes, graphs, texts, images videos and audios	Partially considered

Table 5.2. Comparison of data generation techniques in existing big data benchmarks [19]

5.3 Considerations on Testing Scenario Design

System performance tuning is always a complicated analysis work. Under different testing scenarios (can be different workload or different system configuration), different performance bottlenecks may be observed at different part of the system. For example, Moon, S. [12] manages in pointing out that sufficient network bandwidth is required to realize performance improvements by using SSDs in storing intermediate map outputs.

Wang, L [14] also points out the size of input data volume will have a direct on micro-architecture behaviors. While user-perceivable metrics provided by benchmark will tell the relative goodness of different architecture, the architectural metrics will help engineers in the identification of bottlenecks.

Thus, performance improvement measurements should be taken across a few dimensions including:

- various SSD-based configurations such as varying memory, network bandwidth, etc.
- various types of workload (provided by benchmark)
- range of input data volume (supported by benchmark)

The improvement measurements should be expressed in terms of ranges of values across these dimensions.

5.4 Is Performance Improvement Result Convincing?

Moon, S. [12] demonstrated a use case of SSD where a 5% increase in cost for SSD which in turn brought around 15% increase in performance. Is further improvement possible? Is a theoretical maximum available for reference? Moon, S. [12] used an all-SSD configuration, which cost-performance value was quite poor, cannot not serve such evaluation. The scale-out and scale-up approaches may be used as reference – if a new architecture brings better cost-performance than both scale-out and scale-up, it is already a good design worth consideration for mass production.

6. Conclusions

We study the performance characteristics of SSD and HDD. The most superior feature of SSD over HDD is its outstanding random I/O performance. [12] demonstrates the use of SSD to store intermediate data to boost the random I/O operations during the shuffle stage in the Hadoop MapReduce framework. But [12] only employs Terasort and DFSIO in obtaining performance measurement which is far from comprehensive and make the observations less convincing. We study big data benchmark tools in an attempt to address this deficiency in experiment design and measurement.

Hardly anyone will provide you real life application data, which is likely the most valuable asset of a company, to evaluate new big data system architecture. Big data benchmark tool has data generators to provide experimental data of various type of workload. The design of a good big data benchmark tool should have taken into account the 4V properties [14][19]. The Bigdatabench may be the best benchmark tool for evaluating new SSD-enriched Hadoop implementations.

7. Reference

[1] PriceG2 Research Report. When Will SSD Have Same Price as HDD. http://www.priceg2.com/.

[2] White T. Hadoop: The definitive guide[M]. " O'Reilly Media, Inc.", 2012.

[3] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113.

[4] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system[C]//Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on. IEEE, 2010: 1-10.

[5] Apache Hadoop Project, http://hadoop.apache.org

[6], Venkatachar P. Boost Oracle Data Warehouse Performance Using SanDisk Solid State Drives (SSDs)[J]. 2014.

[7] data source: https://en.wikipedia.org/wiki/Hard_disk_drive_performance_characteristics

[8] data source: https://en.wikipedia.org/wiki/Solid-state_drive

[9] Kambatla K, Chen Y. The Truth About MapReduce Performance on SSDs[C]//LISA. 2014: 109-118.

[10] Graham, Debra H (Intel), (May 31, 2016). " Optimize Hadoop Cluster Performance with Various Storage Media “. Intel[online]. Available: https://software.intel.com/en-us/articles/optimize-hadoop-cluster-performance-with-various-storage-media. [Accessed: Oct. 15, 2017].

[11]https://www.intel.com/content/www/us/en/homepage.html?_ga=2.22741746.89033942.1512553667-1615209752.1512553667

[12] Sangwhan Moon, Jaehwan Lee, Yang-suk Kee. 2014 IEEE International Conference on Cloud Coputing. “Introducing SSDs to the Hadoop MapReduce Framework”.

[13] Xiong W, Yu Z, Bei Z, et al. A characterization of big data benchmarks[C]//Big Data, 2013 IEEE International Conference on. IEEE, 2013: 118-125.

[14] Wang L, Zhan J, Luo C, et al. Bigdatabench: A big data benchmark suite from internet services[C]//High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on. IEEE, 2014: 488-499.

[15] Ghazal A, Rabl T, Hu M, et al. BigBench: towards an industry standard benchmark for big data analytics[C]//Proceedings of the 2013 ACM SIGMOD international conference on Management of data. ACM, 2013: 1197-1208.

[16] Huang S, Huang J, Dai J, et al. The HiBench benchmark suite: Characterization of the MapReduce-based data analysis[C]//Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on. IEEE, 2010: 41-51.

[17] Han R, Lu X, Xu J. On big data benchmarking[C]//Workshop on Big Data Benchmarks, Performance Optimization, and Emerging Hardware. Springer, Cham, 2014: 3-18.

[18] Kim K, Jeon K, Han H, et al. Mrbench: A benchmark for mapreduce framework[C]//Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on. IEEE, 2008: 11-18.

[19] Han, R., Jia, Z., Gao, W., Tian, X., & Wang, L. (2015). Benchmarking Big Data Systems: State-of-the-Art and Future Directions. arXiv preprint arXiv:1506.01494.

8. Individual Contribution Statement

Generally speaking, the team members contribute to this report evenly. Vincent Olivier BROT works for the chapter 3; Sik Hee FUNG works for the chapter 5 and chapter 6; Pavitra NAIK works for chapter 4; Wen YANG works for chapter 2. Also, every member has contributed to the proposal presentation and final presentation. We work closely and help each other. We believe that it is a very good team work. The individual contribution statements by each member are listed in Table 8.1.

NAME	STATEMENT
Vincent, Olivier BROT	I worked on the first part of the project around characteristics of SSDs and HDDS, and was responsible for chapter 3 “An Implementation Use Case”. As my previous studies and my career were oriented towards Business Intelligence, I had a very poor knowledge of everything hardware and performance related. I was happy to be part of this project and papers like [10] and [12] - as well as listening to my teammates and read their work - made me understand a lot about how they function. As I’m eager to work with big data after this master, I’m sure I will be able to leverage this project to better handle the performance issues I’m likely going to face. Thanks team mates ! **Contribution ratio: 25%**
Sik Hee FUNG	The first paper I read in this project was [12]. From [12], I helped formulating the framework of the case study. I was responsible for section 5 “Implementation Issues in [21]”. In the study, I read [12] to [19] and a few texts on Hadoop. Through the readings, I deepen my understanding on the Hadoop MapReduce framework. The reading on big data benchmark was interesting. I got most of the insights on experiment design and measurements from those benchmark papers. I like the case study topic cause in my day to day job, system performance is always one of the issues I need to address. I find big data benchmark is a very interesting research area too. **Contribution ratio: 25%**
Pavitra NAIK	During initial days of this project, I got an understanding on difference between SSD and HDD [7], [8]. Of late, what made me curious was different big data benchmarking tools. So, I focused on papers- [13], [14], [15], [16]. Studied 4V properties of big data and essential requirements of benchmarking. Then, learnt different big data benchmarking tools and their usage. Have had an overview on how researchers design a benchmark and what are the challenges researchers face while designing a benchmark. Even I was able to compare few of these benchmarking tools and figure out differences, benefits and drawbacks. I must say this learning has quite motivated me to explore further, various big data technologies. Over all it has been a very good learning experience for me and my team mates for sure. **Contribution ratio: 25%**
Wen YANG	I work for the chapter 2—SSD versus HDD, I read several papers to get a full understand about: the difference SSD and HDD, the storage usage in MapReduce HDFS and the implementation suggestion from SSD manufactures. The paper and resources I have studied including [1], [2], [3], [4], [5], [6], [7] ,[8], [9] ,[10] and [11]. From this project, I have learned that: the different characters of SSD and HDD, the detail mechanism and workflow of MapReduce and the limitations of using SSD in MapReduce. Although reading papers is time-consuming, it is worthy. **Contribution ratio: 25%**

Table 8.1 individual contribution statement