I/O Characterization of Big Data Workloads in Data Centers

Fengfeng Pan[1 2] Yinliang Yue[1] Jin Xiong[1] Daxiang Hao[1]

[1]Research Center of Advanced Computer Systems, Institute of Computing Technology, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences, Beijing, China
panfengfeng@ncic.ac.cn {yueyinliang, xiongjin, haodaxiang}@ict.ac.cn

Abstract

As the amount of data explodes rapidly, more and more organizations tend to use data centers to make effective decisions and gain a competitive edge. Big data applications have gradually dominated the data centers' workloads, and hence it has been increasingly important to understand their behaviour in order to further improve the performance of data centers. Due to the constantly increased gap between I/O devices and CPUs, I/O performance dominates the overall system performance, so characterizing I/O behaviour of big data workloads is important and imperative.

In this paper, we select four typical big data workloads in broader areas from the BigDataBench which is a big data bench-mark suite from internet services. They are Aggregation, TeraSort, K-means and PageRank. We conduct detailed deep analysis of their I/O characteristics, including disk read/write bandwidth, I/O devices' utilization, average waiting time of I/O requests, and average size of I/O requests, which act as a guide to design high-performance, low-power and cost-aware big data storage systems.

Keywords I/O characterization; Big Data Workloads; Data Centers;

1. Introduction

In recent years, big data workloads [25] are more and more popular and play an important role in enterprises' business. There are some popular and typical applications, such as TeraSort, SQL operations, PageRank and K-means. Specifically, TeraSort is widely used for page or document ranking; SQL operations, such as join, aggregation and select, are used for log analysis and information extraction; PageRank is widely used in search engine field; and K-means is usually used as electronic commerce algorithm.

These big data workloads run in data centers, and their performance is critical. The factors which affect their performance include: algorithms, hardware including node, interconnection and storage, and software such as programming model and file systems. This is the reason for why several efforts have been made to analyse the impact of these factors on the systems [25][19][22]. However, data access to persistent storage usually accounts for a large part of application time because of the ever-increasing performance gap between CPU and I/O devices.

BPOE-4, March 1, 2014,Salt Lake City, Utah, USA.

With the rapid growth of data volume in many enterprises and a strong desire for processing and storing data efficiently, a new generation of big data storage system is urgently required. In order to achieve this goal, a deep understanding of big data workloads present in data centers is necessary to guide the big data system's design and tuning.

Some studies have been conducted to explore the computing characteristics of big data workloads [21][17], meanwhile, lots of work also have been done to depict the storage I/O characteristics of enterprise storages [5][8]. However, to the best of our knowledge, none of the existing research has understood the I/O characteristics of big data workloads, which is much more important in the current big data era. So understanding the characteristics of these applications is the key to better design the storage system and optimize their performance and energy efficiency.

The main metrics used for the I/O characterization in this paper are listed below.

First, *disk read or write bandwidth (MB/s)*: which indicates the amount of data read from or write to the storage systems per unit time. Second, *I/O devices' utilization (percent)*: which shows percentage of CPU time during which I/O requests were issued to the device. Third, *average waiting time of I/O requests (seconds)*: the time spent in queue. Fourth, *average size of I/O requests (the number of sectors, and the size of sector is 512 bytes)*: the average size of the re-quests that were issued to the device.

In this paper, we choose four typical big data workloads as mentioned above, because they have been widely used in popular application domains [25][1], such as search engine, social networks and electronic commerce. Detailed information about I/O metrics and workloads are shown in Section 3.2. Through the detailed analysis, we get the following four observations:

- The change of the number of task slots has no effects on the four I/O metrics, but increasing the number of task slots appropriately can accelerate the process of application execution;
- Increasing memory can reduce the number of I/O requests, alleviate the pressure of disk read/write, and effectively improve the I/O performance when the data size is large;
- The compression of intermediate data mainly affects the MapReduce I/O performance and has little influence on HDFS I/O performance. However, compression consumes some CPU resource which may influence the job's execution time;
- The I/O pattern of HDFS and MapReduce are different, namely, HDFS's I/O pattern is large sequential access and MapReduce's I/O pattern is small random access, so when configuring storage systems, we should take several factors into account, such as the number of devices and the types of devices.

The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 describes the experimental methodology.

Section 4 shows the experimental results. Section 5 briefly brings up future work and concludes this paper.

2. Related Work

Workloads characterization studies play a significant role in detecting problems and performance bottlenecks of systems. Many workloads have been extensively studied in the past, including enterprise storage systems [10][7][24], web server [16][23], HPC cluster [15] and network systems [13].

2.1 I/O Characteristics of Storage workloads

There have been a number of papers about the I/O characteristics of storage workloads [16][12][18].

Kozyrakis et al. [18] presented a system-wide characterization of large-scale online services provided by Microsoft and showed the implications of such workloads on data-center server design. Kavalanekar et al. [16] characterized large online services for storage system configuration and performance modeling. It contains a set of characteristics, including block-level statistics, multi-parameter distributions and rankings of file access frequencies. Similarly, Delemitrou et al. [12] presented a concise statistical model which accurately captures the I/O access pattern of large-scale applications including their spatial locality, inter-arrival times and type of accesses.

2.2 Characteristics of big data workloads

Big data workloads have been studied in recent years at various levels, such as job characterization [21][17][9], storage sytems [5][8].

Kavulya et al. [17] characterized resource utilization patterns, job patterns, and source of failure. This work focused on predicting job completion time and found the performance problems. Similarly, Ren et al. [21] focused on not only job characterization and task characterization, but also resource utilization on a Hadoop cluster, including CPU, Disk and Network.

However, the above researches on big data workloads focused on job level, but not on the storage level. Some studies have provided us with some metrics about data access pattern in MapReduce scenarios [6][14][4], bur these metrics are limited, such as block age at time of access [14] and file popularity [6][4]. Abad et al. [5] conducted a detailed analysis about some HDFS characterization, such as file popularity, temporal locality, request arrival patterns, and then figure out the data access pattern of two types of big data workloads, namely batch and interactive query workloads. But this work concentrates on HDFS's I/O characterization, does not study the intermediate data, and also this work only involves two types of workloads.

In this paper, we focus on I/O characteristics of big data workloads, the difference between our work and the previous work is:

First, we focus on the I/O behaviour of individual workload and choose four typical workloads in broader areas, including search engine, social network, e-commerce, which are popular in the current big data era;

Second, we analyze the I/O behavior of both HDFS and MapReduce intermediate data from different I/O characteristics, including disk read/write bandwidth, I/O devices' utilization, average waiting time of I/O requests, and average size of I/O requests.

3. Experimental Methodology

This section firstly describes our experiment platform, and then presents workloads used in this paper.

3.1 Platform

We use an 11-node (one master and ten slaves) Hadoop cluster to run the four typical big data workloads. The nodes in our Hadoop cluster are connected through 1 Gb ethernet network. Each node has two Intel Xeon E5645 (Westmere) processors, 32 GB memory and 7 disks(1TB). A Xeon E5645 processor includes six physical out-of-order cores with speculative pipelines. Table 1 and 2 shows the detailed configuration information.

Table 1: The detailed hardware configuration information

CPU Type	Intel R Xeon E5645
# Cores	6 cores@2.4G
# threads	12 threads
Memory	32 GB , DDR3
Disk	7 disks (one disk for system, three disks for HDFS data, the other three disks for MapReduce intermediate data), Disk Model Seagate: ST1000NM0011, Capacity: 1TB, Rotational Speed: 7200 RPM, Avg. Seek/Rotational Time: 8.5ms/4.2ms, Sustained Transfer Rate: 150MB/s

Table 2: The detailed software configuration information

Hadoop	1.0.4
JDK	1.6.0
Hive	0.11.0
OS Distribution and Linux kernel	Centos 5.5 with the 2.6.18 Linux kernel
TeraSort	BigDataBench2.1 [2]
SQL operations	BigDataBench2.1
PageRank	BigDataBench2.1
K-means	BigDataBench2.1

3.2 Workloads and Statistics

We choose four popular and typical big data workloads from BigDataBench [25]. BigDataBench is a big data benchmark suite from internet services and it provides several big data generation tools to generate various types and volumes of big data from small-scale real-world data while preserving their characteristics. Table 3 shows the description of the workloads, which is characterized in this paper.

- **Micro Benchmarks: TeraSort**
  TeraSort is a standard benchmark created by Jim Gray. In this paper, we use the data generation tools of BigDataBench2.1 to generate the text data as its input data. According to a previous study [11], TeraSort is an I/O-bound workload;
- **Data Analytics Benchmarks: Hive Query**
  The Hive Query Workload is developed based on the study [20]. It describes Hive queries (e.g. Aggregation and Join) performing the typical OLAP queries. Its input data is generated by the BigDataBench2.1's data generation tools. In this paper, we select the Aggregation operation which is a CPU-bound workload [11];
- **Machine Learning Benchmarks: K-means**
  K-means is a well-known clustering algorithm for knowledge discovery and data mining. It has two stages, namely iteration (CPU-bound) [11] and clustering(I/O-bound) [11]. We use the

graph data which is provided by the BigDataBench2.1 as K-means' input data;

- **Web Search Benchmarks: PageRank**
  PageRank is a link analysis algorithm and it assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of "measuring" its relative importance within the set. BigDataBench also has an implementation of PageRank on Hadoop which uses Google web graph data to generate PageRank's input data. PageRank on Hadoop is a CPU-bound workload [11];

Table 3: The description of the workloads

Workloads	Performance bottleneck	Scenarios	Input Data size
TeraSort (TS)	I/O bound	Page ranking; document ranking	1TB
Aggregation (AGG)	CPU bound	Log analysis; information extraction	1TB
K-means (KM)	CPU bound in iteration; I/O bound in clustering	Clustering and Classification	512GB
PageRank (PR)	CPU-bound	Search engine	512GB

Table 4: Notation of I/O characterization

I/O characterization	Description	Notes
rMB/s and wMB/s	The number of megabytes read from or written to the device per second	Disk Read or Write Bandwitdh
%util	Percentage of CPU time during which I/O requests were issued to the device	Disk utilization
await (ms)	The average time for I/O requests issued to the device to be served. This includes the time spent by the requests in queue and the time spent servicing them	average waiting time of I/O request = await - svctm
svctm (ms)	The average service time for I/O requests that were issued to the device	
avgrq-sz (the number of sectors)	The average size of the requests that were issued to the device. And the size of sectors is 512B	average size of I/O request

Iostat [3] is a well-used monitor tool used to collect and show various system statistics, such as CPU times, memory usage, as well as disk I/O statistics. In this paper, we mainly focus on the disk-level I/O behaviour of the workloads, and we extract information from iostat's report, and the metrics which we focus on are shown in Table 4.

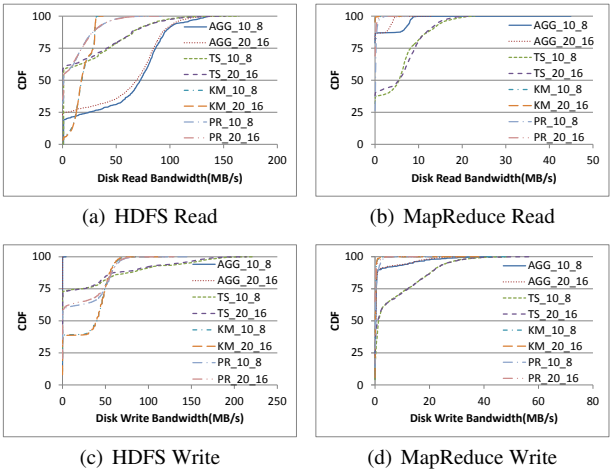
4. Results

In this section, we describe HDFS/MapReduce I/O characteristics from four metrics, namely, disk read/write bandwidth, I/O devices' utilization, average waiting time of the I/O requests, and average size of the I/O requests. Through the comparison of each workloads, we can obtain the I/O characterization of HDFS and MapReduce respectively, and also the difference between HDFS and MapReudce.

In addition, different Hadoop configurations can influence the workloads' execution. So in this paper, we select three factors and analyse their impact on the I/O characterization of these workloads. The three factors are as follows. First, *the number of task slots, including map slots and reduce slots*. In Hadoop, computing resource is represented by slot, there are two types of slot: map task slot and reduce task slot. Here computing resource refers to CPU. Second, *the amount of physical memory of each node*. As we know that memory plays an important role in I/O performance, so memory is an important factor which affects the I/O behaviour of workloads. So, the second factor we focus on is the relationship be-tween memory and I/O characterization. Third, *whether the intermediate data is compressed or not*. Compression involves two types of resources: CPU and I/O. what's the influence on the I/O behaviour of workloads when CPU and I/O resources both change. So, the final factor we focus on is the relationship between compression and I/O characterization.

4.1 Disk Read/Write Bandwidth

4.1.1 Task slots



(a) HDFS Read (b) MapReduce Read

(c) HDFS Write (d) MapReduce Write

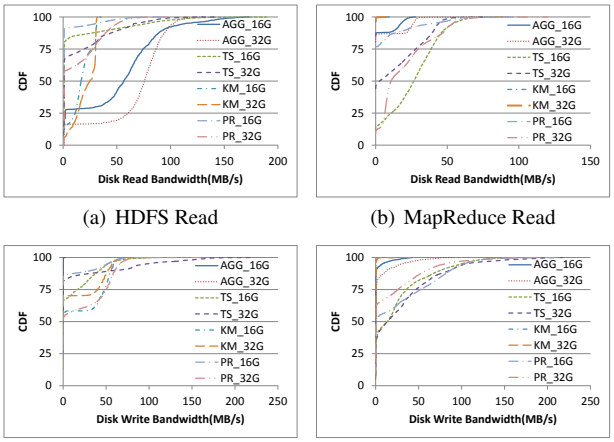
Figure 1: The effects of the number of task slots on the Disk Read/Write Bandwidth in HDFS and MapReduce respectively

Figure 1 shows the effects of the number of task slots on the disk read/write bandwidth in HDFS and MapReduce respectively. In these experiments, each node configured 16GB memory and the intermediate data is compressed. "10_8", "20_16" in the figures mean the number of map task slots and reduce task slots respectively.

From Figure 1 we can get the following two conclusions. First, when the number of task slots changes, there is barely any influence on the disk read/write bandwidth in both scenarios for every workload. Second, the variation of disk read/write bandwidth of different workloads in both scenarios are disparate because the data volume of each workload in different phases of execution are not the same.

Table 5: The Peak HDFS Disk Read Bandwidth of these workloads

Workloads	Configuration	Peak value (MB/s)
Aggregation	AGG_10_8	138.28
	AGG_20_16	141.19
TeraSort	TS_10_8	164.26
	TS_20_16	151.28
K-means	KM_10_8	31.88
	KM_20_16	37.69
PageRank	PR_10_8	91.7
	PR_20_16	95.13



(a) HDFS Read (b) MapReduce Read

(c) HDFS Write (d) MapReduce Write

Figure 2: The effects of memory on the Disk Read/Write Bandwidth in HDFS and MapReduce respectively

This observation is also applied to the peak disk read/write bandwidth of these workloads in HDFS and MapReduce. Table 5 shows the peak HDFS disk read bandwidth of these workloads.

In a word, there is little effect on disk read/write bandwidth when the number of task slots changes. However, configuring the number of task slots appropriately can reduce the execution time of workloads, so we should take it into account when workloads run.

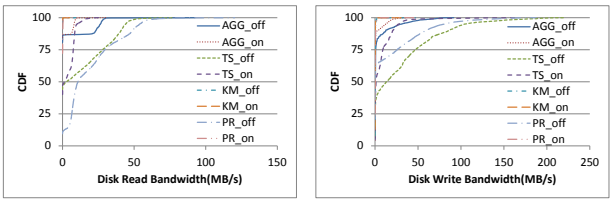
4.1.2 Memory

Figure 2 displays the effects of the memory size on the disk read/write bandwidth in HDFS and MapReduce respectively. In these experiments, task slots configuration on each node is 10_8 and the intermediate data is not compressed. "16G", "32G" in the figures mean the memory size of node.

As Figure 2 shows, the influence of the memory size on disk read/write Bandwidth depends on the data volume. There is a growth of disk read bandwidth in HDFS when memory increases as shown in figure 2(a) due to the large amount of raw data, but after handling and processing the raw data, the disk write bandwidth of each workloads in HDFS are different because of the final data volume. When the final data volume is small, memory has no effects on the disk write bandwidth, such as K-means, as shown in figure 2(c). This result can also be reflected in MapReduce.

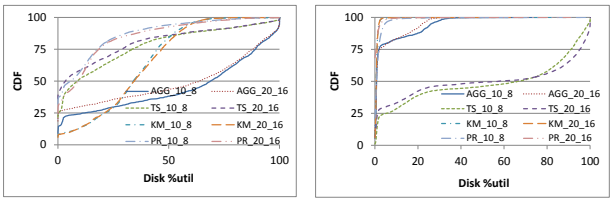
4.1.3 Compression

Figure 3 exhibits the effects of intermediate data compression on the disk read/write bandwidth in MapReduce. In these experiments, each node configured 32GB memory and task slots configuration is 10_8. "off", "on" in the figures mean whether the intermediate data is compressed or not.



(a) MapReduce Read (b) MapReduce Write

Figure 3: The effects of compression on the Disk Read/Write Bandwidth in MapReduce.



(a) HDFS (b) MapReduce

Figure 4: The effects of the number of task slots on the Disk Utilization in HDFS and MapReduce respectively.

Table 6: the HDFS Disk %util ratio

	>90%uitl	>95%uitl	>99%util
AGG	22.6%	16.4%	9.8%
TS	5.2%	3.8%	2.4%
KM	0.4%	0.3%	0.2%
PR	0.5%	0.3%	0.2%

As Figure 3 shows, due to the reduction of intermediate data volume with compression, the disk read/write bandwidth increase in MapReduce.

In addition, compression has little impact on the HDFS's disk read/write bandwidth, so we do not present the result.

4.2 Disk Utilization

4.2.1 Task slots

Figure 4 depicts the effects of the number of task slots on the disk utilization in HDFS and MapReduce respectively.

From Figure 4 we can get the following two conclusions. First, the trends of workloads in disk utilization are the same when the number of task slots changes, i.e. the number of task slots has little impact on the disk utilization in both scenarios. Second, workloads have different behaviour about disk utilization in both scenarios. From the Table 6 and Table 7, the HDFS disk utilization of Aggregation is higher than the others, so Aggregation HDFS disk may be the bottleneck. Similarly, the MapReduce disk utilization of TeraSort is higher than the others; From the figure 4(b), the MapReduce disk utilization of workloads is 50% or less at their most of execution time, so the disks are not busy, except TeraSort because of the large amount of TeraSort's intermediate data.

4.2.2 Memory

Figure 5 depicts the effects of the memory size on the disk utilization in HDFS and MapReduce respectively. As Figure 5(a)

Table 7: the MapReduce Disk %util ratio

	>90%uitl	>95%uitl	>99%util
AGG	0	0	0
TS	27.2%	15.6%	5.5%
KM	0	0	0
PR	0.1%	0.1%	0.1%

shows, increasing memory size has no impact on the disk utilization in HDFS. However, from Figure 5(b) we can see that there is a difference between HDFS and MapReduce. In MapReduce, the disk utilization of Aggregation and K-means has no changes when memory size changes because the devices are not busy before memory changes. However, the disk utilization of TeraSort and PageRank is reduced when memory increases as shown in figure 5(b). So, increasing memory size can help reduce the number of I/O requests and ease the bottleneck of disk.

4.2.3 Compression

From Figure 6(a), we can know that when intermediate data is compressed, the HDFS's disk utilization of TeraSort and Aggregation is high, except the other two workloads. As Figure 6(b) shows, there is no influence on intermediate data's disk utilization of TeraSort, Aggregation and K-means, except PageRank.

4.3 Average waiting time of I/O request

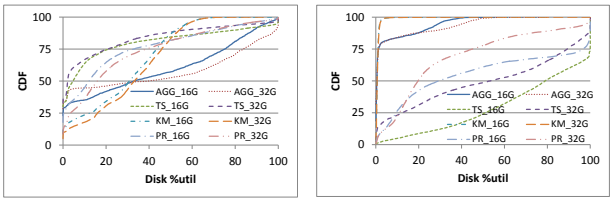
4.3.1 Task slots

Figure 7 shows the effects of the number of task slots on the disk waiting time of I/O requests in HDFS and MapReduce respectively.

As Figure shows, when the number of task slots increases, the disk average waiting time of I/O requests does not change.

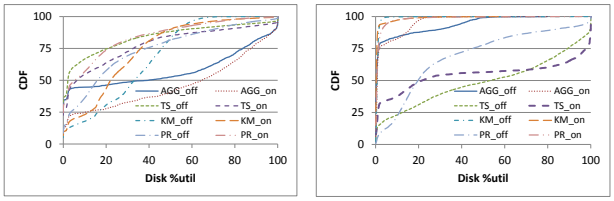
4.3.2 Memory

Figure 8 shows the effects of Memory on the disk waiting time of I/O requests in HDFS and MapReduce respectively.



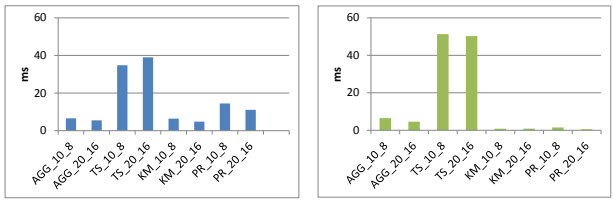
(a) HDFS (b) MapReduce

Figure 5: The effects of memory on the Disk Utilization in HDFS and MapReduce respectively.



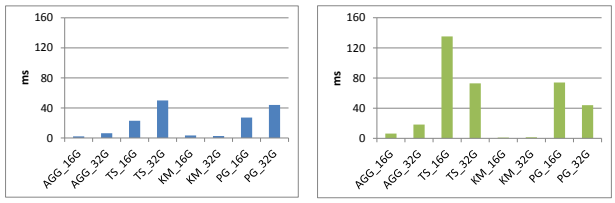
(a) HDFS (b) MapReduce

Figure 6: The effects of compression on the Disk Utilization in HDFS and MapReduce respectively.



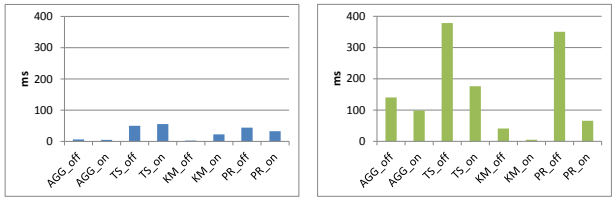
(a) HDFS (b) MapReduce

Figure 7: The effects of the number of task slots on the Disk waiting time of I/O request in HDFS and MapReduce respectively.



(a) HDFS (b) MapReduce

Figure 8: The effects of memory on the Disk waiting time of I/O requests in HDFS and MapReduce respectively.



(a) HDFS (b) MapReduce

Figure 9: The effects of compression on the Disk waiting time of I/O requests in HDFS and MapReduce respectively.

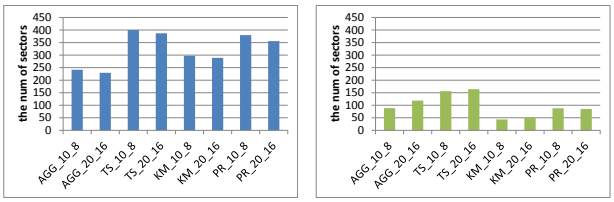
From Figure 8, we can learn that the disk average waiting time of I/O requests of workloads varies with different memory size, in other words, the memory size has an impact on the disk waiting time of I/O requests and the MapReduce disk waiting time of I/O request is larger than the HDFS's.

4.3.3 Compression

Figure 9 depicts the effects of compression on the disk waiting time of I/O requests in HDFS and MapReduce respectively.

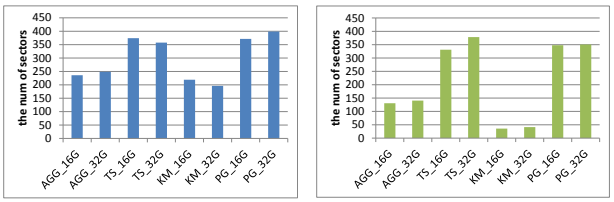
From Figure 9, we can see that the disk average waiting time of I/O requests remains unchanged in HDFS because HDFS's data is not compressed, however, due to the reduction of intermediate data volume with compression, the disk waiting time of I/O request in MapReduce is decreased.

And the MapReduce disk waiting time is larger than the HDFS's because of their different I/O mode in access pattern, i.e. HDFS's access pattern is domated by large sequential accesses, while MapReduce's access pattern is dominated by smaller random access. This result can be seen in Figure 10.



(a) HDFS (b) MapReduce

Figure 10: The effects of the number of task slots on the Disk average size of I/O request in HDFS and MapReduce respectively.



(a) HDFS (b) MapReduce

Figure 11: The effects of memory on the Disk average size of I/O request in HDFS and MapReduce respectively.

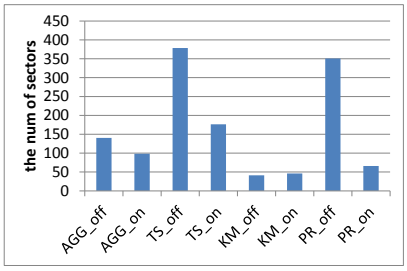


Figure 12: The effects of compression on the Disk average size of I/O request in MapReduce.

4.4 Average size of I/O requests

4.4.1 Task slots

Figure 10 depicts the effects of the number of task slots on the disk average size of I/O request in HDFS and MapReduce respectively.

As the task slots is a kind of computing resource, there is little impact on the disk average size of I/O requests when the number of task slots changes from the figures.

Also, the average size of HDFS I/O requests is larger than the MapReduce's because they have different I/O mode in I/O granularity. In other words, HDFS's I/O granularity is larger than the MapReduce's.

The same result can be seen in Figure 11 which reflects the effects of memory on the disk average size of I/O requests. However, average size of I/O requests in figure 11(b) is larger than the figure 10(b) because of the influence of compression which is reflected in Figure 12.

4.4.2 Compression

Figure 12 displays the effects of compression on the disk average size of I/O requests in MapReduce.

It is seen from figure that as the intermediate data is compressed, the disk average size of I/O requests is decreased, and the percentage of reduction varies with the types of workloads due to the intermediate data volume. As Figure 12 shows, there is little influence on the disk average size of I/O request when the the intermediate data volume is small, such as Aggregation and K-means.

In addition, whether the intermediate data is compressed or not has no impact on the HDFS's disk average size of I/O requests, so we do not present the result.

5. Conclusion and Future work

In this paper, we have presented a study of I/O characterization of big data workloads. These workloads are typical, which are representative and common in search engine, social networks and electronic commerce. In contrast with previous work, we take into account disk read/write bandwidth, average waiting time of I/O requests, average size of I/O requests and storage device utilization, which are important for big data workloads. Some observations and implications are concluded: *1*) task slots has little effects on the four I/O metrics, but increasing the number of task slots can accelerate the process of application execution; *2*) increasing memory can reduce the number of I/O requests, alleviate the pressure of disk read/write, and effectively improve the I/O performance when the data size is large; *3*) the compression of intermediate data mainly affects the MapReduce I/O performance and has little influence on HDFS I/O performance. However, compression consumes some CPU resource which may influence the job's execution time; *4*) HDFS data and MapReduce intermediate data have different I/O mode, which leads us to configuring their own storage systems according to their I/O mode.

In the future, we plan to combine two factors—a low-level description of physical resource, including CPU, Memory and Disk, and the high-level functional composition of big data workloads to reveal the major source of I/O demand in these workloads.

Acknowledgments

This paper is supported by National Science Foundation of China under grants no. 61379042, 61303056, and 61202063, and Huawei Research Program YB2013090048. We would like to thank Zijian Ming and Professor Lei Wang for their help to run the benchmarks.

References

[1] *http://www.alexa.com/topsites/global*.

[2] *http://prof.ict.ac.cn/BigDataBench/*.

[3] *http://linux.die.net/man/1/iostat*.

[4] C. L. Abad, Y. Lu, and R. H. Campbell. Dare: Adaptive data replication for efficient cluster scheduling. *In CLUSTER, 2011*.

[5] C. L. Abad, N. Roberts, Y. Lu, and R. H. Campbell. A storage-centric analysis of mapreduce workloads: File popularity, temporal locality and arrival patterns. *In IISWC, 2012*.

[6] G. Ananthanarayanan, S. Agarwal, S. Kandula, A. Greenberg, I. Stoica, D. Harlan, and E. Harris. Scarlett: coping with skewed content popularity in mapreduce clusters. *In Proceedings of the sixth conference on Computer systems, 2011*.

[7] L. N. Bairavasundaram, A. C. Arpaci-Dusseau, R. H. Arpaci-Dusseau, G. R. Goodson, and B. Schroeder. An analysis of data corruption in the storage stack. *In ACM Transactions on Storage (TOS), 2008*.

[8] Y. Chen, S. Alspaugh, and R. Katz. Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads. *In VLDB, 2012*.

[9] Y. Chen, S. Alspaugh, and R. H. Katz. Design insights for mapreduce from diverse production workloads. 2012.

[10] Y. Chen, K. Srinivasan, G. Goodson, and R. Katz. Design implications for enterprise storage systems via multi-dimensional trace analysis. *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, 2011*.

[11] J. Dai. Performance, utilization and power characterization of hadoop clusters using hibench. *In Hadoop in China, 2010*.

[12] C. Delimitrou, S. Sankar, K. Vaid, and C. Kozyrakis. Decoupling datacenter studies from access to large-scale applications: A modeling approach for storage workloads. *In IISWC, 2011*.

[13] D. Ersoz, M. S. Yousif, and C. R. Das. Characterizing network traffic in a cluster-based, multi-tier data center. *In ICDCS, 2007*.

[14] B. Fan, W. Tantisiriroj, L. Xiao, and G. Gibson. Diskreduce: Raid for data-intensive scalable computing. *In Proceedings of the 4th Annual Workshop on Petascale Data Storage, 2009*.

[15] A. Iamnitchi, S. Doraimani, and G. Garzoglio. Workload characterization in a high-energy data grid and impact on resource management. *In CLUSTER, 2009*.

[16] S. Kavalanekar, B. Worthington, Q. Zhang, and V. Sharda. Characterization of storage workload traces from production windows servers. *In IISWC, 2008*.

[17] S. Kavulya, J. Tan, R. Gandhi, and P. Narasimhan. An analysis of traces from a production mapreduce cluster. *In CCGrid, 2010*.

[18] C. Kozyrakis, A. Kansal, S. Sankar, and K. Vaid. Server engineering insights for large-scale online services. *In IEEE micro, 2010*.

[19] A. Kyrola, G. Blelloch, and C. Guestrin. Graphchi: Large-scale graph computation on just a pc. *In OSDI, 2012*.

[20] A. Pavlo, E. Paulson, A. Rasin, D. J. Abadi, D. J. DeWitt, S. Madden, and M. Stonebraker. A comparison of approaches to large-scale data analysis. *In ACM SIGMOD, 2009*.

[21] Z. Ren, X. Xu, J. Wan, W. Shi, and M. Zhou. Workload characterization on a production hadoop cluster: A case study on taobao. *In IISWC, 2012*.

[22] D. S. Roselli, J. R. Lorch, T. E. Anderson, et al. A comparison of file system workloads. *In USENIX Annual Technical Conference, 2000*, pages 41–54, 2000.

[23] S. Sankar and K. Vaid. Storage characterization for unstructured data in online services applications. *In IISWC, 2009*.

[24] F. Wang, Q. Xin, B. Hong, S. A. Brandt, E. L. Miller, D. D. Long, and T. T. McLarty. File system workload analysis for large scale scientific computing applications. *In MSST, 2004*.

[25] L. Wang, J. Zhan, C. Luo, Y. Zhu, Q. Yang, Y. He, W. Gao, Z. Jia, Y. Shi, S. Zhang, et al. Bigdatabench: A big data benchmark suite from internet services. *In HPCA, 2014*.