Performance Evaluation of Scale-out NAS for HDFS

Makoto Sato
Kanagawa Institute of Technology
Japan
s1021079@cce.kanagawa-it.ac.jp

Hiroki Kanzaki
Graduate School of Kanagawa Institute of Technology
Japan
s1285025@cce.kanagawa-it.ac.jp

Shoji Kawata
Graduate School of Kanagawa Institute of Technology
Japan
s1285014@cce.kanagawa-it.ac.jp

Shun Sugiyama
Zuken NetWave, Inc.
Japan
shun.sugiyama@znw.co.jp

Shingo Otsuka
Kanagawa Institute of Technology
Graduate School of Kanagawa Institute of Technology
Japan
otsuka@ic.kanagawa-it.ac.jp

***Abstract*— Isilon of EMC Corporation, which is the major company of the storage production, announces the correspondence to Hadoop Distributed File System (HDFS). Hadoop is parallel distributed processing base for large-scale data constructed in HDFS and MapReduce. In addition, it can treat huge files using plural computers link in Hadoop. However, the tendency of detailed performance and various parameters is not known. Therefore, in this paper, we perform the comparison in the case with normal HDFS and Isilon for HDFS using a benchmark about writing performance and reading performance.**

Keywords-Hadoop; HDFS; Scale-out NAS; Isilon.

I. INTRODUCTION

Scale-out NAS (Network Attached Storage) has extensibility and the management characteristics that conventional scale-up NAS does not have. And it can expand capacity and the performance seamlessly. Therefore, we are able to be gradually expanded from small constitution as needed. Scale-out NAS manages the cluster as single file system, and it is not necessary to be conscious of the real physical position of data [1]. Isilon of EMC Company announces that it native supports Hadoop Distributed File System (HDFS) in latest OS (OneFS) [2].

Hadoop is parallel distributed processing base for the large-scale data constructed in HDFS and MapReduce. In addition, it can treat huge files by letting plural computers link in Hadoop. HDFS is constructed in Namenode and cluster of Datanode. Datanode manages the data in HDFS divided into the fixed length called the block [3] [4].

Namenode manages the file attribute information called metadata and the information of the file system. Namenode is a single obstacle point in HDFS. In one of the faults of HDFS, The file system becomes offline when Namenode falls. Isilon solves this problem of single obstacle points in HDFS.

In addition, HDFS (Hadoop) maintains a replica function for HDD trouble of Datanode, and the user can decide the number of the replicas depending on the use situation. If there is much number of the replicas, redundancy improves, but the processing capacity decreases because the access to an HDD increases. Generally, the number of the replicas of Hadoop is around three. In the case of Isilon for HDFS, it can handle the number of the replicas with one because Isilon secures redundancy. Therefore, we can make use of HDFS performance to the maximum.

This way, Isilon solves a part of the problems in HDFS and maintains high processing capacity. However, the tendency of detailed performance and various parameters is not known. Therefore, in this paper, we perform the comparison in the case with normal HDFS and Isilon for HDFS using a benchmark about writing performance and reading performance.

II. RELATED WORKS

There are some related work of this study as follows; Evaluation of Hadoop system consisting of Virtual Machines on Multi-core CPUs by Ishii [5], Studies on Evaluating Performance Efficiency of Distributed File Systems by Sakurai [6], Characterization of Remote Data Access for Hadoop Distributed File System over a long-latency environment by Momose [7], and Consideration on Ad hoc query processing with Adaptive Index in Map Reduce Environment by Okudera [8].

 ISBN: 978-1-61208-311-7

III. PARALLEL DISTRIBUTED PROCESSING

A. Hadoop

The Large-scale data are created in various situations by a technological change. The big data such as the GPS of the mobile devices, cameras and action histories of the users with the sensor continue increasing every day. The useful information is provided by analyzing it. Therefore, Hadoop attracts attention as the parallel distributed processing base which can process large-scale data easily.

Hadoop is constructed in HDFS and MapReduce and hBase [9,10]. The MapReduce processing performs important distributed processing. The data are managed by a combination of Key and Value. By the MapRecude processing, three next processing are carried out: Map processing to divide large-scale data into small data, and to extract necessary information, Shuffle processing to bundle up a combination having same Key, and Reduce processing to gather them up, and to output a result.

HDFS is file system to distribute large-scale data, and to manage using plural disks. HDFS improves throughput by being parallel from plural disks and reading data and handles large-scale data efficiently. In addition, it can prevent the loss of data because a value of Replication is set by default to 3, even if it breaks down.

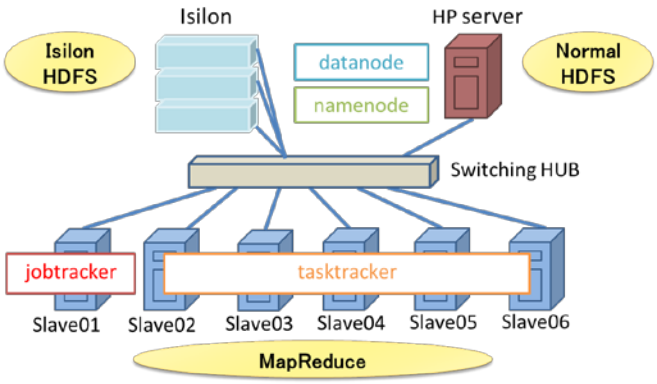

Figure 1. Experiment environment.

B. Isilon

Isilon is scale-out NAS developed by Isilon system. The difference with Isilon and conventional scale-out NAS is the point where NAS controller is not separated.

Isilon system includes CPU, memory and a network device. Furthermore, the system includes software, such as the file system. These devices are called a node and can be expanded from 3 to 144 nodes and the addition of the node is easy. Isilon repositions data automatically and optimizes it if a node is added.

The conventional NAS came to be able to cover it until large-capacity data because the capacity of the disk increased although there is a fault in the scalability in conventional NAS. However, time to load data increases when quantity of data to treat increases because capacity has a limit in the simple substance. Scale-out NAS is designed to solve such a problem and to be able to be expanded to cope with data increasing every day.

IV. PERFORMANCE EVALUATION USING THE BENCHMARK

In this paper, we built a Hadoop environment using Isilon for HDFS, and we performed the comparison in the case with normal HDFS and Isilon for HDFS using a benchmark about writing performance and reading performance.

A. Experiment conditions

The hardware which we used for this experiment is six calculation nodes, Isilon and normal PC (made by Hewlett Packard). Figure 1 shows our experiment environment. We call this normal PC HP server afterward. HP server has the performance that is equal to Isilon. Each calculation node is comprised of CPU (Intel core i5 2500K 3.3GHz), Memory (8GB) and HDD (1TB SATA (7,200rpm)*1). Isilon is comprised of CPU (Nehalem Quad Core), Memory (6GB) and HDD (500GB SATA (7,200rpm)*12). The HP server is comprised of CPU (Xeon E5607 2.26GHz), Memory (6GB) and HDD (1TB SATA (7,200rpm)*8). About the OS, calculation node and the HP server are CentOS6.2, and Isilon is OneFS [11].

One JobTracker and five TaskTracker perform the MapReduce processing. Isilon performs both NameNode and DataNode about the HDFS processing. In comparison, we carried out a similar experiment using HP server.

The number of the Map tasks of each node of Hadoop is two by default and the number of Reduce tasks is one by default. In tour study, we set the number of Map tasks to two and the number of Reduce tasks to two. The number of the replicas of normal HDFS is one by default. The number of the nodes and replicas of Isilon is three because Isilon usually holds three replicas [12].

B. Experiment description

We use three benchmarks (Teragen [13], Grep [9], and Terasort [13]), which are attached to Hadoop. There are one of the Hadoop's widely used benchmarks. Hadoop's distribution contains the input generator, finding keywords and sorting implementations. We performed the comparison about writing and reading performance. In the experiment, we changed parameters of the block size and compared the tendency of the transaction speed. We performed the measurement three times and show the average results. The processing to be carried out in each benchmark is as follow.

a) Teragen: Teragen is the program that only a designated number generates character string of 100 bytes per one record. It is used to measure writing performance.

b) Grep: Grep is a program to count the number of times of appointed character string to develop in input data. It is used to measure reading performance.

c) Terasort: Terasort is a program to output after sorting input data. It is used to measure writing performance and reading performance.

 ISBN: 978-1-61208-311-7

In Grep and Terasort, it is necessary to prepare for data to calculate. Therefore, we generate the random character string data using Teragen. The size of data to treat by this experiment is 20GB, 40GB and 60GB. The parameter of block size in Teragen and Grep is 32MB, 64MB, 256MB, 512MB and 1GB. And the parameter of block size in Terasort is 32MB, 64MB and 256MB.

It is possible to set block size from 4KB to 1GB in Isilon. A value of the defaults of the block size is 64MB. We set a value of 32MB, 64MB, 256MB, 512MB and 1GB in our experiments.

C. Results

Figures 2-4 show the results of the experiment by normal HDFS. Figures 5-7 show the results of the experiment by Isilon for HDFS. The vertical axis shows the processing time, and the cross axle shows data size.

As a result, we understand that the processing time became short, so that block size is big by the Teragen of normal HDFS. In the case of data size 60GB, there is the difference of approximately 50 seconds for 32MB, 64MB and 256MB, 512MB, 1GB. As the results of Teragen in Isilon, the difference of the processing time by the difference in block size is not seen. However, there is difference in 100 seconds when we compared the processing time of Isilon with the processing time of normal HDFS in the case of data size 20GB.

As the results of Grep, we understand that there is little processing time if block size is big in normal HDFS as Teragen. In addition, as the results of Teragen, it followed that the differences between 32MB and 64MB spread as data size grew big. The Grep of Isilon resembled processing of normal HDFS in a tendency. The processing time becomes short if a value of the block size is big although 256MB is slightly earlier than 1GB.

Finally, as the Terasort results, we understand that the processing time tended to be fast if block size is big by the processing by normal HDFS. In addition, the difference gradually spread as data size becomes big. In contrast with this, the results of 32MB and 64MB are the earliest block size by the Terasort of Isilon and a big difference matched the result of 256MB.

As the above experimental results, we show different trends in normal HDFS and Isilon. In normal HDFS, processing time is the best if the block sizes is 1GB in read and write processing. And processing time tends to become slow as block size becomes small. On the other hand, processing time is good using Isilon if block size is big in Teragen and Grep as normal HDFS. In contrast to normal HDFS, processing time tends to become fast as block size becomes small.

In our results, the read/write performance of Isilon is equal to or greater than normal HDFS. In addition, the number of replicas of HDFS is one and Isilon is three in the experiment. Therefore, we consider that the difference in performance between the HDFS and Isilon is become greater because the replica number is 2 or 3 using HDFS usually.

V. CONCLUSION

In this paper, we built a Hadoop environment using Isilon for HDFS. And we also performed the comparison in the case with normal HDFS and Isilon for HDFS using a benchmark about writing performance and reading performance. We are going to investigate a tendency when we changed the number of nodes to use and the number of disks.

REFERENCES

[1] "All of scale out NAS "Isilon"": http://ascii.jp/elem/000/000/730/730401/ [retrieved: October, 2013]

[2] Isilon Scale-out Network Attached Storage (NAS) for Big Data – EMC: http://www.emc.com/domains/isilon/index.htm. [retrieved: October, 2013]

[3] Dhruba Borthaku.: "The hadoop distributed file system: Architecture and design", 2007.

[4] GHEMAWAT, Sanjay; GOBIOFF, Howard; LEUNG, Shun-Tak. The Google file system. In: ACM SIGOPS Operating Systems Review. ACM, 2003. pp. 29-43.

[5] Asaha Ishii, Yonghwan Kim, Junya Nakamura, Fukuhito Ooshita, Hirotsugu Kakugawa, and Toshimitsu Masuzawa, "Evaluation of Hadoop system consisting of Virtual Machines on Multi-core CPUs", High Performance Computing (HPC) 2012-HPC-136(20), pp. 1-7, 2012.

[6] Masashi Sakurai, "Studies on Evaluating Performance Efficiency of Distributed File Systems", 2011.

[7] Asuka Momose and Masato Oguchi, "Characterization of Remote Data Access for Hadoop Distributed File System over a long-latency environment", The institute of Electronics, Information and Communication Engineers DE lab & PRMU lab, 6: 19-24, 2011.

[8] Shohei Okudera, Daisaku Yokoyama, Miyuki Nakano, and Masaru Kitsuregawa, "Consideration on Ad hoc query processing with Adaptive Index in Map Reduce Environment", Technical report of IEICE, The Institute of Electronics, Information and Communication Engineers, 2012.

[9] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Sixth Symposium on Operating System Design and Implementation (OSDI'04), 2004.

[10] Fay Chang, et al., "Bigtable: A Distributed Storage System for Structured Data", ACM Transactions on Computer Systems (TOCS), vol. 26 Issue 2, 2008.

[11] "Construction of the Hadoop storage environment by EMC Isilon scale out NAS", 2012.

[12] "High availability and data security of EMC ISILON scale out NAS", 2012.

[13] Tom White, "Hadoop: The Definitive Guide, 3rd Edition", O'Reilly Media / Yahoo Press, 2012.

 ISBN: 978-1-61208-311-7

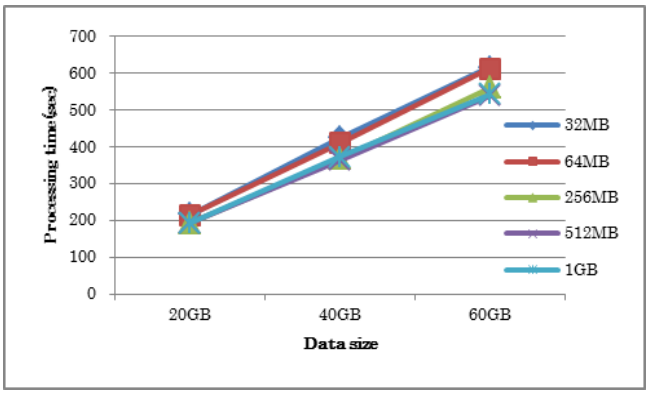

Figure 2. Teragen(HDFS).

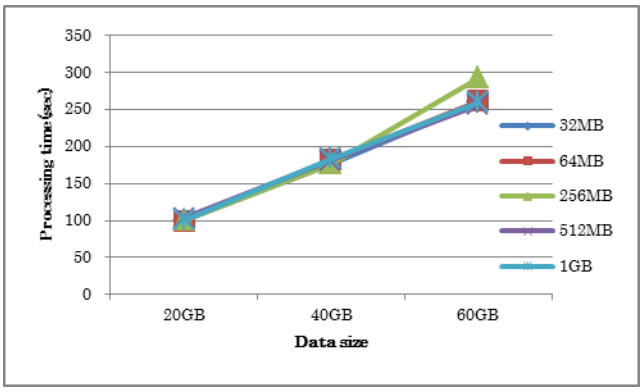

Figure 5. Teragen(Isilon).

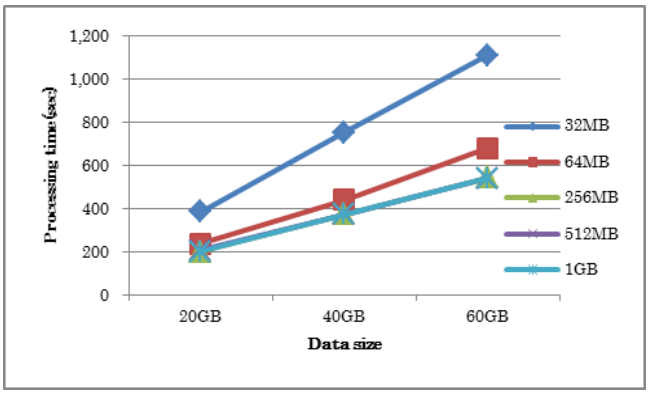

Figure 3. Grep(HDFS).

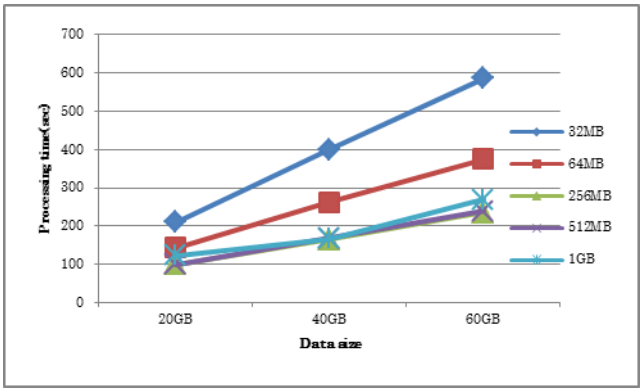

Figure 6. Grep(Isilon).

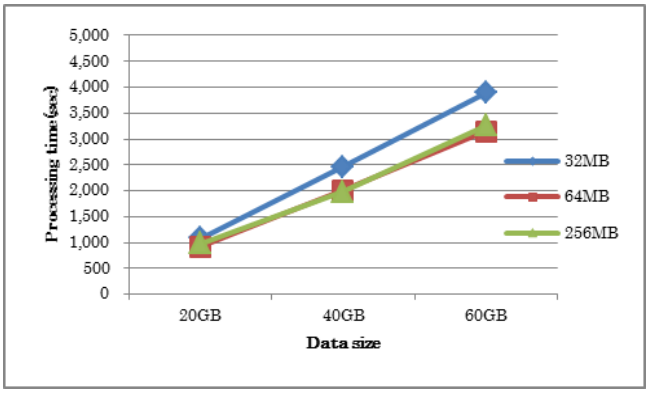

Figure 4. Terasort(HDFS).

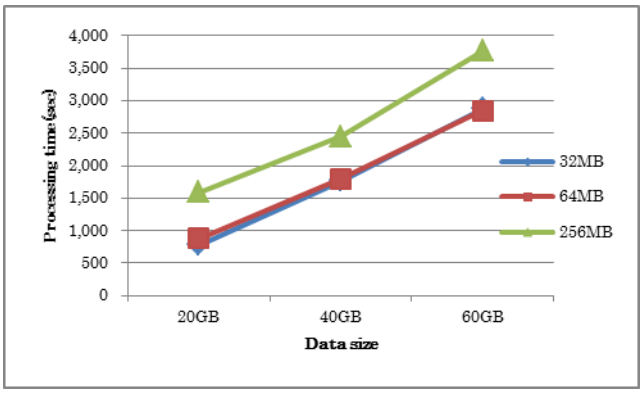

Figure 7. Terasort(Isilon).