A Proposal of Storage Power Control Method with Data Placement Control

Naho IIMURA♡
Norifumi NISHIKAWA♢
Miyuki NAKANO♠
Masato OGUCHI♣

In recent years, the scale of datacenters has become larger due to the explosive increase in the amount of digital data. As a result, the growth of energy consumption is an important factor in the management cost of datacenters. Storing and processing such large volumes of data by database applications are the core technologies in this Big Data era. However, storage accounts for a significant percentage of a datacenter's energy consumption. Therefore, we try to reduce the energy of storage to save on the total cost of datacenters. The purpose of this study is to reduce the energy consumption of storage while minimizing the deterioration of application performance. Although many methods for storage energy saving have been discussed, since it is difficult to control it efficiently only at the storage level, we have investigated the storage power control mechanism on middleware (database) layer. In this paper, we use TPC-H (a database benchmark) as an application example of data processing. We evaluate the data placement control method of storage proposed for energy saving in the database runtime processing suitable for a large-scale environment with many HDDs.

1 Introduction

In recent years, the scale of datacenters has become larger due to an explosive increase of digital data. The volume of digital data ten years from now is estimated to be approximately 44 times larger than that of the present day. Because the amount of storage is increasing, management and operation costs of storage should not be overlooked, and efficient management of data has been focused on.

The energy consumption of datacenters in the world in 2050 is estimated to be about three times larger than the total amount of power generation in Japan, where more than 120 million people are living in 2010[1]. In this society, where energy saving is needed, it is urgent to reduce the energy consumption of datacenters in which huge volumes of data are stored. Storing and processing such huge volumes of data by database applications are the core technologies in this Big Data era. Because storage accounts for 13% of datacenter energy consumption, reducing the power consumption of storage is an efficient way to save energy in datacenters.

In terms of energy saving of storage, it is possible to reduce electric power consumption by shifting hard disks from an active state to a standby one when they are not accessed. However, if hard disks are accessed when they are at the standby state, they must be shifted back to the active state first, so that application performance should be degraded. Additionally, shifting hard disks from the standby state to the active one consumes electric power more than that of keeping them to be the active state. Therefore, shifting hard disks to the standby state should be executed at an appropriate timing, and its decision is crucial for energy saving of storage.

In order to reduce electric power consumption of storage, there are several approaches in some layers of the system. First, it is possible to make a decision at an upper layer, the application level. Because applications know when I/O is issued and when it is not, it is possible to decide the timing to shift hard disks to the standby state by analyzing applications for achieving energy saving of storage. However, for this approach, we must analyze all applications executed on the system and it is not realistic. On the other hand, I/O can be observed at a lower layer, storage level. If hard disks are shifted to the active state when I/O is issued and shifted to the standby state when it is not at this layer, electric power consumption of storage is reduced. However, it is extremely difficult to decide the timing of I/O issues at the storage level.

Compared with these approaches, it is realistic to make a decision at middleware, the database layer. Because SQL sentences are analyzed to achieve optimal execution of applications at the database layer, the timing of I/O issues can be decided when it is executed. Therefore, it is possible to reduce electric power consumption of storage by shifting hard disks to the standby state based on this observation in database runtime processing.

In our research, we save the power from database processing in the cloud computing systems through efficient management of data, and the purpose of this study is to reduce the energy consumption of storage while minimizing the deterioration of application performance. Although energy saving for storage has been discussed in many literatures, it is difficult to predict the behavior and control it efficiently only at the storage level. In addition, although static analysis of the behavior of applications is studied intensively, it is hard to predict their dynamic behavior during the execution. Therefore, we have investigated the dynamic storage power control mechanism during the execution on middleware, the database layer.

In this paper, we use TPC-H as an application example of data processing, which is a widely used database benchmark that executes typical decision support processing on data [2]. First, power saving during runtime processing is investigated. Next, based on the analytical result, a data placement control method is proposed in which data allocation is changed depending on the access frequencies. We evaluate the control method of storage proposed for energy saving in the database runtime processing suitable for a large-scale

♡ Student Member Ochanomizu University
naho@ogl.is.ocha.ac.jp
♢ Member Yokohama Research Laboratory, Hitachi, Ltd.
norifumi.nishikawa.mn@hitachi.com
♠ Member Shibaura Institute of Technology
miyuki@sic.shibaura-it.ac.jp
♣ Member Ochanomizu University
oguchi@computer.org

environment with many HDDs.

The remainder of this paper is organized as follows. Section 2 explains related works of our research. Section 3 describe our proposed method. Section 4 introduces the experiment environment. Power consumption characteristics of HDDs are evaluated in Section 5. Section 6 shows that the energy savings and performance of the storage are improved by our proposed method using data placement control. Section 7 presents our concluding remarks.

2 Related Work

Thus far, many methods for storage energy saving have been proposed [3],[4],[5],[6],[7]. In these studies, various methods that suspend disks according to the I/O interval for storage are proposed to realize energy savings in storage. However, it is not easy to predict the storage level I/O behaviors precisely.

In addition, there are many studies about static methods for power saving of service by an analysis of applications before their execution. In practice, however, the power-saving method during the execution of the service has not been studied. While this cannot be solved by the platform provider, tailored power control to suit a specific application on the user side with different applications is also highly expensive. Therefore, by putting the power saving function on the middleware layer (database in this study) that can monitor the control of input and output, we have tried the storage power saving control in runtime processing that does not depend on the application.

An energy saving method with efficient usage of storage by cooperative applications is proposed [8],[9]. To construct an energy efficient storage management system combined with data-intensive applications, a power saving method for storage is proposed that utilizes application level I/O behaviors. The power consumption of the storage can be reduced by using the proposed method.

We are interested in the performance of data-intensive applications on datacenters in addition to the power savings of the system. Therefore, we focus on the Service Level Agreement (SLA) that copes with both energy savings and the performance of storage. The goal of this study is to reduce energy consumption of storage while the deterioration of application performance is minimized. In this paper, we evaluate the proposed method suitable for larger environments with many hard disks.

When the power saving in storage is discussed, to replace hard disks to Solid State Drives (SSDs) is one of the candidate options. Many literatures have discussed to save energy by using SSD [11],[12],[13]. It depends on the cost whether it is possible to replace hard disks to SSDs or not, and it is expected to replace them in the future. In any case, our approach can also be applied to storage composed of SSDs. Additionally, the power consumption of SSD for shifting from the standby state to the active one is much less than that of hard disks. Therefore, our proposed method should be promissing even more in the era of SSDs.

3 Proposed Method

In this section, we describe our proposed method. Our proposed method makes use of data placement control for energy saving of storage. Generally, in a real environment, application data is divided and placed almost equally on each HDD if it is larger than a single disk. That is, each HDD has almost equal amount of data. We call it “Naive Method” as a general data placement method. However, in the naive method, achieving efficient energy saving is difficult because I/O is issued to each HDD almost evenly. Therefore, we modify the data placement to get longer I/O interval (the period until the next I/O is issued). Accordingly, we control the energy state of HDDs and change them to energy saving (Standby) mode while I/O is not issued. Thus, we can reduce energy consumption of HDDs during runtime of appllications.

In a database used in a real environment, the frequency of access to each data is uneven. Furthermore, it is possible to know the frequency of access to each data based on collected statistics at middleware, the database layer. For this reason, our proposed method can be generalized for a real environment by investigation and reference to statistics of data access frequency, and modifying the data placement.

Energy consumption and system performance are more important for runtime of data intensive applications than that of CPU intensive applications. Therefore, in our research, our proposed method is evaluated using TPC-H (the standard of database benchmark) as a representative of data intensive applications.

As a first step for preparaions of evaluation, we investigate the variety of transition states of HDDs and measure the power consumption of each state. Next, we calculate Break-Even Time that is an indicator for energy saving of storage.

After that, we evaluate our proposed method. The evaluation plan of our proposed method is following. First, we investgate I/O frequencies of TPC-H data. Next, based on the I/O frequencies, we modify the data placement on HDDs. We evaluate our proposed method by comparing its amount of energy comsumption and system performance with those of the naive method during TPC-H runtime processing.

Table 1: The Specifications of the Storage Server and Power Meter.

OS	CentOS 5.10 64bit
CPU	AMD Athlon 64 FX-74 @ 3GHz(4 cores) x2
Memory	8 GB
HDD	Seagate Barracuda 7200 series 3.5 inch SATA 6 Gb/s 3 TB 7200 rpm 64 MB 4K sector x 11
DBMS	HITACHI HiRDB Single Server Version 9
Power Meter	YOKOGAWA WT1600 Digital Power Meter

4 Experiment Environment

We used a storage server and a power meter to construct an experimental environment, which is supposed to emulate a part of datacenters. Table 1 shows the specifications of the storage server and the power meter used for the measurements. The number of HDDs is 11 in total, and 10 of them are used for the data storage. This experimental environment can be accessed and executed remotely.

The power meter is connected to the HDDs of the server and controlled by a dedicated computer. The storage server, power meter, and computer to control the power meter can be controlled remotely for the experiments.

Table 2: Power and Energy Consumption of Disk States.

Standby 1 (W)	Standby 2 (W)	Idle (W)	Active (W)
1.05	0.88	5.22	7.25
Spindown (J)	Spinup 1 (J)	Spinup 2 (J)	
6.79	108.5	105.5	

5 Power Consumption Characteristics of HDDs

In this section, we investigate the variety of transition states of HDDs and measure the power consumption of each state. On the basis of this investigation, we calculate the Break-Even Time, which is a measurement value that indicates the possibility of power-saving.

5.1 Transition States and Power Consumption of HDDs

In this paper, we use four varieties of transition states: Standby 1, Standby 2, Idle, and Active. Spindown means switching state from Idle or Active to Standby 1. Spinup 1 means switching state from Standby 1 to Idle or Active. Spinup 2 means switching state from Standby 2 to Idle or Active.

[9] uses three varieties of transition states: Standby, Idle, and Active. We examined the detailed transition states of the disk used in this study. As a result, the duration of the two different power consumptions during Standby was observed. Therefore, we distinguish them into two types of states during standby: Standby 1 and Standby 2.

We measure the power consumption of the disk in each state. Table 2 shows the power consumption of each state. The values of Standby 1, Standby 2, Idle, and Active states are the maximum. The values of Spindown, Spinup 1, and Spinup 2 states are the average.

5.2 Break-Even Time

Break-Even Time is the amount of time to continue the Standby state that satisfies the following condition. The amount of energy needed for the spinup or spindown of the disk is equal to that of the energy saved by remaining in the Standby state during Break-Even Time. We define the parameters as follows:

E_d: the amount of energy needed for Spindown
E_{u2}: the amount of energy needed for Spinup 2
P_{s1}: the power comsumption of the HDD during the state of Standby 1
P_{s2}: the power comsumption of the HDD during the state of Standby 2
P_i: the power comsumption of the HDD during the state of Idle
T_d, T_{u2}: the amount of time required to Spindown or Spinup 2
T_{s1}, T_{s2}: the amount of time remaining for Standby 1 or Standby 2

Using these parameters, Break-Even Time T_{be} is calculated as follows:

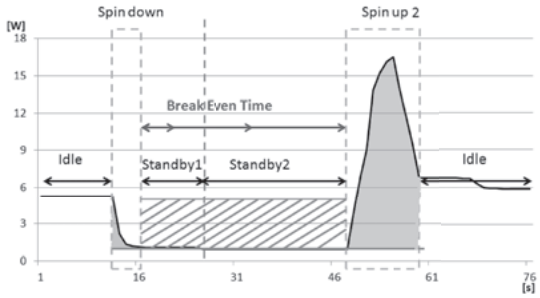

Fig. 1: Power Consumption of the Transition of the Disk and the Break-Even Time.

$$T_{be} = (E_d + E_{u2} - P_{s2} \times T_d - P_{s2} \times T_{u2} + T_{s1} \times (P_{s1} - P_{s2})) / (P_i - P_{s2})$$

We distinguish them into two types of states during Standby. Therefore, we calculated the Break-Even Time by referencing [14]. The Break-Even Time of HDDs used in this measurement was approximately 24 seconds. According to this result, to reduce power consumption by using the Standby state, an I/O interval of approximately 24 seconds or more is needed. Figure 1 shows the transition of disk power consumption used in this measurement. The state transition is: Idle to Standby 1 to Standby 2 to Idle.

6 Data Placement Control

We showed that TPC-H runtime power saving is possible when the I/O interval and the Break-Even Time are used. It is possible to change the state to the energy saving one after a short period of timeout, as no I/O has occurred during that period. However, this energy saving method is too naive because, in this method, we use the simple behavior of the disk without respect to applications.

In this section, we investigate the I/O frequencies of data, tables, and indexes of TPC-H during runtime processing of a TPC-H query. Next, we discuss the placement of data on the disk to control the I/O interval during TPC-H runtime processing. We prepared the environment for cases where the number of used HDDs is 10 maximum, and then we evaluated our proposed method. The remainder of this section is organized as follows. Subsection 6.1 investigate the I/O frequencies of each data. We discuss the data placement method using table partition in subsection6.2. Subsection 6.3 evaluate our proposed method in 10 HDDs environment.

6.1 The Investigation of I/O Frequencies

First, we investigate the I/O frequencies of data, tables, and indexes of TPC-H during runtime processing of a TPC-H query to evaluate our proposed method. We used two patterns of scale factor: 10 and 30. I/O interval is obtained by the `pdbufls` command [15] (DB buffer statistical information retrieval tool that comes with the DBMS) for every second. DB is placed on the raw device. The number of investigated buffers is 23. The survey period is from the beginning to the end of the query execution. The queries are executed in numerical order from Q1 to Q22.

In this investigation, we focus on the actual number of times of the HDD READs among the obtained data items.

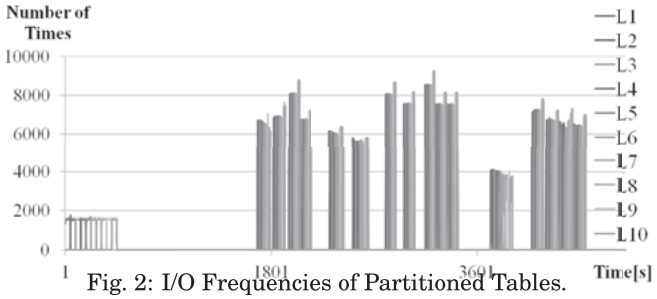

Fig. 2: I/O Frequencies of Partitioned Tables.

The purpose of this experiment is to investigate and analyze I/O frequencies. In general, DBMS is used in the state in which a part of the DB resides in the buffer (called a Hot state). Therefore, the DB is in the Hot state in our experiment. The DB buffer size is approximately 0.58 GB for the table data and approximately 0.21 GB for the index data. The size of the DB varies based on the scale factor. According to the result of the investigation, the number of buffers containing data that have I/O was 13, whereas the number without I/O was 10.

6.2 Data Placement Control with Table Partitioning

In previous papers, we have shown that the proposed method in the three disk environment is effective for power saving[17]. In addition, LINEITEM table stored in one buffer accounts for the most of the amount of data in the TPC-H. Thus, there is a limit to how to change the arrangement. In order to use more flexible arrangement of data, we divided LINEITEM table and indexes into 10 buffers. The data placement control method is evaluated for this arrangement. The scale factor of DB is 10, and we set optimization options (HASH JOIN preferred option) of RDBMS.

6.2.1 Table and Index Partitioning

Two types of table partitioning methods are known, one is the hash partitioning and the other is the key range partitioning. In our method, we use the hash partitioning because of its practical utility. We divided LINEITEM table and indexes into 10 buffers in this experiment, as the maximum number of HDDs is 10.

6.2.2 Investigation of the Input and Output Frequencies

We investigate the I/O frequencies of partitioned LINEITEM data, tables, and indexes during runtime processing of a TPC-H query to evaluate our proposed method. I/O interval is obtained by the `pdbufls` command [15] for every second. Figure 2 shows the I/O frequencies of runtime TPC-H processing of partitioned LINEITEM tables. "xx" of "Lxx" indicated in the figure shows the order number of partitioning. We conjecture partitioned tables are used in numerical order, from Figure 2. Consequently, we confirm a longer I/O interval is obtainable by placing partitioned tables and indexes with near numbers on the same disk.

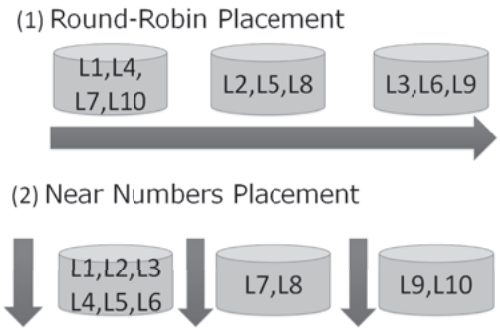

Fig. 3: Two Patterns of Data Placement about Partitioned Data.

Table 3: Number of Times for Length of I/O Issue Interval

I/O Interval (sec)	0-24	25-100	101-200	201-
(1) Round-Robin (times)	111	58	9	8
(2) Near Numbers (times)	12	35	14	14

6.2.3 The Method of Data Placement

ALL data of TPC-H are placed on three HDDs. As shown in Figure 3, we design two patterns of data placement about partitioned data, tables and indexes.
(1) Partitioned data is placed almost evenly by round-robin placement.
(2) Partitioned data with near numbers are placed on the same HDDs.
Other data is placed such that the amount of data is equal on each HDD.

6.2.4 Evaluation of Two Patterns of Data Placement

Regarding two patterns of data placement, we investigate and compare I/O frequencies of each HDD during runtime processing of TPC-H. "I/O frequencies" means "the span and number of times of I/O" in this investigation. Table 3 shows the number of times for the length of I/O issue interval of the three disks on each data placement method. We confirm that method (1) (round-robin placement) tends to be shorter I/O interval, which is less than the Break-Even Time. In other words, placement of partitioned data almost evenly by round-robin is inefficient in this case.

With both of placement methods, we compare the amount of energy saving and the rate of delay with and without the standby state. Figure 4 shows the comparison of the amount of energy consumption, and Figure 5 shows the comparison of the runtime processing of TPC-H. In the method (2), we succeed in saving energy about 33% by switching the disk to the standby state, and the delay rate of runtime processing TPC-H is about 8%. In contrast, in the method (1), the saving energy is about 15%, and the delay rate is about 22%. As a result, it is possible to reduce the power consumption more in the method (2) because I/O intervals are longer. Besides, the delay rate of the method (2) is smaller than that of the method (1) because the seek overhead is smaller. Therefore, in this environment, more efficient energy saving is available by placing partitioned data with near numbers on the same

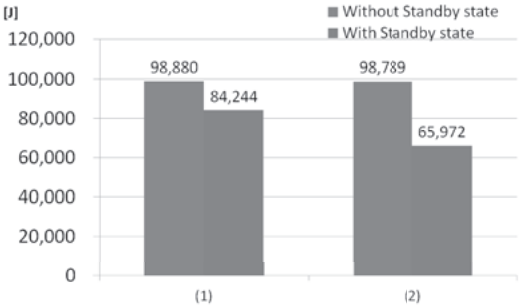

Fig. 4: Comparison of the Amount of Energy Consumption among Different Data Placement (three-disks).

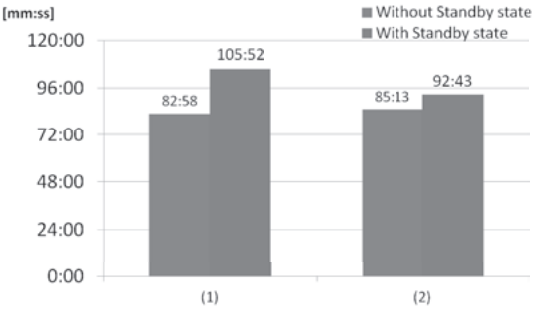

Fig. 5: Comparison of the Runtime Processing among Different Data Placement (three-disks).

disk, compared with the placement by round-robin.

6.3 Evaluation of Proposed Method in 10 HDDs Environment

We have already evaluated our storage power control method [17] in a small-scale environment, and this method can be applied to a large-scale environment with many hard disks. Based on the previous result, we assess our proposed method by preparing 10 HDDs. We prepared three kinds of disk environment. The numbers of used HDDs are 10, 5 and 2. In addition, we also prepared two placement methods in each environment, “Naive Method” and “Proposed Method.” “Naive Method” is the simple method. “Proposed Method” is the placement that the data is placed based an I/O frequencies. On the other hand, “Naive Method” is the placement in which partitioned data is placed almost evenly without considering I/O frequencies.

6.3.1 Data Placement

In this evaluation, we prepared following data placement.

1. 10 HDDs Environment
   - Naive Method (10 Hot HDDs):
     Data are placed on ALL HDDs without consideration of data I/O frequeny. The number of Hot HDDs (placed I/O data) is ten.
   - Proposed Method (9 Hot HDDs):
     Partitioned data of LINEITEM tables are placed on HDD1, HDD2, HDD3. The rest of the data that have I/O is placed each schema on one of HDDs (HDD4 - HDD9). The data that have no I/O is placed on HDD10. The number of Hot HDDs (placed I/O data) is nine.
2. 5 HDDs Environment
   - Naive Method (5 Hot HDDs):
     Data are placed on 5 HDDs (HDD1 - HDD5) without consideration of data I/O frequeny. No data is placed on HDD6 - HDD10. The number of Hot HDDs (placed I/O data) is five.
   - Proposed Method (4 Hot HDDs):
     LINEITEM tables are placed on HDD1, HDD2, HDD3, and other data that have I/O on HDD4. In addition, we placed the data that have no I/O on HDD5. No data is placed on HDD6 - HDD10. As mentioned in Section 6.2.1, we partitioned LINEITEM table. Partitioned data with near numbers are placed on the same HDDs. To be specific, L1-L4 are placed on HDD1, L5-L7 are placed on HDD2, L8-L10 are placed on HDD3. The number of Hot HDDs (placed I/O data) is four.
3. 2 HDDs Environment
   - Naive Method (2 Hot HDDs):
     Data are placed on 2 HDDs (HDD1, HDD2) without consideration of data I/O frequeny. No data is palced on HDD3 - HDD10. The number of Hot HDDs (placed I/O data) is two.
   - Proposed Method (1 Hot HDD):
     We classfied the data into 2 types: (1) the data that have I/O, (2) the data that have no I/O. (1) is placed on HDD1, (2) is placed on HDD2. No data is placed on HDD3 - HDD10. The number of Hot HDDs (placed I/O data) is one.

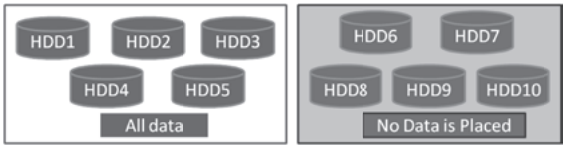

Fig. 6: Example of Data Placement in 5 HDDs Environment (Naive Method).

We select this placement to investigate the relationship between the storage power consumption and the execution time of the query. We can investigate how much performance is obtained with a heavy load applied to the disk.

The power consumption states of Hot HDDs are Idle or Active in each “Naive Method.” We set timeout (5 seconds) to ALL HDDs in “Proposed Method (9 Hot HDDs)” of 10 HDDs environment, set timeout (5 seconds) to switch to Standby ALL HDDs in “Proposed Method (4 Hot HDDs)” of 5 HDDs environment. As a different pattern of “Proposed Method (4 Hot HDDs)”, we set timeout to HDD1, HDD2, HDD3, HDD5-HDD10. We call this placement as “Proposed Method’ (4 Hot HDDs’)” . We set the timeout (5 seconds) switch to Standby HDD2-HDD10 in “Proposed Method (1 Hot HDD)” of 2 HDDs environment.

We show examples of data placement about naive method and proposed method of 5 HDDs environment in Figure 6 and Figure 7. The red HDD indicates the state of HDD is Hot. The blue HDD indicates the state of HDD is Cold (the no I/O data is placed) or no data is placed. In addition, the HDDs enclosed in gray indicates that we set the timeout (5 seconds) switch to Standby.

Fig. 7: Example of Data Placement in 5 HDDs Environment (Proposed Method).

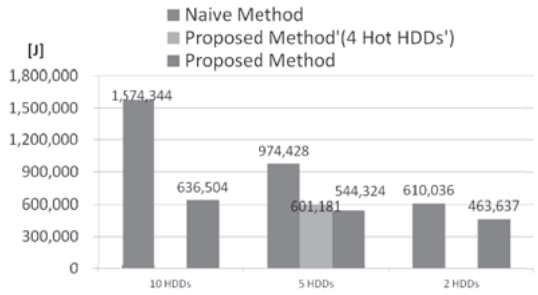

Fig. 8: Comparison of the Amount of Energy Consumption among Different Data Placement (10 HDDs Environment).

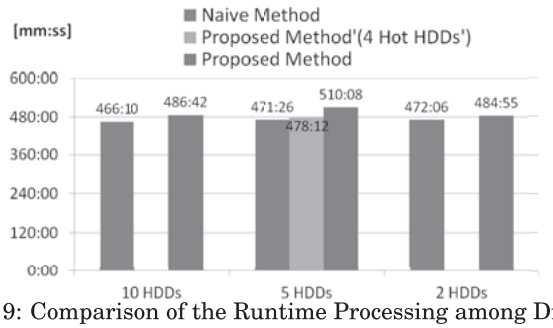

Fig. 9: Comparison of the Runtime Processing among Different Data Placement (10 HDDs Environment).

Table 4: Relation of Energy Consumption Rate and Delay Rate

Disk Environment	10	10	5	5	5	2	2
Number of Hot HDDs	10	9	5	4'	4	2	1
Energy Consumption Rate (%)	100	43	66	41	37	41	31
Delay Rate (%)	0.0	4.4	1.1	2.6	9.4	1.3	4.0

6.3.2 Evaluation of System Performance and Energy Comsumption

We compare energy consumption and system performance during runtime of TPC-H processing of each data placement. The scale factor of DB is 30. The volume of data with this scale factor can be stored in 1-2 HDDs actually. However, we used 10 HDDs maximum in our experiments to evaluate cases for processing much larger DBs.

Figure 8 shows the comparison of energy consumption with and without the control of data placement. Figure 9 shows the comparison of query processing time. Table 4 shows the energy consumption rate and delay late of response time against those of 10 Hot HDDs. 4' in the table means "Proposed Method' (4 Hot HDDs') " in 5 HDDs environment.

As shown in Figure 8, the amount of power consumption is remarkably different depending on the numbers of Hot HDDs. We succeed in saving energy greatly with using our proposed method, "Proposed Method (1 Hot HDD)" of 2 HDDs environment. This is because the data that have I/O is placed one HDD, and power consumption state of other HDDs are Standby. As shown in Figure 9, the delay rate of response time of "Proposed Method (4 Hot HDDs)" of 5 HDDs environment is the highest than other placement. This is due to spin up overhead caused by setting timeout to switch the standby state HDD4. However, "Proposed Method' (4 Hot HDDs')" of 5 HDDs environment that sets timeout to the HDDs other than HDD4 takes much smaller delay rate. In any case, though the delay rate of the naive methods are smaller compared with that of the proposed methods with the same number of used HDDs, the delay rate is a few percent at most, which is much smaller than the decreased rate of energy consumption that achieves tens of percent. From the above, our proposed method, data placement control, is effective for energy saving of storage.

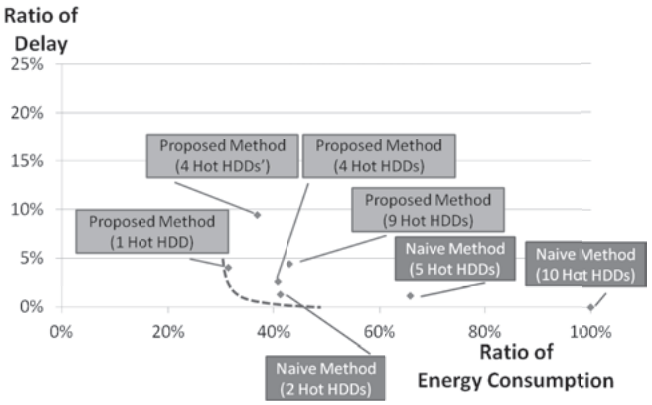

Fig. 10: Relation between Power Consumption and Query Processing Time.

Figure 10 shows the relation between power consumption and query processing time. The dot-line shows the estimated pareto optimal curve in Figure 10, on which both the consumption rate of energy and the delay rate of response cannot be reduced simultaneously. An ideal limit line should exist around here. The points of "Proposed Method " are closer to this line. From the above, our proposed method can achieve closer to the pareto optimal compared with the naive method using the same number of HDDs.

7 Conclusion

We consider energy savings of datacenters by reducing the energy consumption of storage through the efficient management of data. In this paper, the evaluation of a data placement control method we proposed suitable for a large-scale system environment is shown.

Based on the existing research, we analyzed the performance and energy consumption during runtime disk access. Next, in consideration of two patterns of standby (the energy saving state of the disk), we calculated Break-Even Time precisely. Furthermore, as an evaluation of our proposed method, we use TPC-H (a database benchmark) as a data-intensive application and evaluate the control method of storage we proposed for energy savings during the runtime database benchmark. The data placement control method is shown to be effective for energy savings during runtime ap-

plication processing. Furthermore, we evaluated our method considering the SLA, in which the energy consumption of storage is reduced while the deterioration of application performance is minimized. Comparing naive method, while suppressing the performance deterioration, we showed that our proposed method can achieve energy saving of storage.

Future works include an examination of more detailed data placement on 10 disks for energy savings. In addition, a detailed examination of the relationship of trade-off between performance (delay of runtime processing) and power consumption of storage is needed. We will perform these examinations.

8 Acknowledgment

This work is partly supported by the Ministry of Education, Culture, Sports, Science and Technology, under Grant 24300034 and 25280022 of Grant-in-Aid for Scientific Research.

[Bibliography]

[1] GIPC Survey and Estimation Committee Report FY2009 (Summary), http://www.greenit-pc.jp/activity/reporting/100707/index.html, 2009

[2] TPC-H: http://www.tpc.org/tpch/default.asp

[3] Jorge Guerra, Himabindu Pucha, Joseph Glider, Wendy Belluomini, and Raju Rangaswami: Cost Effective Storage using Extent Based Dynamic Tiering, In Proc. 9th USENIX Conference on File and Storage Technologies, pp.1-14, 2011.

[4] Dushyanth Narayanan, Austin Donnelly, and Antony Rowstron: Write Off-Loading: Practical Power Management for Enterprise Storage, In Proc. 6th USENIX Conference on File and Storage Technologies, pp.253-267, 2008.

[5] Athanasios E Papathanasiou and Michael L Scott: Energy Efficient Prefetching and Caching, In Proc. the annual conference on USENIX Annual Technical Conference, 2004.

[6] Akshat Verma, Ricardo Koller, Luis Useche, and Raju Rangaswami: SRCMap : Energy Proportional Storage using Dynamic Consolidation, In Proc. 8th USENIX Conference on File and Storage Technologies, 2010.

[7] Charles Weddle, Mathew Oldham, Jin Qian, An-I Andy Wang, Peter Reiher, and Geo Kuenning: PARAID: A Gear-Shifting Power-Aware RAID, In Proc. 5th USENIX Conference on File and Storage Technologies, Vol. 3, pp. 245-260, October 2007.

[8] Norifumi Nishikawa, Miyuki Nakano, and Masaru Kitsuregawa: Runtime Disk Energy Saving Method Using Application I/O Behavior and Its Evaluation : Energy Saving Efficiency for Online Transaction Processing, IEICE transactions on information and systems, Vol.J95-D, No.3, pp.447-459, March 2012.

[9] Norifumi Nishikawa, Miyuki Nakano, and Masaru Kitsuregawa: Energy Efficient Storage Management Cooperated with Large Data Intensive Applications, In Proc. 28th IEEE International Conference on Data Engineering (IEEE ICDE 2012), pp.126-137, April 2012.

[10] Naho Iimura, Norifumi Nishikawa, Miyuki Nakano, and Masato Oguchi: A Proposal of Storage Control Method for Energy Saving on Runtime Database Processing, In Proc. Multimedia, Distributed, Cooperative, and Mobile Symposium (DICOMO 2013), pp.1646-1652, 7C-1, July 2013.

[11] Jian Ouyang, Shiding Lin, Zhenyu Hou, Peng Wang, Yong Wang, and Guangyu Sun. Active SSD design for energy-efficiency improvement of web-scale data analysis, IEEE International Symposium on Low Power Electronics and Design (ISLPED 2013), pp.286-291, September 2013.

[12] Peng Li, Gomez, K., Lilja, D.J. Exploiting free silicon for energy-efficient computing directly in NAND flash-based solid-state storage systems, IEEE High Performance Extreme Computing Conference (HPEC 2013), pp.1-6, September 2013.

[13] Devesh Tiwari, Sudharshan S. Vazhkudai, Youngjae Kim, Xiaosong Ma, Simona Boboila, and Peter J. Desnoyers. Reducing Data Movement Costs Using Energy-Efficient, Active Computation on SSD, USENIX Workshop on Power-Aware Computing and Systems (HotPower '12), October, 2012.

[14] Y.H. Lu, G.D.Micheli: Comparing System-Level Power Management Policies, IEEE Design & Test of Computers, Vol.18, No.2, pp.10-19, March 2001.

[15] pdbufls: http://www.hitachi.co.jp/Prod/comp/soft1/manual/pc/d635540/W3550027.HTM

[16] smartd.conf: http://smartmontools.sourceforge.net/man/smartd.conf.5.html

[17] Naho Iimura, Norifumi Nishikawa, Miyuki Nakano, Masato Oguhci, An Evaluation of Data Placement Control Method on Runtime Database Benchmark Considering System Performance and Storage Energy Saving, In Proc. Multimedia, Distributed, Cooperative, and Mobile Symposium (DICOMO 2014), pp.1818-1825, July, 2014.

Naho IIMURA
graduated from Ochanomizu University, Information Science in 2013. She is a master course student of Graduate School of Humanities and Sciences, Ochanomizu University. She has been engaged in research on storage power control method with data placement control.

Norifumi NISHIKAWA
received his BFA from the Department of Measurement Engineering, Kobe University, in 1989. He received his MFA from the Department of Instrumental Engineering, Graduate School of Engineering, Kobe University, in 1991 and joined Hitachi, Ltd., that same year. He received his PhD from the Department of Electronics and Information, Graduate School of Information Science and Technology, University of Tokyo, in 2012.

Miyuki NAKANO
received her BS and her Ph.D from the University of Tokyo, respectively. In 1981, she worked at Fujitsu Corporation and in 1985, she began working at the University of Tokyo. In 2004, she became an associate researcher at the Institute of Industrial Science, University of Tokyo. In 2009, she became an associate professor at the same institute. She is currently a professor at the Shibaura Institute of Technology. Her interests include the study of database systems and storage systems. She is currently engaged in engineering data research.

Masato OGUCHI
received Ph.D from the University of Tokyo in 1995. He joined Ochanomizu University in 2003 as an associate professor. Since 2006, he has been a professor at the Department of Information Sciences, Ochanomizu University. His research field is in network computing middleware, including high performance computing as well as mobile networking.