An Evaluation of Data Placement Methods for Multiple-gear Power-proportional Distributed File Systems

Hieu Hanh Le, Haruo Yokota
Department of Computer Science, Tokyo Institute of Technology, Tokyo, Japan
hanhlh@de.cs.titech.ac.jp, yokota@cs.de.cs.titech.ac.jp

Abstract— *Recently, power-aware distributed file systems for efficient Big Data processing has increasingly moved towards power-proportional designs. However, inefficient gear-shifting in such systems has become a vital issue that could hugely degrade the performance of the systems. To address this issue, we previously proposed a power-proportional data placement method known as Accordion, which uses data replication to comprehensively arrange the data layout and provide efficient gear-shifting by reducing the amount of re-transferred data. In this paper, we discussed in detail the effectiveness of Accordion and another data placement method relating to block size, which is configurable in many distributed file systems like the Hadoop Distributed File System (HDFS) or the Google File System. Extensive empirical experiments using actual machines based on the HDFS demonstrated that suitable block size setting was able to shorten the execution time of the data reallocation process in gear-shifting by more than* 40%.

Keywords: green data centers, power-proportional, gear-shifting, HDFS, data placement

1. Introduction

The power consumption at data centers all over the world has been increasing. U.S. data centers consumed an estimated 91 billion kilowatt-hours of electricity in 2013 and expectedly consumes nearly 140 billion kilowatt-hours in 2020 [1]. This trend has also been observed in other reports in Europe [2] and developing Asia countries [3], [4]. Furthermore, driven by Big Data and IoT, a huge increasing volume of data is generated as by 2025 the global datasphere will grow to 163 ZB which is ten times the 16.1 ZB of data generated in 2016 [5]. Such a huge volume of data is needed to be efficiently stored and managed by storage systems, which have become the next largest component after servers and cooling systems that consume from 25% to 35% of the total power consumption in the data centers [6], [7]. Hence, not only performance-efficient but also energy-efficient strategies for storage systems have been a challenging problem in both academia and industry.

Moreover, power-aware storage systems are progressively moving towards power-proportional design [8]. In these systems, the power-proportionality can be achieved using well-designed data placement methods to control the total number of active nodes to store and retrieve the data [9]–[11]. The common idea in these methods is to place the data's replicas at organized nodes. The system divides all of the nodes into a set of small separated groups. These groups are then configured to operate in multiple gears where each gear contains a different number of groups and offers a different level of parallelism and aggregate Input/Output workload.

We previously proposed and showed the efficiency of a flexible and reliable data placement method, named Accordion, for efficient gear-shifting in multiple-gear power-proportional distributed file systems [10]. By carefully designing the location of the primary data, Accordion reduced the amount of the data that should be transferred during gear-shifting. During operations, the system may have to update the data sets modified in a low gear when a subset of the nodes was powered off for decreasing power consumption. When the system moves to a higher gear to gain a better power-proportionality by reactivating inactive nodes, it needs to provide the new power proportionality in serving the requests from clients. Therefore, in the background, the system has to internally re-transfer the updated data to the reactivated nodes to share the load equally among all of the active nodes in order the obtain better performance. Inefficient gear-shifting with a large amount of re-transferred data is believed to degrade the performance of power-proportional distributed file systems greatly.

Furthermore, in distributed file systems, to increase parallelism in accessing files, generally a file is divided into many blocks, in which the block size is predefined. The block size is needed to be optimized to gain the best performance from the file systems which support a wide range of Big Data and IoT applications. As a result, the effect of block size to the file systems performance is very important and is needed to be well studied.

In this paper, we evaluate the effect of block size on the efficiency of data placement methods relating to efficient gear-shifting at a multiple-gear power-proportional distributed file system. We measured the execution time for reflecting the updated data during gear-shiftings with varying the block size for Accordion and Sierra [11] which share the similar design. From the experimental results, it is verified that suitable block size improves the performance of updated data

ISBN: 1-60132-487-1, CSREA Press ©

reflection process in both methods. In general, compared with Sierra, Accordion can shorten the gear-shifting time in all configurations, especially when the gear-shifting process occurs at a high gear.

The remainder of this paper is organized as follows. Related studies are discussed in Section 2 and the design of Accordion is described in Section 3. Section 4 presents an experimental evaluation and Section 5 summarizes this paper.

2. Related work

There have been two main approaches relating to improving the energy efficiency of the storage systems in the data centers, i.e., Power-saving and Power-proportional approach.

2.1 Power-saving approach

There have been a number of studies have aimed to reduce the total power consumption based on a trade-off with response performance in the general storage system, rather than power-proportional designs [12]–[15].

PARAID [12] uses a skewed pattern to replicate and stripe data blocks to the disks. It facilitates adaptation to the system load by varying the number of active disks in the system. PARAID focuses only on the RAID unit, and it is unreliable when adapting to a distributed environment.

GRAID [13] is similar to PARAID because it is also a green storage architecture that aims to improve the energy efficiency and reliability of a RAID unit. In this study, the data-mirroring redundancy of RAID10 has been extended by incorporating a dedicated log disk, which stores all of the updates since the last mirror disk update. Using this log disk information, the system only needs to update the mirroring disks periodically so it can spin down all of the mirroring disks in a low-power mode for most of the time, which saves energy.

One of the first attempts to improve the energy efficiency of HDFS was performed by Leverich et al. [14] who showed that it was possible to recast the data layout and task distribution of HDFS to allow significant portions of a cluster to be powered down while still fully operational. They also confirmed that the energy could be conserved at the expense of performance, so there was a trade-off between the two.

Later, Kaushik et al. [15] proposed an energy-conserving multiple zone approach for HDFS, which utilized life cycle information as data. They divided the storage clusters of the HDFS into hot and cold zones. The frequently accessed data were placed in the hot zone where all of the data nodes were active, so they consumed power. The less frequently accessed data were placed in the cold zone, which allowed data nodes to be inactive. Thus, the power consumption was lower in the cold zone than in the hot zone.

2.2 Power-proportional approach

Recently, the power-aware storage systems have been moving towards power-proportional designs.

Kim et al. [7] proposed a fractional replication method to balance the power consumption and system performance. In this method, the data placement layout was inspired by PARAID where fractional replication and downshifting of the operational modes to a lower gear saved power using a probabilistic prediction model based on historical observations.

Rabbit [9] was the first method to provide power proportionality to the Hadoop Distributed File System (HDFS) by focusing on the read performance. Rabbit uses an equal work–data-layout policy based on data replication. The primary replicas are stored evenly among the primary nodes. The remaining replicas are stored on additional and increasingly large subsets of nodes. Each node in the subsets has a fixed order, and it stores blocks that is inversely related to its order, which guarantees that the system can distribute the workload equally among all of the active nodes. However, Rabbit still does not support the write workload so it cannot consider the cost of reflecting the updated data in a low gear.

Sierra [11] was designed as a power-proportional distributed storage system for general data centers where a replicated object store supports the write-and-read workloads during multiple-gear operations. Like Rabbit, Sierra is also a data placement for a power-proportional distributed file system by using the data replication. Sierra also organizes the replicas of the data set such that, each replica of the data set is stored in a group of nodes. The primary replicas are stored on a specific subset of nodes, called primary nodes. Then, the secondary replicas are stored on a different group of nodes and so on. Sierra differs from Rabbit that each replica of the data set is evenly distributed to all the nodes in each group. As a result, there is no constraint of the number of nodes in each group like in Rabbit.

We also previously presented an architecture known as NDCouplingHDFS [16], [17] to facilitate the efficient reflection of updated data in a power-proportional HDFS. NDCouplingHDFS focuses on coupled metadata management and data management on each HDFS node, which efficiently localizes the range of data maintained by the metadata. This method reduces the cost of managing the metadata generated during changes in the system configuration.

A system that integrates NDCouplingHDFS and Accordion has been proposed and developed [18]. Extensive empirical experiments using actual machines based on the HDFS demonstrated that the proposed system gains up to 22% better throughput-per-watt performance. However, the effect of block size to the data reflection process during gear-shifting has not well studied in our previous works.

There have been several works that focus on the specification of the servers and the access frequency of files to control the data placement [19]–[21]. The idea of above

power-proportional data placement methods can easily to be applied in such works.

3. Accordion data placement

The Accordion was designed to provide power proportionality and a high data I/O throughput in cluster file systems that use commodity computer servers such as the HDFS, the Google File System. In Accordion, the files are divided into a large number of blocks, and a number of replicas of each data block are distributed among the nodes of the cluster. Depends on the block size, the number of blocks for each file may change.

Like other approaches, Accordion aims to control the power consumption of the system by dividing the nodes in a cluster into several separate groups. A system that uses the Accordion data placement layout can then operate in a multiple-gear mode where each gear contains a different number of groups. The higher gears have more groups of nodes.

In Accordion, the nodes are arranged geometrically in a horizontal array because the nodes that belong to lower groups are bounded by the nodes of higher groups. $Group_a$ is higher than $Group_b$ if $a > b$. For example, we assume that a system is operating in a two-gear mode with two groups of nodes, i.e., $Group_1$ and $Group_2$. The nodes from $Group_1$ are activated in Gear 1, and the nodes from $Group_1$ and $Group_2$ are turned on in Gear 2. In this case, $Group_2$ is higher than $Group_1$. Thus, the nodes of $Group_1$ are bounded by the nodes of $Group_2$. Furthermore, each node is identified by a symbol Node $a-i$, Here, a is the identifier of the $Group_a$ and i is the order of the node this group. For instance, Node $3-1$ is the node number 1 of $Group_3$ and so on.

3.1 Skewed data replication in Accordion

The three goals of Accordion are to provide power proportionality in the read performance, reduce the cost of reflecting updated data when the system changes gear, and guarantee the data reliability in all file system operating modes. Thus, the below policies are applied to satisfy these goals.

3.1.1 Location of primary data

First, the primary data in the data set are distributed to all of the nodes in the system. Hence, each node stores the same amount of data.

3.1.2 Location of backup data

Starting with the highest $Group_G$, the data stored in this group are replicated to the next lower group $Group_{G-1}$. The process is finished when the backup data for the nodes in $Group_2$ are replicated to the nodes in $Group_1$, the lowest group.

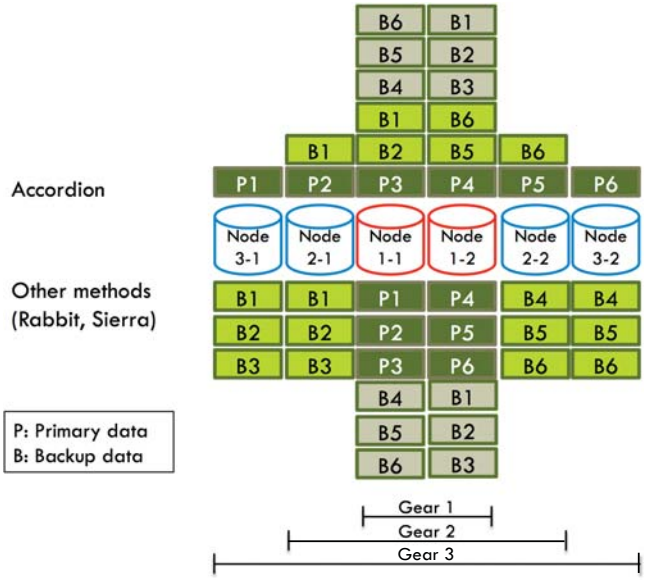

Fig. 1
AN EXAMPLE OF A THREE-GEAR SYSTEM BASED ON ACCORDION AND OTHER METHODS.

3.1.3 Chained de-clustering at the smallest group

Chained declustering has been proved to provide superior performance in the event of a failure while maintaining a very high degree of data availability [22]. As a result, to guarantee the data reliability in the lowest gear, the chained de-clustering policy is applied to the smallest group ($Group_1$). Each node replicates its data to its neighbor node, which guarantees that all of the data in the data set are replicated in the two neighbor nodes.

Figure 1 shows an example of a three-gear system based on Accordion and other data placement methods like Rabbit or Sierra. Here, in this figure, $Group_a$ contains the nodes of Node $a-i$ when $1 \leq a \leq 3$ and $i = 1, 2$. Gear 1 contains only the nodes in $Group_1$. To serve a request at Gear 2, the system requires the nodes in $Group_1$ and $Group_2$ to be activated, while Gear 3 requires all of the nodes in $Group_1$, $Group_2$, and $Group_3$ to be activated. In Accordion, the primary data are stored equally among all of the nodes. Next, the data in $Group_3$ are replicated in $Group_2$ and the data in $Group_2$ are replicated in $Group_1$. Finally, the chained declustering method is used to replicate the data in $Group_1$. However, in Rabbit or Sierra, the primary data are stored in $Group_1$ nodes, the secondary replicas are stored in $Group_2$ nodes and the tertiary replicas are stored in $Group_3$ nodes. It is easy to verify that the size of the data stored in the non-covering set nodes ($Group_2$ and $Group_3$) in Accordion is smaller than in Rabbit or Sierra. Hence, the amount of updated data when the system shifts gear in Accordion in smaller.

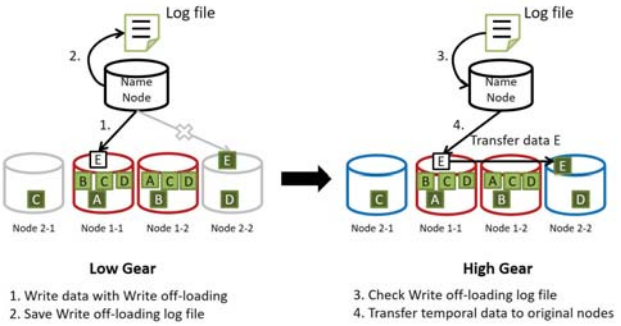

Fig. 2
REFLECTING UPDATED DATA USING WRITE OFF-LOADING.

The detail description of Accordion can be referred at [23].

3.2 Writing data by write off-loading

When a subset of a group of nodes is deactivated in a low gear, the system can accept write requests from clients using the write off-loading technique [24]. Next, when the system moves to a higher gear by reactivating nodes, it transfers the updated data internally in the background without stopping the processing of reading files from clients.

3.2.1 Writing new data using the write off-loading technique

Because the system operates in low gear, certain parts of the new data cannot be written to their corresponding deactivated nodes. Thus, the system selects another node randomly from the active nodes to serve this request. Information about the data, the temporary node, and the intended node are saved in a log file.

3.2.2 Reflecting updated data

When the system changes to high gear to serve a request for high-performance processing with the newly updated data set, it needs to perform two functions. The first function is to transfer the data written in temporary nodes to their actual intended nodes. It can be achieved by reading the information in the log files. The second function is to serve a request by scanning the new data set in the storage system. When there is no need to correct the data layout, compared with normal service in high gear, the cost of allowing the system to operate in multiple modes to save power will depend greatly on the quantity of data that needs to be transferred.

Figure 2 shows an example of writing new data at a low gear and reflecting updated data in a high gear. In this example, Node $2-2$ should serve the write request of data E due to the original data placement policy. However, it is powered off the log gear so the system decides to use an alternative node, i.e. Node $1-1$. The information of the written data and alternative node are recorded in the Log file. When the system moves to the High gear, it check the Log file and transfer the data E to the original node, i.e. Node $2-2$.

3.3 Fault tolerance

A specific algorithm to deal with node failures in a system is beyond the scope of this study. When a node fails, however, all of the nodes in the system are reactivated and the data from a failed node can be reconstructed based on the backup data. In a future study, we plan to provide a specific solution to this problem in more detail.

4. Experimental evaluation

In this part, we evaluate the proposed updating data reflection method with enabling serving read requests and efficiency of Accordion and Sierra relating to reflecting the updated data in multiple-gear systems in which the number of gears is high with variable block sizes.

4.1 Implementation

In this study, we implemented a prototype of the power-proportional distributed file system based on a modified HDFS [18]. During the updating data reflection during gear-shifting, we forward the requests to the nodes that contain the metadata structures of the low gear.

We changed the current class used to select locations of blocks via an interface where different data-layout policies can be performed. The write off-loading policy was implemented in the low power mode, which selects the temporal nodes for writing the new data instead of the currently inactivate nodes. To guarantee the data reliability, the Accordion writes block replicas to distinct nodes in all of the operation modes.

4.2 Experimental environments and method

We evaluated Accordion with Sierra in terms of their execution times needed to reflect updated data when the system moved from a low gear to a higher gear. The reason we choose Sierra as a comparative method is that Sierra and Accordion shares a common feature of flexible configuration of the multiple-gear systems, especially when the number of gear is high. In contrast, Rabbit requires very large numbers of nodes in higher gears because the number of nodes in each gear is determined by an exponential function.

The systems used in this experiment were operated using a six-gear configuration based on six groups of nodes. Each group contains four nodes. The numbers of active nodes from Gear 1 to Gear 6 were set to 4, 8, 12, 16, 20 and 24 consequently. There is another node to act as the NameNode of the HDFS. The data set contained 420 files in which the file's size was fixed to 64 MB. The block size used in HDFS

Table 1
SPECIFICATION OF A NODE

CPU	TM8600 1.0GHz
Memory	DRAM 4 GB
NIC	1000 Mb/s
HDD	500 GB
OS	Linux 3.0 64bit
Java	JDK-1.7.0

Table 2
EXPERIMENTAL ENVIRONMENT

Number of gears	6
Number of active nodes at each gear	4, 8, 12, 16, 20 and 24
Number of files	420
File size [MB]	64
HDFS version	0.20.2
Block size [MB]	16, 32 and 64

was varied to 16, 32 and 64 MB. Table 2 summarizes the above experiment settings. We focused on the energy-aware commodity system so we used low power consumption ASUS Eeebox EB1007 machines [25], the specifications for which are provided in Table 1.

The data set is written when the systems operated in Gear 2. In detail, the replicas of files that should be written to deactivated nodes would be temporally sent to other activating nodes. From the fault tolerance point of view, we guaranteed the constraint that the replicas of the same file are not written to the same node. Next, we consequently shifted the systems to higher gears, step by step from Gear 3 to Gear 6. During each shift, the temporal data were re-transferred from temporal nodes to original nodes that were reactivated according to each data placement method, Sierra or Accordion. The average execution times for reflecting the temporal data at each shift were measured and reported.

4.3 Experimental results

Figure 3 describes the average execution time needed to move the updated data at several configurations of both Sierra and Accordion. Here, Gear $2-3$ means the execution time for reflecting updated data when the system shifts from Gear 2 to Gear 3 and so on. From this figure, at each block size setting, it is seen that the measured execution times in Sierra were almost unchanged in all configurations. The reason is that the total amount of data needed for re-transferring were equal and large. Specifically, each group of nodes in Sierra stores the amount of the whole data set, i.e. 26.88 GB (420 of 64-MB-files). Hence, when the system moves to a next higher gear, the whole data set is re-transferred. As the total amount of re-transferring data was so large, although the number of blocks was different, Sierra did not receive any benefit from setting big block size, which means a smaller amount of blocks.

Additionally, it could also be derived that Accordion outperformed Sierra in all configurations as the total re-

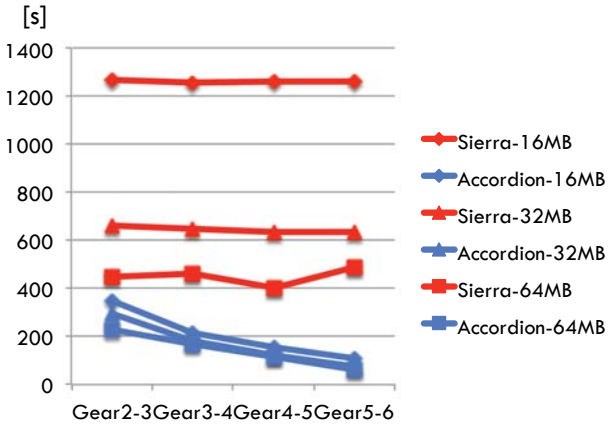

Fig. 3
EXECUTION TIME

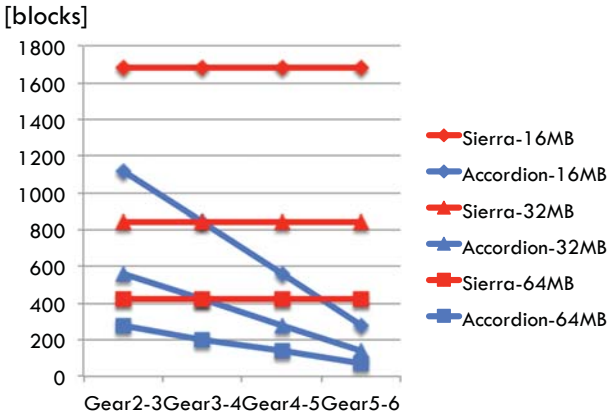

Fig. 4
NUMBER OF TRANSFERRED BLOCKS

transferred data when the system up to a next higher gear, only a part of the data set is need to be reallocated. As described in Section 3, a whole data set is equally stored in all nodes, only a small part of the data set is written in a group of nodes and is needed to be deallocated during each gear-shifting.

Furthermore, the effect of Accordion became bigger in the configurations in which the gear-shifting was performed at higher gears. Accordion gained the highest result in configuration Gear 5 to Gear 6, i.e. (Gear $5-6$), where it improved the execution time by almost 91% compared to Sierra when the block size was 16 MB i.e., 109.3 seconds in Accordion compared with $1,261.4$ seconds in Sierra.

By changing the value of the block size, we could evaluate the effect of the number of blocks to the performance of Accordion during gear-shifting. Figure 3 also describes that

in both Sierra and Accordion, the execution times were shorter with larger block size. The reason is mainly owing to the metadata management cost in HDFS. Larger block size, which leads to smaller the number of blocks in the HDFS, decreased the cost of the metadata management in our experiments. Figure 4 shows the number of transferred blocks for each configurations. In Accordion, 64 MB setting shortened the execution time at most 43% in Gear $5-6$ compared with 16 MB setting. The number of transferred blocks were 68 and 280 respectively.

Furthermore, the effects became larger in Sierra, as the 64 MB block size improved more than 60% in all configurations compared with 16 MB block size. The execution time of the 64 MB and 16 MB block size were 446.3 and $1,261.4$ seconds correspondingly. The reason is that the number of transferred blocks of the 16 MB setting was $1,680$ compared with 420 of the 64 MB setting.

In general, Accordion gains better power-proportionality performance than Rabbit or Sierra by approximately 20%. The reason is that because of the different design in the location of the primary data, the amount of re-transferring data during gear shifting of Accordion is significantly reduced. The detail of the power-proportionality results can be referred at [23].

5. Conclusion

Recently, energy-efficient infrastructures for Big Data processing have been gaining much attention from both industrial and academia as the power consumption at data centers all over the world are increasing. There are many data placement methods which have been proposed to address the issue of ineffective gear-shifting in multiple-gear power-proportional distributed file systems. However, the effect of block size in such file systems has not been studied yet.

This paper evaluated the effects of block size on Accordion and Sierra relating to updated data reflection during gear-shifting in a multiple-gear power-proportional distributed file system. From the empirical experiments, larger block size settings were found to improve the execution time for gear-shifting by more than 60% in Sierra and 43% in Accordion. Moreover, in all configurations, the gear-shifting execution time in Accordion is reduced by at most 91% compared with the time in Sierra.

In the future, we would like to study more about other perspectives relating to the multiple-gear distributed file systems, for example improving the QoS during gear-shifting.

References

[1] J. Whitney and P. Delforge, "Data Center Efficiency Assessment," in *Issue paper on NRDC (The Natural Resource Defense Council)*, 2014.

[2] M. Avgerinou, P. Bertoldi, and L. Castellazzi, "Trends in Data Centre Energy Consumption under the European Code of Conduct for Data Centre Energy Efficiency," *Energies*, vol. 10, no. 10, 2017.

[3] "Data Centre Energy Efficiency Benchmarking - Final Report," in *National Environment Agency (NEA)*, 2018.

[4] I. F. Sulaiman, "Emerging Indonesian Data Center Market and Energy Efficiency Opportunities," in *Technicle Report, Asian Development Bank*, 2017.

[5] D. Reinsel, J. Gantz, and J. Rydning, "Data Age 2025: The Evolution of Data to Life-critical - Don't Focus on Big Data; Focus on the Data That's Big," in *IDC White Paper*, April 2017.

[6] B. Battles, C. Belleville, S. Grabau, and J. Maurier, "NetApp White Paper: Reducing Data Center Power Consumption Through Efficient Storage," February 2007.

[7] J. Kim and D. Rotem, "Energy Proportionality for Disk Storage using Replication," in *Proceedings of the 14th International Conference on Extending Database Technology*, March 2011, pp. 81–92.

[8] L. A. Barroso and U. Hölzle, "The Case for Energy-proportional Computing," *Computer*, vol. 40, pp. 33–37, December 2007.

[9] A. Hrishikesh, C. James, G. Varun, G. Gregory R., K. Michael A., and S. Karsten, "Robust and Flexible Power-proportional Storage," in *Proceeding of the 1st ACM Symposium on Cloud Computing*, ser. SoCC '10, June 2010, pp. 217–228.

[10] H. H. Le, S. Hikida, and H. Yokota, "Efficient Gear-shifting for a Power-proportional Distributed Data-placement Method," in *Proc. 2013 IEEE International Conference on Big Data*. IEEE, October 2013, pp. 76–84.

[11] E. Thereska, A. Donnelly, and D. Narayanan, "SIERRA: Practical Power-proportionality for Data Center Storage," in *Proceedings of the 6th European Conference on Computer Systems*, ser. EuroSys '11. ACM, April 2011, pp. 169–182.

[12] W. Charles, O. Mathew, Q. Jin, W. A.-I. Andy, R. Peter, and K. Geoff, "PARAID: A Gear-Shifting Power-Aware RAID," *Transaction on Storage*, vol. 3, no. 3, pp. 13:1–13:33, 2007.

[13] B. Mao, D. Feng, H. Jiang, S. Wu, J. Chen, and L. Zeng, "GRAID: A Green RAID Storage Architecture with Improved Energy Efficiency and Reliability," in *Proc. IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems*, ser. MASCOTS '08, September 2008, pp. 1–8.

[14] J. Leverich and C. Kozyrakis, "On the Energy (In)efficiency of Hadoop Clusters," *SIGOPS Operating System Review*, vol. 44, pp. 61–65, 2010.

[15] R. T. Kaushik and B. Milind, "GreenHDFS: Towards an Energy-Conserving, Storage-Efficient, Hybrid Hadoop Compute Cluster," in *Proceedings of the 2010 International Conference on Power Aware Computing and Systems*, ser. HotPower'10, October 2010, pp. 1–9.

[16] H. H. Le, S. Hikida, and H. Yokota, "NameNode and DataNode Coupling for Power-proportional Hadoop Distributed File System," in *Proc. the 18th International Conference on Database System for Advanced Applications, Part II*, ser. DASFAA '13, vol. 7826. Springer Verlag, April 2013, pp. 99–107.

[17] H. H. Le, S. Hikida, and H. Yokota, "NDCouplingHDFS: A Coupling Architecture for a Power-proportional Hadoop Distributed File System," *IEICE Transactions on Information and Systems*, vol. E97-D, no. 2, pp. 213–222, 2014.

[18] H. H. Le, S. Hikida, and H. Yokota, "An Efficient Gear-shifting Power-proportional Distributed File System," in *Proc. the 26th International Conference on Database and Expert Systems Applications*, ser. DEXA '15, September 2015, pp. 153–161.

[19] R. Xiong, J. Luo, and F. Dong, "Optimizing Data Placement in Heterogeneous Hadoop Clusters," *Cluster Computing*, vol. 18, no. 4, pp. 1465–1480, 2015.

[20] Y. Lin and H. Shen, "Eafr: An Energy-efficient Adaptive File Replication System in Data-intensive Clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1017–1030, 2017.

[21] X. T. Tran, T. Van Do, C. Rotter, and D. Hwang, "A New Data Layout Scheme for Energy-efficient MapReduce Processing Tasks," *Journal of Grid Computing*, pp. 1–14, 2018.

[22] H. Hsiao and D. DeWitt, "Chained Declustering: A New Availability Strategy for Multiprocessor Database Machines," in *Proc. 6th International Conference on Data Engineering*. IEEE, February 1990, pp. 456–465.

[23] H. H. Le, S. Hikida, and H. Yokota, "Accordion: An Efficient Gear-shifting for a Power-proportional Distributed Data-placement Method," *IEICE Transactions on Information and Systems*, vol. E98-D, no. 5, pp. 1013–1026, 2015.

[24] D. Narayanan, A. Donnelly, and A. Rowstron, "Write Off-loading: Practical Power Management for Enterprise Storage," *ACM Transaction on Storage*, vol. 4, no. 3, pp. 10:1–10:23, 2008.

[25] "EeeBox PCs - EeeBox PC EB1007 - ASUS," http://www.asus.com/EeeBox_PCs/EeeBox_PC_EB1007/specifications/.