International Journal of Control and Automation
Vol. 11, No. 4 (2018), pp.197-206
http//dx.doi.org/10.14257/ijca.2018.11.4.18

SSD Garbage Collection Detection and Management with Machine Learning Algorithm

Jung Kyu Park[1] and Jaeho Kim[2*]

[1]Department of Computer Software Engineering, Changshin University, 51352, Korea
[2]Department of Electrical and Computer Engineering at Virginia Polytechnic Institute and State University, 24061, USA
smartjkpark@gmail.com, kjhnet@gmail.com

Abstract

The garbage collection (GC) task of removing invalid pages due to insufficient free space on the flash memory chip inside the SSD has a significant impact on SSD performance. This GC operation is performed by the FTL, which is an internal firmware of the SSD, so that it is impossible to know whether the SSD is garbage collected at the operating system level. In this paper, we attempt to manage GC overhead at the operating system level. In our approach, first, we use a machine learning technique to devise a GC detecting mechanism at the operating system level, and second, we show that by making use of this mechanism performance variance normally observed on SSDs can be reduced. We develop a GC-detector that detects garbage collection of SSDs and request TRIM operations to the SSD when GC is detected. Experimental results running the GC-detector show increase average bandwidth and low performance variance compared to when not using GC-detector.

Keywords: *Garbage Collection, Machine Learning, SSD, TRIM*

1. Introduction

Unlike hard disks, SSDs based on NAND flash memories are widely used in PCs and mobile phones due to the fact that the physical movement of the head is unnecessary and the input/output is fast and consumes less electricity. SSD has the above advantages, but it is limited in data center because of higher price than hard disk. However, since the MLC and TLC technologies, which store several bits in one cell, which is the smallest unit for storing data of the flash memory, have been developed and the price has been lowered, the usage rate is gradually increasing in the data center.

These SSDs cannot be overwritten, and the Flash Translation Layer (FTL) is required for compatibility with the characteristics of NAND flash memory that can perform erase operations on a block-by-block basis and the conventional block level interface. In order to provide compatibility with the block level interface, the FTL performs firmware management such as mapping table management, wear leveling, and so on. NAND flash memory has a feature that cannot be overwritten, and it needs “erase before writing” operation to update the data. This “erase before write” operation causes a lot of overhead, and in order to reduce this overhead, FTL manages the mapping table to reduce this overhead. In this process, an invalid page is created. When invalid pages are piled up and the free space in the flash memory chip becomes insufficient, it is necessary to rearrange the invalid pages and revert to the valid page in order to secure the storage space. This process is called garbage collection.

Received (December 28, 2017), Review Result (February 23, 2018), Accepted (March 10, 2018)
[*] Corresponding Author

Copyright © 2018 SERSC Australia

The garbage collection task takes a lot of overhead and the SSD does not perform I/O operations properly, which reduces I/O speed. Figure 1 shows the SSD with continuous writes and bandwidth measurements. Figure 1 shows the 4 bandwidth reduction due to garbage collection. At the operating system level, when garbage collection occurs as shown in Figure 1, it experiences a sudden drop in bandwidth. These results make it difficult to provide QoS (Quality of Service) in the data center and cause problems such as the inability to predict the real-time system response time [1-3].

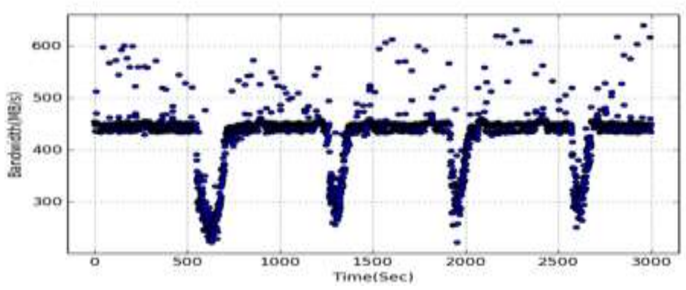


Figure 1. Rapid Bandwidth Reduction Due to Garbage Collection

In order to solve the problem of garbage collection, studies have been conducted such as reducing the influence of garbage collection by adjusting the over-provisioning space, which is a reserved space allocated to the SSD, and improving the garbage collection algorithm [1-3]. However, such a study is an attempt to make garbage collection efficient, and at the operating system level, it does not address the abrupt bandwidth reduction that occurs when garbage collection occurs. Therefore, it cannot be said that it solves the problem of garbage collection completely.

This paper presents a method to reduce the impact on the operating system when garbage collection occurs. To solve this problem, we first detected the garbage collection task of the SSD at the operating system level. Since the garbage collection operation of the SSD is performed by the SSD firmware, it is impossible to know accurately whether garbage collection occurs at the operating system level. Therefore, we used a method of predicting whether to perform garbage collection by analyzing the state information data of the SSD that can be measured at the operating system level through a machine learning algorithm. Machine learning algorithm is a method to obtain information through data. In this study, we analyze the state information when garbage collection occurs and the state information when garbage collection does not occur through machine learning algorithm, and finally obtains information on garbage collection. In this paper, we use decision trees among machine learning algorithms to predict the garbage collection of SSDs using C4.5 algorithm with fast prediction speed and low overhead [11, 12].

This paper is composed as follows. Chapter II, Background, discusses garbage collection and machine learning algorithms. In Chapter III, the state information of the SSD is collected and analyzed by the machine learning algorithm. In Chapter IV describes the garbage collection detection for multiple SSDs, and the garbage collection detector for sending TRIM jobs to the SSD for which garbage collection has been detected. In Chapter V, we examine whether the garbage collection effect is reduced when the garbage collection detector is applied. Finally, Chapter VI concludes the paper.

2. Related Works

2.1. Garbage Collection

SSDs require FTL for NAND flash memory characteristics and compatibility with traditional block level interfaces. The NAND flash memory consists of a block of pages and several pages. In addition, NAND flash memory cannot overwrite data, and read / write operations are performed in page units, and erase operations are performed in block units. Both of these properties result in a large overhead when updating the page. For example, to write different data to a page that stores valid data, you need to: First, the page erase operation must be performed because overwriting is not possible [1, 2].

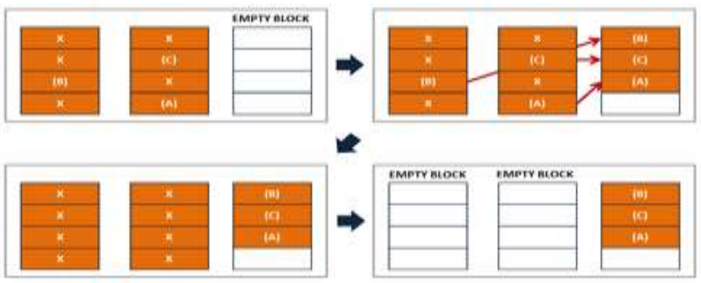


Figure 2. Rapid Bandwidth Reduction due to Garbage Collection

However, since the erase operation is performed on a block-by-block basis, it is necessary to copy the valid pages in the block to another block, update the page, and copy the copied page again. To solve this problem, the FTL uses a mapping table. When a specific page is updated, after writing data to another page, the existing page is invalidated and the mapping table is updated to reduce overhead due to the update. If these operations continue, invalid pages will accumulate and the space inside the flash memory chip will become insufficient. When data is written to the flash memory chip in order to regain storage space, when the free space falls below a certain level, garbage collection operation for collecting invalid pages occurs. Figure 2 shows the garbage collection operation proceeds as follows.

(1) Select the victim block where garbage collection will occur.

(2) After selecting the block to move the valid page of the victim block, copies the page to the selected block.

(3) Perform erase operation on the victim block.

Such an operation causes a large overhead in the SSD, resulting in a reduction in the bandwidth of the write operation. This garbage collection operation will continue to occur from the time garbage collection occurs, since the garbage collection operation recovers invalid pages as much as necessary when a write operation is performed. Therefore, the bandwidth of the write operation is continuously decreased from the time of garbage collection at the operating system level. This problem can be solved by executing a TRIM command to secure storage space inside the SSD [3].

2.2. Machine Learning

Machine learning algorithms are the task of obtaining concrete information through data [4-6]. The machine learning algorithm proceeds in the following order.

(1) Collecting learning data

(2) Algorithm training phase

(3) Algorithm test step

(4) Application

Machine learning algorithms include instructional learning and non-learning learning. Map learning is a method using labeled data. The collected data, such as the e-mail spam filtering system above, indicates whether or not the collected data is spam mail. Algorithms for learning maps include decision trees, Naïve Bayes, and Neural networks. Decision tree method is the most used technique [6].

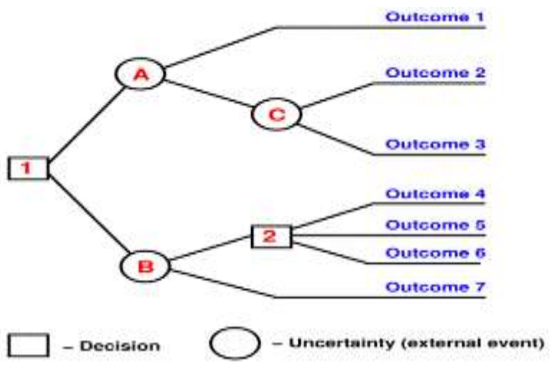


Figure 3. Example of Decision Tree

The decision tree method constructs a decision tree through algorithm training with learning data and classifies the data using a tree. Figure 3 is an example of a simple decision tree that can be used for spam classification. The decision tree method has an advantage that the classification result is easy to understand, the decision tree is applied after the decision tree is generated, and the classification speed is fast because there are not many operations. In this study, C4.5 algorithm is used to analyze garbage collection situation and non-garbage collection situation [11-13].

Unsupervised learning uses unclassified learning data as opposed to map learning. Clustering is one of the types of learning non-visuality. It generates new information by judging similarity of data and grouping them. There is a K-Means algorithm that classifies data into k similar clusters with a representative algorithm [12]. Although the K-Means algorithm can classify data based on the similarity of unclassified data, it has a disadvantage that the classification time is long due to a large number of operations because it is necessary to obtain an average among respective data at the time of classification.

3. Garbage Collection Detection

This chapter describes how to detect garbage collection of an SSD using a machine learning algorithm. To detect garbage collection, we used the C4.5 algorithm of decision trees. The order of these operations is very similar to the example of the email spam filtering system mentioned in 2.2 Machine Learning Algorithm. First, the SSD information that can be collected at the operating system level is collected as learning data and the algorithm training step is performed to train the algorithm using the collected learning data. In the algorithm test step, the accuracy of the classification method obtained through the algorithm training step was evaluated. Finally, the garbage collection sensor was developed and the results of the machine learning algorithm were applied.

3.1. Learning Data Collection

In order to collect the training data to train the algorithm, the disk state attributes appearing in the operating system are collected while the disk is being written. FIO and Filebench benchmark programs were used to generate SSD write requests [7, 8]. The FIO benchmark was used to generate persistent write requests with the Micro benchmark. The 512K block random write was performed using the FIO benchmark. We also used the Macro benchmark Filebench to generate a workload similar to the actual situation. We used the fileserver and varmail workloads where filebench write requests were most common. Two benchmark tools, Filebench and FIO, were used to collect data for various situations.

iostat was used to collect SSD status data in the event of a write request [9]. *iostat* is a disk monitoring tool that measures various items such as storage input/output statistics in real time. *iostat* provides a total of 17 attributes. We collected a total of 17 attributes in 1 second and then included the standard deviation of 50 cumulative values of each attribute in the data. We removed unnecessary attributes among the 34 attributes collected during this process.

First, the attributes related to CPU represent the CPU usage related to IO in the whole system. This is not an attribute value related to each SSD, and since the SSD used in the same system has the same value, it is not suitable for finding out whether each SSD is garbage collected and is not included in the data. We also removed the property of reading and writing bandwidth. The read operation does not affect the garbage collection, and the bandwidth of the write operation is also affected by external factors such as block size and does not represent the garbage collection situation. However, the standard deviation of the bandwidth variation of the write operation was included in the data.

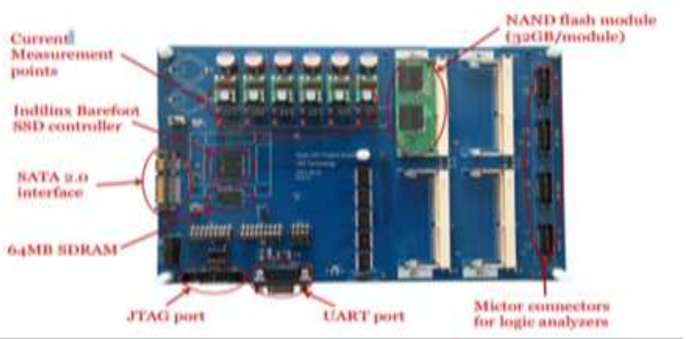

Current Measurement points
Indilinx Barefoot SSD controller
SATA 2.0 interface
64MB SDRAM
JTAG port
UART port
NAND flash module (32GB/module)
Mictor connectors for logic analyzers

Figure 4. Architecture of OpenSSD

The data collected through the above method are applied to the learning of the machine learning algorithm. Map learning classifies whether the data is *A* state or *B* state when *X* data is input by learning algorithm by providing data on *A* state and *B* state. Since the problem of SSD garbage collection detection is consistent with the above situation, map learning is suitable for solving this problem. Therefore, the collected data was labeled as the collected data under certain conditions (GC or non-GC). However, at the operating system level, it was not possible to accurately determine whether garbage collection was being performed using a commercial SSD. We use the *OpenSSD* to solve this problem [10].

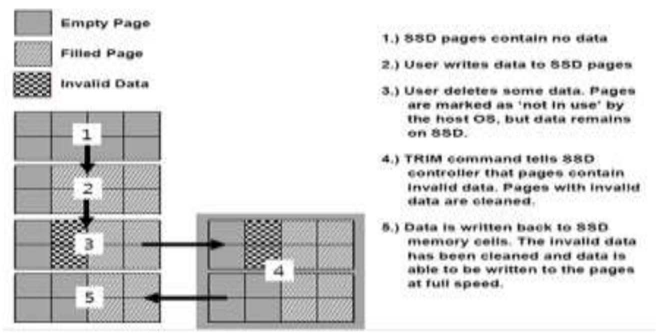
Figure 4 shows the *OpenSSD* that is an experimental board that can be directly developed and applied to SSD firmware. *OpenSSD's* FTL can be modified and various status information of the SSD can be checked at the operating system level. We modified the FTL and applied it to *OpenSSD* to output the debugging code when garbage collection occurred. This allows you to verify that garbage collection has been performed at the operating system level when the debugging code is output. This information allowed us to categorize the state of the data (GC or non-GC) collected via *iostat*. *OpenSSD* is connected to the server via a UART cable and the modified FTL sends the debugging code over the cable when garbage collection occurs.

3.2. Algorithm Training and Testing

In order to analyze the data collected in 3.1, C4.5 algorithm was applied to decision trees during machine learning algorithm learning [11]. The C4.5 algorithm is one of the top 10 algorithms of data mining and has the advantage of being able to understand the result quickly [6].

As a result of analyzing the collected data, it was found that the standard deviation value of *svctm* among the *N* items classifies the situation where garbage collection occurs and the situation where it does not. *svctm* represents the average processing time of I/O requests being processed by the device. The standard deviation of *svctm* increases when garbage collection occurs. It seems that the deviation of *svctm* in the processing of garbage collection and write requests in SSD is increased.

In order to measure the accuracy of the trained algorithm, we collected the data in the same way and applied the tree generated in the above procedure. As a result, the accuracy was 99.66%. We also measured data using the FIO benchmark in SSD to see if it applies to commercial SSDs. Figure 5 shows that garbage collection is detected at the point of time when the bandwidth decreases.



1.) SSD pages contain no data
2.) User writes data to SSD pages
3.) User deletes some data. Pages are marked as 'not in use' by the host OS, but data remains on SSD.
4.) TRIM command tells SSD controller that pages contain invalid data. Pages with invalid data are cleaned.
5.) Data is written back to SSD memory cells. The invalid data has been cleaned and data is able to be written to the pages at full speed.

Figure 5. Result of Garbage Collection Detection on Commercial SSD

4. SSD Garbage Collection Detector

Result of Garbage collection detection on commercial SSD. In this chapter, machine learning is applied to the garbage collection to produce a garbage collection detector. The Garbage Collection Detector is built with Python and detects whether all SSDs attached to the system are garbage-collected in real time. If there is a detected SSD, TRIM command is issued to secure the storage space inside the SSD.

We used open source *pyC45* to apply the C4.5 algorithm to the garbage collection detector [13]. *pyC45* constructs XML decision trees through training data during training. Next, we classify the data using XML decision tree in the algorithm test. The garbage collection detector constructs a decision tree of each SSD connected to the server through a learning process. Then, the *svctm* value of each SSD is measured every second, and the standard deviation value of the last 50 values is applied to the decision tree to judge whether or not the SSD is garbage collected. If an SSD that is judged to be garbage collection is detected, it requests a *TRIM* command manually to the corresponding SSD.

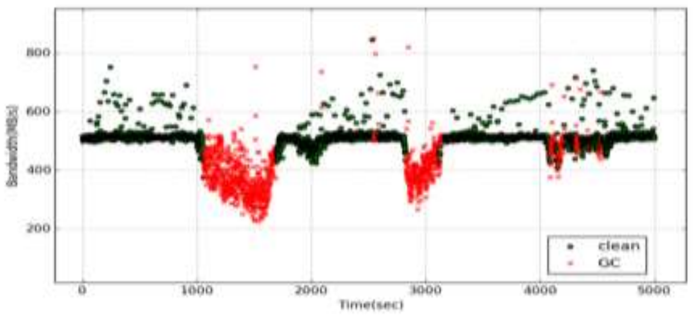


Figure 6. Execution Process of TRIM Command

Copyright © 2018 SERSC Australia

The TRIM command is a command that sends information about files deleted from the operating system to the SSD. When the SSD needs to secure the storage space inside the flash memory, the data deleted from the operating system is actually deleted from the flash memory, thereby securing the internal storage space of the flash memory [11]. The detailed procedure is shown in Figure 6. As a result of the TRIM operation, the SSD will free up storage space and will no longer perform garbage collection operations and will have normal I/O speed.

SSDs require FTL for NAND flash memory characteristics and compatibility with traditional block level interfaces. The NAND flash memory consists of a block

5. Experiment and Evaluation

In Chapter III, the garbage collection of the SSD is detected by the machine learning algorithm. In Chapter IV, we implemented a garbage collection detector that collects information on all SSDs connected to the server, detects garbage collection in real time, and issues a TRIM command to the SSD that is being garbage collected. In this chapter, we have experimented and evaluated whether the garbage collection effect was reduced when the garbage collection detector was applied.

The experiment was performed in the Linux kernel 3.10.4 environment. The SSD used in the experiment was Samsung 840 pro 128GB model. To collect learning data, we executed a random write workload of the FIO benchmark and a fileserver workload of the Filebench benchmark, *varmail*, to collect a total of 16,118 data. Using the collected learning data, we created an XML-like decision tree through the learning process of pyC45 and made it available to the garbage collection detector.

Experiments were conducted to measure the effects of garbage collection detectors. After creating four partitions with 10GB, 20GB, 30GB, and 68GB capacities in the SSD, We set up an automatic TRIM that allows the SSD to perform TRIMs during idle times. Then, using the FIO benchmark's random write workload on a 68GB partition, we filled in the partition and ran the capacity using the *rm* command. Partitions with the remaining 10GB, 20GB, and 30GB capacities were used to write workloads using the *iozone* benchmark. In the above experiment environment, we created 4 different size partitions and then repeatedly perform the write and erase operations so that the SSD has the minimum idle time.

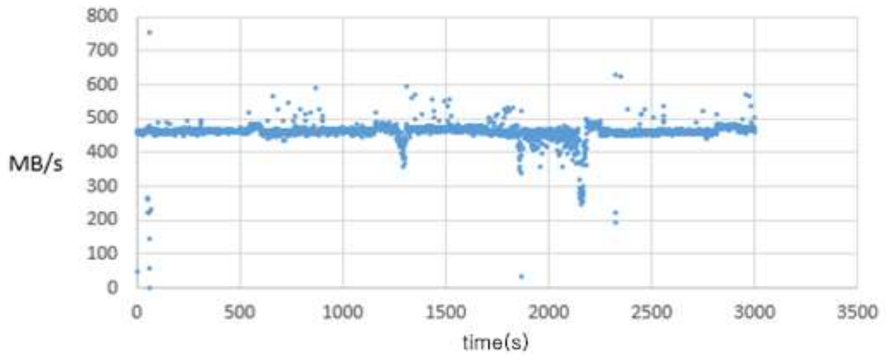


Figure 7. Before Applying the Garbage Collection Detector

This simulated the environment in which I/O requests come into the SSD from time to time, like the data center. The first experiment measured the bandwidth of the SSD for 3000 seconds in an environment that did not perform the garbage collection detector, and Figure 7 shows the result of the first experiment. In the first experiment, the average bandwidth of the SSD was 449MB/s and two bandwidth reductions were observed. This is due to garbage collection, reduced bandwidth, and automatic TRIM setting, so SSD bandwidth is expected to rise again due to the operating system executing the TRIM command during idle time. The number of times the bandwidth has dropped below 400MB/s is 137, which means that it has been affected by garbage collection for at least 137 seconds for 3000 seconds. As a result, it was measured that the input/output is performed on the SSD even when the automatic TRIM is set, and the effect of the garbage collection is obtained when there is no idle time.

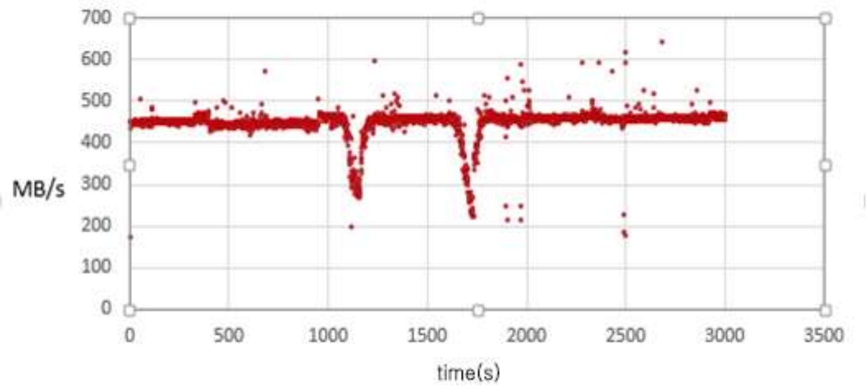


Figure 8. After Applying the Garbage Collection Detector

The second experiment measured the bandwidth of the SSD for 3000 seconds in an environment that performed a garbage collection detector. As a result, the bandwidth as shown in Figure 8 was measured. In the second experiment, the average bandwidth of SSD was 462MB/s, which was 13MB/s higher than the first experiment. The second experiment also showed two bandwidth reduction phenomena like the first experiment. However, in the first experiment, the time with a bandwidth of less than 400MB/s is 137 seconds, whereas in the second experiment, the time with a bandwidth of 400MB/s or less is measured as 66 seconds. It can be concluded that the garbage collection detector reduces the impact of garbage collection by receiving less than 52% garbage collection effects than when the first garbage collection detector was not run.

6. Conclusion

In this paper, we have detected garbage collection of SSDs using SSD state information and machine learning algorithms that can be observed at the operating system level. Also, the TRIM command is executed on the detected SSD to secure the storage space of the SSD, thereby reducing the influence of the garbage collection. As a result, bandwidth reduction caused by garbage collection is mitigated and average bandwidth of SSD is increased.

References

[1] N. Shahidi, M. T. Kandemir, M. Arjomand, C. R. Das, M, Jung, and A. Sivasubramaniam, “Exploring the Potentials of Parallel Garbage Collection in SSDs for Enterprise Storage Systems”, in Proc. of the International Conference for High Performance Computing, Networking, Storage and Analysis, UT, USA, **(2016)** November 13-18.

[2] M. Lin, and Z. Yao, “Dynamic garbage collection scheme based on past update times for NAND flash-based consumer electronics”, IEEE Transactions on Consumer Electronics, vol. 61, no. 4, **(2015)**, pp. 478-483.

[3] J. Kim, C. Lee, S. Lee, I. Son, J. Choi, S. Yoon, H. Lee, S. Kang, Y, Won, and J. Cha. “Deduplication in SSDs: Model and Quantitative Analysis”, in Proc. of 29th IEEE Symposium on Massive Storage Systems and Technologies (MSST), CA, USA, **(2013)**, May 6-10.

[4] T. O. Iwasaki, S. Ning, H. Yamazawa, C. Sun, S. Tanakamaru, and K. Takeuchi, "Machine Learning Prediction for 13X Endurance Enhancement in ReRAM SSD System", in Proc. of 2015 IEEE International Memory Workshop (IMW), CA, USA, **(2015)**, May 17-20.

[5] P. Harrington, “Machine Learning in Action”, Manning Publications Company, New York, **(2012)**.

[6] X. Wu, and et al, “Top 10 Algorithms in Data Mining”, Journal of Knowledge and Information Systems, vol. 14, no. 1, **(2007)**, pp. 1-37.
[7] FIO [Internet], Available: http://linux.die.net/man/1/fio/.
[8] Filebench [Internet], Available: http://linux.die.net/man/1/filebench/.
[9] FIO [Internet], Available: http://linux.die.net/man/1/iostat/.
[10] The OpenSSD Project [Internet], Available: http://www.openssd-project.org/
[11] J. Quinlan, “C4.5: Programs for Machine learning”, Morgan Kaufmann, San Mateo, **(1992)**.
[12] S. Moedjiono, Y, R. Isak, and A. Kusdaryono, “Customer loyalty prediction in multimedia Service Provider Company with K-Means segmentation and C4.5 algorithm”, in Proc. of International Conference on Informatics and Computing (ICIC), Mataram, Indonesia, **(2016)**, October 28-29.
[13] M. K. Pakhiram, “A Fast K-Means Algorithm Using Cluster Shifting To Produce Compact And Separate Clusters”, International Journal of Engineering, Transaction A: Basics, vol. 28, no. 1, **(2015),** pp. 35-43.

Authors

Jung Kyu Park, received the M.S. and Ph.D. degrees in computer engineering from Hongik University, Seoul, Korea, in 2002 and 2013, respectively. He has been a research professor at the Dankook University since 2014. From 2016 to 2017, he was a visiting professor at Department of Digital Media Design and Applications, Seoul Women’s University. In 2018, he joined the assistant professor of Department of Computer Software Engineering, Changshin University. His research interests include operating system, new memory, embedded system and robotics theory and its application.

Jaeho Kim, received the BS degree in information and communications engineering from Inje University, Gimhae, Korea, in 2004, and the MS and PhD degrees in computer science from the University of Seoul, Seoul, Korea, in 2009 and 2015, respectively. He is currently a postdoctoral researcher in the Department of Electrical and Computer Engineering at Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg in Virginia, US. His research interests include storage systems, operating systems, and computer architecture.

Copyright © 2018 SERSC Australia