A Technique for decreasing the SSD's GarbageCollection overhead using ML techniques

Ms. Hepi Suthar[1, a)] and Dr. Priyanka Sharma[2, b)]

[1]*School of Information Technology, AI & Cyber Security Department at Rashtriya Raksha University Gandhinagar, Gujarat, India*

[2]*Department of Research & Publications, Rashtriya Raksha University Gandhinagar, Gujarat, India*

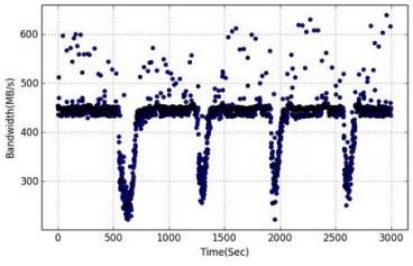
Abstract: In this article, we try to control Garbage Collection over head at the OS level in this work. In our this work method, we first develop a Garbage Collection detecting mechanism at the OS level using a ML technique, and then we demonstrate how using this mechanism may minimize performance variation often found on SSDs. We created Garbage Collection detector that looks for SSD Garbage Collection and asks TRIM operations when it does. When employing the Garbage Collection detector, experimental findings demonstrate higher average bandwidth and reduced performance variation.

Keywords: NAND Flash, SSD, TRIM, Garbage Collection,ML Algorithm

INTRODUCTION

In contrast to hard disks drives, which require the head's actual movement, SSDs using NAND flash memory are utilized extensively in computers and smartphones since the input and output is quick & uses less electricity. Although SSD provides the aforementioned benefits, its use in data centers is constrained by its greater cost than hard disk [4]. The rate of utilization in the data center is, however, progressively rising as a result of the development and price reduction of the technologies of Multi-layer cell and Triple Layer Cell, which store multiple bits per cell, the lowest storage unit for data in the NAND flash memory [1] [12]. These SSDs cannot be overwritten, therefore the FTL is necessary to work with NAND flash memory's abilities to do block-by-block erase operations as well as the traditional interface at the block level. The Flash Translation Layer manages firmware to be able to enable compliance with the interface at the block level, including wear levelling, mappingtable maintenance, and other tasks. NAND flash memory includes a characteristic that prevents overwriting, therefore updating the data requires a "erase before writing" action [2]. This erase prior write action generates a significant amount of overhead, which FTL handles by managing the mapping table. An invalid page is produced during this operation. Rearranging the erroneous pages and switching back the appropriate page is important to protect the storage space when the amount of in the flash memory chip's available area runs out. Garbage collection is what we call this. The SSD performs I/O operations improperly and consumes a significant amount of overhead, which decreases Fig. 1. Garbage Collection I/O speed causes a rapid loss in bandwidth.

Fig. 1. Garbage collection causes a rapid bandwidth drop



The SSD is displayed in Figure 1 together with bandwidth measurements and continuous writing. The 4 bandwidth decreases brought on by garbage collection are shown in Figure 1. When GC happens, as in Fig. 1, the OS's bandwidth suddenly decreases. As a result, it is challenging to deliver QoS in the data center, which leads to issues including the failure to forecast system response times in real time [5], [6], and [7]. Studies have been done on optimizing the garbage collection algorithm and lowering the impact of GC by modifying the over-provisioning area, a designated area assigned to the Solid state Drive [5] [6] [7]. Such a research, however, focuses on improving the efficiency of garbage collection rather than addressing the sudden bandwidth drop brought on by garbage collection at the OS level. As a result, it cannot be argued that the issue of garbage collection is entirely resolved.

Fig. 2. Garbage collection procedure

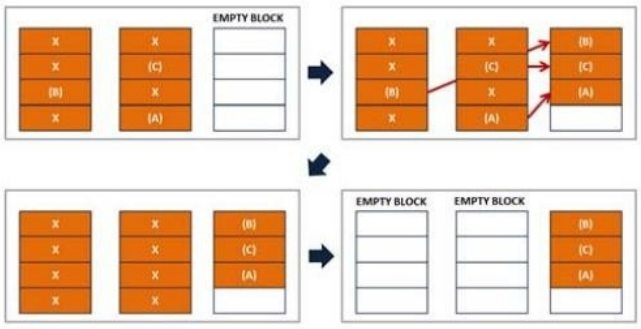
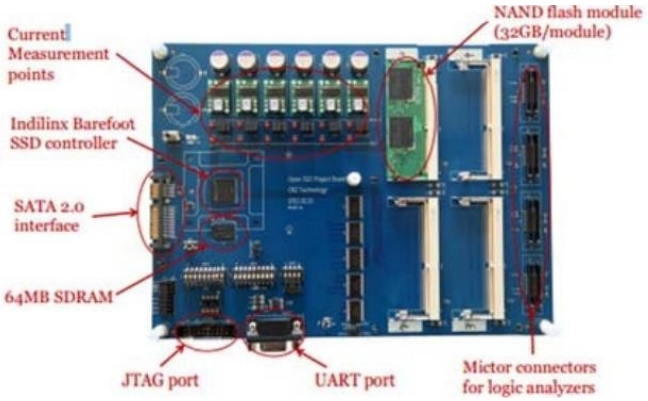


Fig. 3. OpenSSD Objects Model [20]



This document outlines a technique to lessen the effect garbage collection has on the OS. In order to address this issue, we first identified the SSD's Garbage Collection job at the OS level. It is hard to determine with accuracy whether garbage collection occurs at the OS level since the SSD's garbage collection function is carried out by the SSD firmware. In order to determine whether to undertake garbage collection, we thus utilized an approach that involved analyzing SSD status information data that may monitored at the OS level using an ML algorithm. An ML algorithm is a technique for learning from data. In this work, we used a machine learning method to analyze, the information on when waste collection does place and the information on when it does not ultimately obtaining information on GC [9].

In this article study, we apply decision trees among ML methods to forecast the SSD garbage collection utilizing the rapid prediction speed and low overhead C4.5 algorithm [11,12]. The structure of this essay is as follows. Machine learning methods and garbage collection are covered in background section. The SSD's state information is gathered and analyzed by the ML algorithm in Section 3 [8]. The GC detector for delivering TRIM Function tasks to the Solid state drive for which GC has been identified is described in Section 4 together with the Garbage Collection detection for multiple SSDs [3]. In Section 5, we investigate if using the GC detector lessens the effect of garbage collection.

METHEDOLOGY

GARBAGE COLLECTION

NAND flash memory properties and compatibility with conventional block level interfaces necessitate FTL for SSDs. There are numerous pages and a block of pages in the NANDflash memory. NAND flash memory also has limitations on data overwriting, performs read/write operations in page units, and performs erase operations in block units. The overhead for updating the page is significant due to both of these characteristics. For instance, you must first conduct the page wipe operation since overwriting is not feasible in orderto copy alternative data to a page that holds valid data [1, 2].Since the erase operation is carried out block-by-block, it is essential to duplicate the block's valid pages to another block, update the copied page, and then copy the updated copied page once again. The FTL makes use of a mapping table to resolve this issue. When a specific page is modified, the mapping table is updated to decrease cost associated with the update and the existing page is invalidated after sending datato another page. If these actions continue, invalid pages will build up and the flash memory chip's available space will runout. When data is stored to a flash memory chip to reclaim storage space, garbage collection operations for collecting invalid pages start when the free space falls below a certain threshold. The process of garbage collection is depicted in Figure 2 as follows. (1) Decide which victim block will haverubbish collected there. (2) Copies the page to the chosen block after choosing the block to which the victim block should be moved. 3) Run an erase operation on the block that was the victim. The write operation's bandwidth is decreased as a result of this operation's significant overhead on the SSD. Because garbage collection recovers invalid pages as much as possible when a write operation is conducted, this garbage collection procedure will continue to occur from the momentgarbage collection begins. As a result, Beginning with garbage removal at the operating system level, the write operation's bandwidth is gradually decreased. Running a TRIM command to secure storage space on the SSD will fix this problem [3].

MACHINE LEARNING

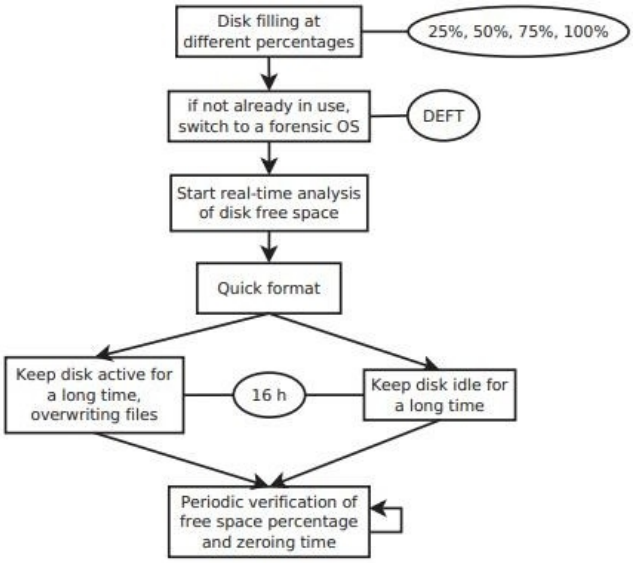
Finding specific information from data is the work of machine learning algorithms [4-6]. The following is the sequence in which the machine learning algorithm works. (1) Gathering learning data; (2) training the algorithm; and (3) testing the algorithm (4) Application Instructional learning and non-learning learning are both incorporated into machine learning algorithms. A technique employing labelled data is map learning. The acquired data reveals whether or not the collected data is spam mail, much as the aforementioned email spam filtering system. Decision trees, naive bayes, andneural networks are examples of algorithms for learningmaps. The most popular strategy is the decision tree method [6].

The decision tree approach uses a decision tree to classify the data after training an algorithm using learning data. A straightforward decision tree that may be utilized for spam categorization is shown in Figure 3. The decision tree approach has the advantages of being used once the decision tree is formed, the classification speed being quick due to the lack of procedures, and the classification result being simple to grasp. The C4.5 technique is utilized in this work to assess both garbage collection and non-garbage collection scenarios[11–13]. In contrast to map learning, unsupervised learning employs unclassified learning data. One of the categories of non-visual learning is clustering. By classifying the data anddetermining how similar they are, it produces new information. K-Means is an algorithm that uses a representative method to divide data into k groups that are comparable to one another [12]. The K-Means technique may categories data based on how similar unclassified data is, butit has the drawback of taking a long time to classify because of the numerous operations required to average out the relevant data at the time of classification.

DETECTION OF GARBAGE COLLECTION

This article explains how to use a ML technique to find garbage collection (GC) on an SSD. We applied the decision tree C4.5 method to find garbage collection. These activities are performed in a manner that is quite similar to the spam screening mechanism for email example from section 1.2 ML Algorithm. The OS-level SSD data that may be gathered is first gathered as learning data, and after that, the algorithm istrained using the learning data that was gathered [10]. The algorithm test phase evaluated the classification method's accuracy based on the results from the algorithm training stage. In the end, the GC sensor was created and the ML algorithm's findings were used.

Fig 4. Garbage collection test flow.



learning how to collect data The OS's disk state characteristics are gathered when the disk is being written toin order to gather the training data needed to train the algorithm. The benchmark programs FIO and File-benchwere used to produce Solid state drive write requests [7], [8]. Persistent write requests were generated using the Micro benchmark using the FIO benchmark. The FIO benchmark was utilized to complete the 512K block random write [17]. In order to create a workload that was representative of the existing circumstances, we also employed the macro benchmark File-bench. Where file write requests were most often made, we employed the fileserver and varmailworkloads. To gather information for various scenarios, File-bench and FIO, two benchmarking programs, were employed [18]. Data about SSD state was gathered using Iostat if a write request is made [9].

Iostat is a disc monitoring programmer that continuously tracks several metrics, including storage input and output information [13] [19]. Iostat has 17 properties in total. We gathered an overall 17 characteristics in a single second, andwe added the average variation of each attribute's 50 total cumulative values to the data. From the 34 qualities gatheredthrough this approach, we eliminated the attributes that were superfluous. First, the CPU-related properties describe the system's overall I/O-related CPU utilization. Since each Solidstate drive utilized in the same system has the same value andthis is not a value for an attribute specific to each Solid statedrive, determining if the trash from each SSD is removed and not stored with the data is not possible. We also got rid of theread/write bandwidth feature. The read operation has noimpact on the garbage collection, while the write operation's bandwidth is likewise influenced by outside variables like does not and block size accurately reflect the GC condition. The data did, however, provide the standard deviation of thewrite operation's band-width fluctuation.

The data gathered using the aforementioned technique is used to train the machine learning algorithm. When X data is input by a learning system, map learning classifies the data by supplying information on both A state and B state. Machine learning is an appropriate solution for this issue since the Solid state drive GC detection problem corresponds to the aforementioned scenario [11] [15]. As a result, the data was labelled as having been obtained under particular circumstances (GC or non-GC). The use of a commercial Solid state drive for GC could not be correctly determined at the OS level, though. To address this issue, we employ OpenSSD [10]. The experimental board known as the OpenSSD, which may be used to create and implement SSD firmware, is seen in Fig. 4. The Flash Translation Layer of OpenSSD may be changed, and different SSD state details can be seen at the OS level. To output the debugging code while GC takes place, we updated the Flash Translation Layer and applied it to OpenSSD [20]. When the code for debugging is displayed, this enables you to confirm that Garbage Collection has been carried out at the OS level. We were able to classify the type of data (GC or non-GC) that was being gathered using iostat thanks to this information [14].

RESULT & EXPRIMENT

The results of the first experiment, which tested the SSD's bandwidth for 3000 seconds without the use of a garbage collection detector, are shown in Fig. 5. The SSD's average bandwidth in the first testing was 449 MB/s, and two bandwidth decreases were noted. SSD bandwidth is anticipated to increase once again as a result of the OS performing the TRIM command when not in use. This is because of Garbage Collection, lower bandwidth, and automated TRIM setting. The band-width speed has decreased lower than 400 MB/s 137 times, which indicates that 3000 seconds of the bandwidth have been impacted by garbage collection.

As a consequence, it was determined that even when automated TRIM is enabled, input and output are still conducted on the SSD, and the GC effect is produced if there is never any downtime. In the second experiment, a garbage collection detector-like environment was used to assess the SSD's bandwidth over the course of 3000 seconds. This led to the measurement of the bandwidth depicted in Fig. 6. The second experiment's average SSD bandwidth was 462 MB/s, which is 13 MB/s more than the first experiment's. Like the previous experiment, the second experiment demonstrated two occurrences of bandwidth decrease. However, the duration assessed as 66 seconds with a band-width of 400 megabyte per second or less in the second trial as opposed to 137 seconds in the first experiment. By getting less than 52% of the impacts of garbage collection compared to when the first GC detector wasn't running, it can be said that the GC detector lessens the impact of GC.

Fig. 5. The Garbage Collection detector should be used before

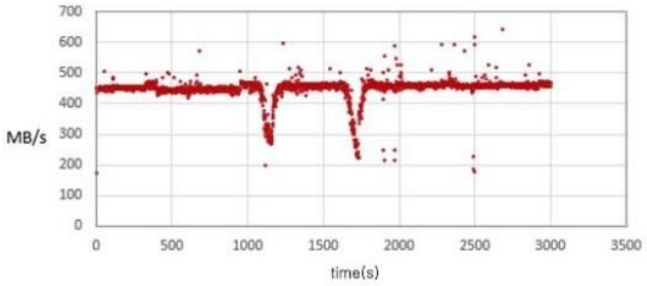
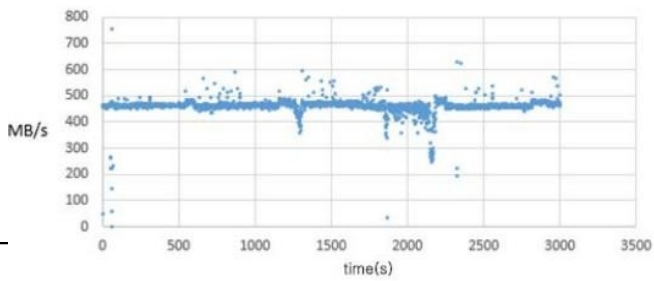


Fig. 6. Following use of the garbage detector



CONCLUSION

By the use of Solid state Drive status information and ML Technique methods that may be seen at the OS level, we have discovered SSD garbage collection in this research. In order to safeguard the SSD's storage space and lessen the impact of the Garbage Collection, the TRIM function command is also carried out on the identified Solid state drive. As a result, Garbage Collection's bandwidth loss is decreased and average SSD bandwidth is raised.

ABBRIVATION

MLC –Multi Layer Cell TLC – Triple Layer Cell QoS – Quality of Service SSD – Solid state drive
FTL - Flash Translation LayerML – Machine Learning HDD – Hard disk drive

REFERENCES

[1] SLC, MLC or TLC NAND for Solid State Drives by Speed Guide.net. https://www. speedguide.net/faq/slcmlc-or-tlc-Nand-for-solid-state- drives-406. Accessed 1 June 2020

[2] NAND Bad Columns analysis and removal by ruSolute.http://rusolut.com/nand -badcolumns-analysis-andremoval/. Accessed 1 July 2020

[3] Gubanov, Y., Afonin, O.: Recovering evidence from SSD drives: understanding TRIM, garbage collection and exclusions, Belkasoft, Menlo Park (2014)

[4] Geier, F.: The differences between SSD and HDD technology regarding forensic investigations, Sweden (2015)

[5] G. Zhu, J. Han and Y. Son, "A Preliminary Study: Towards Parallel Garbage Collection for NAND Flash-Based SSDs," in IEEE Access, vol. 8, pp. 223574-223587, 2020, doi: 10.1109/ACCESS.2020.3043123.

[6] S. Wang, Y. Zhou, J. Zhou, F. Wu and C. Xie, "An Efficient Data Migration Scheme to Optimize Garbage Collection in SSDs," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 40, no. 3, pp. 430-443, March 2021, doi: 10.1109/TCAD.2020.3001262.

[7] S. Wu et al., "GC-Steering: GC-Aware Request Steering and Parallel Reconstruction Optimizations for SSD-Based RAIDs," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 12, pp. 4587-4600, Dec. 2020, doi: 10.1109/TCAD.2020.2974346.

[8] C. Gao, L. Shi, C. J. Xue, C. Ji, J. Yang and Y. Zhang, "Parallel all the time: Plane Level Parallelism Exploration for High Performance SSDs," 2019 35th Symposium on Mass Storage Systems and Technologies (MSST), 2019, pp. 172-184, doi: 10.1109/MSST.2019.000-5.

[9] Q. Song, H. Jin, and X. Hu, Automated Machine Learning in Action. New York, NY: Manning Publications, 2022.

[10] Yong-Yeon Jo, Sang-Wook Kim, Moonjun Chung and Hyunok Oh, "Data mining in intelligent SSD: Simulation-based evaluation," 2016 International Conference on Big Data and Smart Computing (BigComp), 2016, pp. 123-128, doi:10.1109/BIGCOMP.2016.7425810.

[11] I. Gammoudi, R. Ghozi and M. A. Mahjoub, "Hybrid Learning Method for Image Segmentation," 2021 18th International Multi-Conference on Systems, Signals & Devices (SSD), 2021, pp. 667-672, doi: 10.1109/SSD52085.2021.9429446.

[12] M. Fukuchi, Y. Sakaki, C. Matsui and K. Takeuchi, "20% System- performance Gain of 3D Charge-trap TLC NAND Flash over 2D Floating-gate MLC NAND Flash for SCM/NAND Flash Hybrid SSD," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), 2018, pp. 1-5, doi: 10.1109/ISCAS.2018.8351309.

[13] Quinlan, J., C4.5: Programs for Machine learning. Morgan Kaufmann, San Francisco, 1992.

[14] U. Dixit, S. Bhatia and P. Bhatia, "Comparison of Different Machine Learning Algorithms Based on Intrusion Detection System," 2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-

CON), 2022, pp. 667-672, doi: 10.1109/COM-IT-CON54601.2022.9850515.

[15] H. Shi and M. Xu, "A Data Classification Method Using Genetic Algorithm and K-Means Algorithm with Optimizing Initial Cluster Center," 2018 IEEE International Conference on Computer and Communication Engineering Technology (CCET), 2018, pp. 224-228,doi: 10.1109/CCET.2018.8542173.

[16] Github.com. [Online]. Available: https://github.com/zhangchiyu10/pyC45/. [Accessed: 31-Aug-2022].

[17] “fio(1): flexible I/O tester - Linux man page,” Die.net. [Online]. Available: http://linux.die.net/man/1/fio/. [Accessed: 31-Aug-2022].

[18] “filebench(1) - Linux man page,” Die.net. [Online]. Available: http://linux.die.net/man/1/filebench/. [Accessed: 31-Aug-2022].

[19] “iostat(1) - Linux man page,” Die.net. [Online]. Available: http://linux.die.net/man/1/iostat/. [Accessed: 31-Aug-2022].

[20] [5] “The OpenSSD project,” The OpenSSD Project. [Online]. Available: http://www.openssd-project.org/. [Accessed: 31-Aug- 2022].