사물인터넷융복합논문지 Vol. 10, No. 3, pp. 13-18, 2024 https://doi.org/10.20465/KIOTS.2024.10.3.013

Wear Leveling Technique using Random Selection Method in Flash Storage

Jung Kyu Park[1], Eun Young Park[2*]
[1]Professor, Department of Computer Engineering, Changshin University
[2]Professor, Rinascita College of Liberal Arts and Sciences, Shinhan University

플래시 스토리지에서 랜덤 선택 방법을 활용한 마모도 평준화 기법

박정규[1], 박은영[2*]
[1]창신대학교 컴퓨터전공 교수, [2]신한대학교 리나시타교양대학 교수

Abstract Recently, reliability has become more important as flash-based storage devices are actively used in cloud servers and data centers. Flash memory chips have limitations in reading/writing, so if writing is concentrated in one location, the chip can no longer be used. To solve this problem and improve reliability, it is necessary to equalize the wear of flash memory chips. However, in order to equalize the wear of flash memory with increasing capacity, the workload increases proportionally. In particular, when searching for a block with the maximum/minimum number of deletions for all blocks of a flash memory chip, the cost increases depending on the capacity of the storage device. In this paper, a random selection method of blocks was applied to solve the previous problem. When k is the randomly selected block, actual experimental results confirmed that searching all blocks with an k value of 4 or more yields similar results.

Key Words : Cloud; Flash Memory; FTL; Storage; Wear-leveling

요 약 최근에는 클라우드 서버, 데이터센터 등에서 플래시 기반의 저장장치가 활발히 활용되면서 신뢰성이 더욱 중요해지고 있다. 플래시 메모리 칩은 읽기/쓰기에 제한이 있어 한곳에 쓰기가 집중되면 칩을 더 이상 사용할 수 없게 된다. 이와 같은 문제를 해결하고 신뢰성을 향상시키기 위해서는 플래시 메모리 칩의 마모를 균등화하는 것이 필요하다. 그러나 대용량이 되어가는 플래시 메모리의 마모 균등화를 위해서는 작업 부하가 비례적으로 증가한다. 특히, 플래시 메모리 칩의 전체 블록의 삭제 횟수가 최대/최소인 블록 블럭을 검색할 때 저장장치의 용량에 따라 비용이 증가한다. 본 논문에서는 앞의 문제를 해결하기 위해서 블럭의 무작위 선택 방법을 적용하였다. 무작위로 선택하는 블록을 k 라고 할 때 실제 실험 결과를 통해 k 값이 4 이상 전체 블록을 검색하는 것과 비슷한 결과를 보여주는 것을 확인하였다.

주제어 : 클라우드, 플래시메모리, FTL, 저장장치, 마모 평준화

This work was supported by Changshin University Research Fund of 2024-011
*교신저자 : 박은영(eypark@shinhan.ac.kr)
접수일 2024년 04월 29일 수정일 2024년 05월 10일 심사완료일 2024년 05월 27일

1. Introduction

Flash memory, which was mainly used in mobile devices such as smartphones and laptops, has recently been actively used in servers and data centers [1-6]. For this, the following two requirements must be satisfied: First, it has large capacity. Recently, flash memory is being made using a process of 20 nm or less, and the degree of integration is being increased by introducing 3D NAND technology and MLC (Multi-Level Cell) technology [7-8]. As a result, 256GB flash chips are being released, and connecting them through multi-channel/way makes storage devices larger than TB possible.

The second requirement is high reliability. Flash has an endurance problem due to a limited number of erase times, and also generates disturbance error due to interference and retention error due to charge discharge. Therefore, high-performance ECC (Error Correction Code) that can correct more error bits is required, and RAID technology is also introduced to increase reliability [9-12]. In particular, the durability of MLC is decreasing to less than 1,000 cycles compared to 100,000 cycles of existing SLC, and the importance of wear-leveling is increasing accordingly.

Wear leveling technology distributes stored data evenly throughout the flash memory rather than concentrating it in one place, ultimately improving the overall lifespan [13, 14]. Wear leveling can be divided into dynamic wear leveling, which attempts leveling when a write request occurs, and static wear averaging, which targets the entire flash area. Static wear leveling has a greater leveling effect and is recognized as an essential element in MLC-based flash. Therefore, this study focuses on static wear averaging. However, the proposed technique is also applicable to dynamic wear leveling techniques.

The basic idea of static wear leveling is to replace the block with the largest and smallest number of erases among all blocks that make up flash memory. To do this, it is necessary to compare the number of deletions for all blocks, but as flash storage devices have recently become larger in capacity, this cost is rapidly increasing. For example, if the flash memory chip size is 256GB and each block size is 2MB (8KB page * 256 page/block), there will be 128K blocks, and in order to retrieve the maximum/minimum values, this size is needed. Comparison becomes necessary.

Specifically, the proposed technique selects n blocks from all blocks and replaces the blocks with the maximum/minimum number of deletions. As a result of the experiment, the effect of wear equalization could be obtained even when n was 2, and when k was 4, performance close to the overall comparison technique could be obtained. Ultimately, the proposed technique in this paper has the advantage of reducing the time consumed by wear leveling while maintaining the advantages of wear leveling.

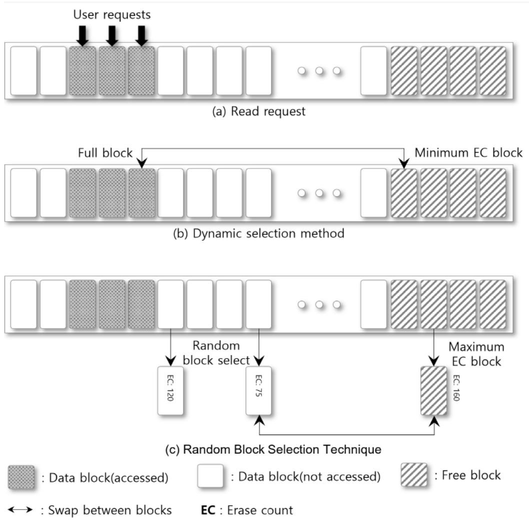
In this paper, we propose a new random-based static wear leveling technique to reduce this load. The basic idea of this technique is to randomly select a few data from a large data set and select the maximum/minimum among them over a long period of time, ultimately becoming similar to a complete comparison [15, 16]. In fact, this idea was used to select worker nodes in large-scale data centers such as Hadoop.

2. Random Block Selection Technique(RBST)

In this paper, we present a new wear leveling technique based on random selection. When selecting blocks for wear leveling, the RBST (Randon Block Selectoin Technique) compares the existing static wear leveling technique by checking the number of deletions in all blocks one by one. Unlike selecting the block with the smallest number of deletions, a block is randomly selected from all blocks and replaced.

The dynamic technique in Figure 1(b) shows a technique in which wear leveling is performed on the block on which the current write operation is about to be performed. As shown in Figure 1(a), flash memory separately manages a clean area (free block) where data is not written in addition to the area where data is written. The write operation must be performed in an area where data has not been written, but if the requested area is full or data has already been written, a new block for the write operation must be selected. At this time, by selecting a block with a small number of delete operations from among the clean blocks prepared in advance and performing a write operation, the wear of the blocks being written can be evenly distributed. However, in this technique, if data is continuously written only to specific blocks, other blocks are not used at all, and only the blocks that are used are continuously deleted. This causes a situation where only certain blocks expire earlier than other blocks and become bad blocks.

If the number of block replacements is small, RBST, which randomly selects and replaces blocks, will have poor distribution of the number of deletions across blocks compared to the existing static wear leveling technique. However, wear leveling requires blocks to be replaced continuously at certain points until the cell's lifespan expires, so a significant number of replacements occur. Therefore, even if blocks are selected randomly, it can be expected that all blocks will be selected evenly in the end. In addition, a new method is proposed by taking advantage of the fact that in existing studies, the method of selecting two at random, comparing them, and then selecting an appropriate value shows superior results in selecting them evenly than selecting just one [12]. A method was used to randomly select two and select one of them.



[Fig. 1] Structure of RBST

Fig. 1(c) shows the structure of RBST proposed in the paper. Like the static wear leveling technique, a block is selected from all data blocks. When selecting a block, RBST randomly selects two blocks and replaces the block with the lowest erase count (EC) with the data block. Doing it this way can improve the leveling effect of wear more than just picking one at random.

3. Experimental resutls

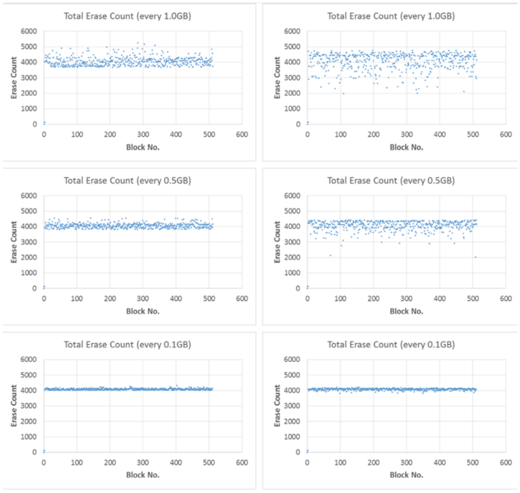
To verify the proposed technique, an experiment was conducted using a flash memory simulator called OpenNFM. In addition to FTL, OpenNFM includes several software layers to reproduce operations similar to actual hardware. The experiment was conducted by measuring the number of deletions of all blocks when a total of 2TB write requests occurred for only 100 specific blocks, and the simulator settings are shown in Table 1.

〈Table 1〉 Setup for OpenFNM

Configurations	
Page Size	8 KB
Pages per Block	128 Pages
Flash Total Size	512 MB
Read/Write	100 Blocks
Total Request Size	2 TB

Fig. 2 shows the number of deletions of entire blocks when experiments were performed with different execution frequencies of the wear leveling technique. In Fig. 2, the x-axis represents the total block number, and the y-axis represents the number of deletions of the block. The left is the result when using the existing static wear leveling technique, and the right is the result when using RBST. Table 2 shows the characteristics of SWL and RBST. Both SWL and RBST are performed in the background, but there is a significant difference in execution time. SWL searches all blocks to exchange blocks, but RBST randomly selects two blocks. Additionally, SWL considers the number of deletions when examining blocks, but RBST selects blocks without any conditions. For this reason, RBST is superior in terms of time complexity.

Based on this, Fig. 3 shows the standard deviation of the number of deletions of all blocks according to the execution frequency when using the existing static wear leveling technique and when using RBST. In Fig. 3, the x-axis represents the interval of write requests, and the y-axis represents the standard deviation.



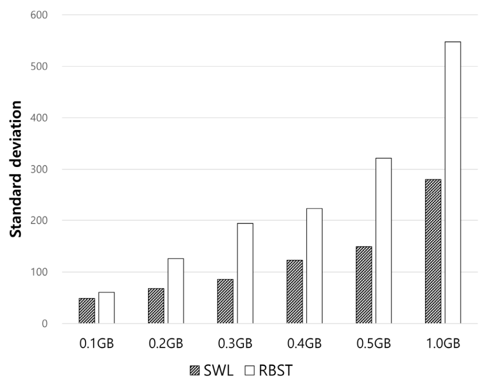
[Fig. 2] Block deletion count comparison

〈Table 2〉 Comparison of SWL and RBST

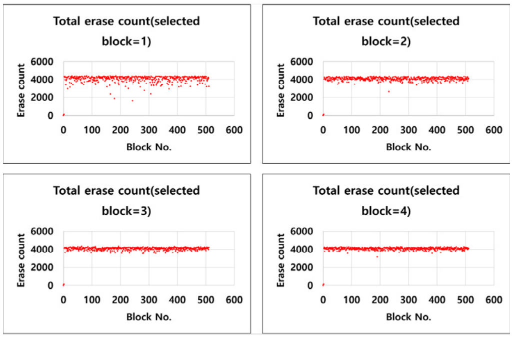
	SWL	RBST
Run	background	background
Target	all block	random 2 block
Condition	erase count	no condition
Time complexity	high	how

It can be seen that the number of deletions in the front block of each SWL and RBST figure in Fig. 2 is very small. This is a block in which code that is not subject to block replacement is stored. As shown in Fig. 2 and Fig. 3 if the wear leveling technique is performed every time a 1GB write request comes in, the standard deviation of the static wear leveling technique is smaller, so it can be said to equalize the wear level for the entire block than RBST, but RBST also has a certain degree of effect. is showing. And as the execution frequency decreases, the standard deviation also decreases, so RBST has almost the same effect as the existing static wear leveling technique.

Additionally, in order to compare performance according to the number of randomly selected and compared blocks, an experiment was conducted by performing RBST every time a write request of 300MB was received. Fig. 4 shows the number of deletions for each, and Fig. 5 is a graph comparing the standard deviation. In Fig. 4, the x-axis represents the total block number, and the y-axis represents the number of deletions of the block. Fig. 4 shows that the greater the number of randomly selected blocks, the more evenly distributed the wear is. However, when there are two or more blocks selected, it can be seen that there is no significant difference that affects the lifespan. The more blocks are selected, the more comparison operations are performed, and experimental results show that the method of selecting two is the most efficient.

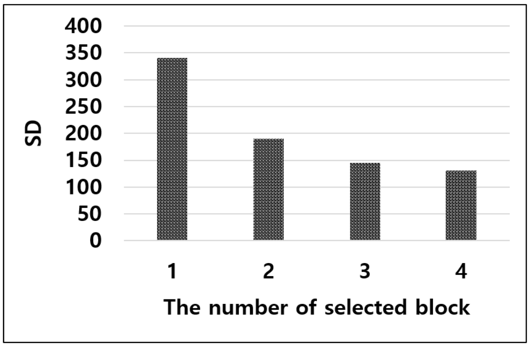


[Fig. 3] Standard deviation of number of deletions



[Fig. 4] Number of deletions based on number of block selections

Through the above experiment, it was confirmed that as the execution frequency increases, RBST shows almost the same effect as the existing static wear leveling technique. In Fig. 5, the number of selected blocks is displayed, and the y-axis displays the standard deviation.



[Fig. 5] Standard deviation according to the number of block selections

The average time taken to select a block to replace through one wear leveling technique was measured to be 0.78 ms for the SWL technique and 0.20 ms for the RBST technique. It can be seen that RBST reduces the execution time by more than 1/3 compared to the existing wear leveling technique. Since the previous experimental results are the results when the total number of blocks on the flash memory development board is 4096, it is expected that the performance speed will be higher in actual flash memory storage products with much more blocks. In the case of existing wear leveling techniques, the time to perform wear leveling increases significantly as the capacity increases, but RBST has the advantage of having a constant execution time regardless of the number of blocks.

4. Colclusion

In this paper, we propose an efficient wear leveling technique that randomly selects and replaces blocks when applying the wear leveling technique to all blocks. In addition, through experiments on a simulator and an actual development board, it was proven that the proposed technique not only provides similar wear to the existing technique but also performs at a faster speed. This is equivalent to repeating random selection over a long period of time and eventually all blocks will be examined. The result is wear leveling where all blocks are inspected.

REFERENCES

[1] H.Uchigaito, S.Miura and T.Nito. "Efficient Data-Allocation Scheme for Eliminating Garbage Collection During Analysis of Big Graphs Stored in NAND Flash Memory," IEEE Transactions on Computers, Vol.67, No.5, pp.646-657. 2018.

[2] J.K.Park, Y.Seo and J.Kim, "A Flash-Based SSD Cache

Management Scheme for High Performance Home Cloud Storage," IEEE Transactions on Consumer Electronics, Vol.65, No.3, pp.418-425. 2019.

[3] F.Wang, Y.Feng, X.Zhan, B.Chen and J.Chen, "Implementation of Data Search in Multi-Level NAND Flash Memory by Complementary Storage Scheme. IEEE Electron Device Letters," Vol.41, No.8, pp.1189-1192, 2020.

[4] J.K.Park and E.Y.Park, "Algorithm Performance Comparison for Encryption of Big Data Stored in Cloud," Journal of Next-generation Convergence Technology Association, Vol.7, No.10, pp.1577-1584, 2023.

[5] J.K.Park, "Machine Learning Algorithm to Predict Wildfire Occurrence Based on IoT Sensor Data. Journal of Next-generation Convergence Technology Association, Vol.6, No.6, pp.967-973, 2022.

[6] Y.Cho, "An Exploratory Study on Consumers' Perception towards Upcycling through Big Data Analysis," Journal of Next-generation Convergence Technology Association, Vol.6, No.2, pp.172-185, 2022.

[7] C.Huang, F.Liu, Q.Wang and Z.Huo. Low Power Program Scheme With Capacitance-Less Charge Recycling for 3D NAND Flash Memory. IEEE Transactions on Circuits and Systems II: Express Briefs, Vol.68, No.7, pp.2478-2482, 2021.

[8] E.K.Jang, I.J.Kim, C.A.Lee, C.Yoon and J.S.Lee, "Analysis of Residual Stresses Induced in the Confined 3D NAND Flash Memory Structure for Process Optimization," IEEE Journal of the Electron Devices Society, Vol.10, pp.104-108, 2023.

[9] J.Kim and J.K.Park, "Building Reliable Massive Capacity SSDs through a Flash Aware RAID-Like Protection," Applied Science, Vol.10, pp.9149, 2020.

[10] H.S. Lee, "A Design of SSD Dedicated RAID System for Efficient Resource Management", Journal of Korea Internet of Things Society, Vol.10, No.2, pp.109-114, 2024.

[11] S.Wu, C.Du, W.Zhu, J.Zhou, H.Jiang, B.Mao and L.Zeng. "EaD: ECC-Assisted Deduplication With High Performance and Low Memory Overhead for Ultra-Low Latency Flash Storage," IEEE Transactions on Computers, Vol.72, No.1, pp.208-221, 2023.

[12] J.Cui, C.Liu, J.Liu, J.Huang and L.T.Yang, "Exploiting Uncorrectable Data Reuse for Performance Improvement of Flash Memory," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.41, No.6, pp.1785-1798, 2022.

[13] Y. Song, Y. Lv and L. Shi, "Adaptive Differential Wearing for Read Performance Optimization on High-Density nand Flash Memory," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.43, No 1, pp.380-393, 2024.

[14] M.C.Yang, Y.H.Chang, T.W.Kuo and F.H.Chen. "Reducing Data Migration Overheads of Flash Wear Leveling in a Progressive Way," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.24, No.5, pp.1808-1820, 2016.

[15] M.Raquibuzzaman, A.Milenkovic and B. Ray, "Intrablock Wear Leveling to Counter Layer-to-Layer Endurance Variation of 3-D NAND Flash Memory," IEEE Transactions on Electron Devices, Vol.70, No.1, pp.70-75, 2023.

[16] M. Mitzenmacher, "The power of two choices in randomized load balancing. IEEE Transactions on Parallel and Distributed Systems," Vol.12, No.10, pp.1094-1104, 2001.

박 정 규(Jung Kyu Park) [종신회원]

- 2002년 2월 : 홍익대학교 컴퓨터공학과 (공학석사)
- 2013년 8월 : 홍익대학교 컴퓨터공학과 (공학박사)
- 2016년 3월 ~ 2007년 2월 : 서울여자대학교 초빙교수
- 2018년 3월 ~ 현재 : 창신대학교 컴퓨터소프트웨어공학과 교수

〈관심분야〉
사물인터넷, 로보틱스, 임베디드 시스템

박 은 영(Eun Young Park) [정회원]

- 2006년 2월 : 경희대학교 교육학과 (교육학석사)
- 2014년 2월: 서울시립대학교 에너지환경시스템공학과 (공학박사)
- 2020년 9월 ~현재 신한대학교 리나시타교양대학 조교수
- E-Mail : eypark@shinhan.ac.kr

〈관심분야〉
Nonlinear computer modeling, 폐기물 자원화, 환경화학