Journal of The Korea Society of Computer and Information
Vol. 22 No. 3, pp. 1-8, March 2017
www.ksci.re.kr
https://doi.org/10.9708/jksci.2017.22.03.001

Time-Aware Wear Leveling by Combining Garbage Collector and Static Wear Leveler for NAND Flash Memory System

Sang-Ho Hwang*, Jong Wook Kwak**

Abstract

In this paper, we propose a new hybrid wear leveling technique for NAND Flash memory, called Time-Aware Wear Leveling (TAWL). Our proposal prolongs the lifetime of NAND Flash memory by using dynamic wear leveling technique which considers the wear level of hot blocks as well as static wear leveling technique which considers the wear level of the whole blocks. TAWL also reduces the overhead of garbage collection by separating hot data and cold data using update frequency rate. We showed that TAWL enhanced the lifetime of NAND flash memory up to 220% compared with previous wear leveling techniques and our technique also reduced the number of copy operations of garbage collections by separating hot and cold data up to 45%.

▸ Keyword : NAND Flash Memory, Wear Leveling, Garbage Collection, Elapsed Time, Update Frequency Rate

I. Introduction

비휘발성, 저전력, 작은 크기 등의 장점을 가지고 있는 낸드 플래시 메모리는 저장매체로 사용하기 위해서 다음과 같은 특징들을 고려해야한다. 낸드 플래시는 제자리 갱신이 불가능하여, 저장된 데이터를 업데이트하기 위해서는 지움 연산을 먼저 수행해야하는 쓰기 전 지움 (erase-before-write) 특징을 가지고 있다[1]. 최소 연산 단위인 페이지 단위로 이루어지는 읽기 및 쓰기 연산과 다르게 지움 연산은 큰 단위인 블록 단위로 이루어진다. 업데이트에서의 오버헤드를 줄이기 위해서 낸드 플래시 시스템은 업데이트가 이루어지는 기존의 데이터는 무효화하고 업데이트된 데이터를 다른 빈 공간에 쓰는 정책을 사용하고, 달라진 물리 주소를 연결하기 위해 사상 테이블 (Mapping Table)을 사용한다. 이 과정에서 생성되는 무효화된 공간들은 지움 연산을 통하여 다시 빈 공간으로 만들어지며, 이 과정을 가비지 컬렉션 (Garbage Collection)이라 한다. 또한 낸드 플래시는 수명이 있어 쓰기 및 지움 연산수가 한정되어 있다. 사용되는 애플리케이션에 따라 차이가 있지만 블록에 저장되는 데이터들은 일반적으로 시간 및 공간 지역성을 가지고 있기 때문에, 낸드 플래시 시스템들은 모든 블록들을 골고루 사용할 수 있도록 마모도 평준화 (Wear Leveling) 기법을 사용한다. 마모도 평준화 기법은 크게 CB (Cost Benefit), CAT(Cost Age Time)와 같이 가비지 컬렉션 수행 시 핫 블록의 마모도를 같이 고려하는 동적 마모도 평준화 (Dynamic Wear Leveling) 기법과 BET (Block Erasing Table), AWL (Adaptive Wear Leveling)과 같이 전체 블록의 마모도를 고려하는 정적 마모도 평준화 (Static Wear Leveling) 기법으로 구별할 수 있다[2].

일반적으로 동적 마모도 평준화 알고리즘들은 무효 페이지가 많은 핫 블록들을 마모도 평준화 대상으로 선택하여 오버헤드가 적은 편이지만 갱신이 이루어지지 않는 완전 콜드 데이터를 가지고 있는 블록들이 많은 경우 마모도 평준화를 수행하지 못해 시스템 수명이 짧아지는 단점이 있다. 반면, 정적 마모도 평준화 기법은 마모가 거의 일어나지 않는 콜드 블록을 대상으

• First Author: Sang-Ho Hwang, Corresponding Author: Jong Wook Kwak
*Sang-Ho Hwang(snailcom@ynu.ac.kr), Dept. of Computer Engineering, Yeungnam University
**Jong Wook Kwak(kwak@yu.ac.kr) Dept. of Computer Engineering, Yeungnam University
• Received: 2016. 09. 24, Revised: 2016. 10. 17, Accepted: 2016. 12. 01.
• This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(No. NRF-2014R1A1A2057146).

로 마모도 평준화를 수행하기 때문에 이주되는 데이터의 양이 많은 단점이 있지만, 전체 블록을 대상으로 마모도 평준화를 수행하기 때문에 수명 연장에 장점을 가지고 있다. 각각의 기법들에 대한 단점들을 상쇄하기 위해서, 동적 마모도 평준화 기법과 정적 마모도 평준화 기법들은 동시에 사용할 수 있다. 하지만 지금까지 제안된 많은 정적 마모도 평준화 기법들은 가비지 컬렉션 알고리즘으로 마모도 평준화를 수행하지 않는 GA (Greedy Algorithm) 기법에 최적화되어 있거나 별도의 메모리 공간을 사용하여 그에 따른 오버헤드를 가지고 있다. 이로 인하여 이전의 정적 마모도 평준화 기법들을 동적 마모도 평준화 기법을 동시에 사용하는 시스템에 적용하는 경우 수명 연장이 제한적이거나 메모리 효율이 떨어지는 단점이 있다.

본 논문에서는 낸드 플래시 시스템의 수명을 향상시키는 TAWL (Time-Aware Wear Leveling)을 제안한다. 제안하는 기법은 가비지 컬렉션 수행 시 블록의 마모도를 고려하는 동적 마모도 평준화 기법과 낸드 플래시 시스템 내의 모든 블록들의 마모도를 고려하는 정적 마모도 평준화를 동시에 수행하는 하이브리드 마모도 평준화 기법이다. 제안하는 기법은 동적 마모도 평준화 기법과 정적 마모도 평준화 기법을 동시에 적용하여 동적 마모도 평준화 기법과 비슷한 수준의 적은 오버헤드로 정적 마모도 평준화 기법 이상으로 수명을 연장하고 있다. 또한 제안하는 기법은 블록을 할당한 시간을 기반으로 콜드 데이터를 판별하는 방법을 사용하여 이주된 데이터가 가까운 미래에 또 다시 이주되는 것을 방지하여 쓰기 연산 오버헤드를 줄이고 있다.

이하 논문의 구성은 다음과 같다. 2장에서는 낸드 플래시에서 발생할 수 있는 문제와 기존 연구에 대하여 기술한다. 3장에서는 본 논문에서 제안하는 마모도 평준화 기법을 소개하고, 4장에서는 실험을 통해 기존의 기법들과 비교하고, 5장에서는 결론 및 향후 연구과제에 대하여 기술한다.

II. Related Works

지금까지 낸드 플래시 시스템의 수명을 증가시키기 위해서 마모도 평준화 기법들이 많이 연구되어왔다. 마모도 평준화 기법은 크게 동적 마모도 평준화와 정적 마모도 평준화로 구별할 수 있다. 동적 마모도 평준화 기법은 삭제 및 업데이트가 자주 일어나는 핫 블록들을 대상으로 마모도 평준화를 수행한다. 이를 통해 동적 마모도 평준화 기법들은 블록 내에 무효 페이지가 많은 블록들이 희생블록으로 선택되어 마모도 평준화에 대한 오버헤드가 적은 장점이 있다. 대표적인 동적 마모도 평준화 알고리즘으로는 GA가 있다[3]. GA는 블록 내에 유효페이지가 가장 적은 블록을 가비지 컬렉션의 대상으로 하는 방법으로, 오버헤드가 낮지만 업데이트 빈도가 낮은 콜드 데이터가 많은 어플리케이션에 적용하게 되면 특정 블록을 빠르게 소모하여 마모도 불균형이 발생하는 단점이 있다.

GA에서 콜드 데이터가 많은 블록의 경우 희생블록으로 선정되지 않는 단점을 해결하기 위해 CB 기법이 제안되었다. CB는 식 (1)을 사용하여 가장 큰 값을 가지는 블록을 희생블록으로 선정하는 방법을 통하여 마모도 평준화를 수행한다[4].

$$age_i \times \frac{1-u}{2u} \quad (1)$$

식 (1)에서 u는 유효 페이지의 비율이며, age_i는 현재 시간에서 블록의 페이지 중에 마지막으로 무효화된 시간을 뺀 것으로, 무효화 된 이후 경과 시간을 의미한다. $1-u$는 무효 페이지에서 오는 이득이며, $2u$는 유효페이지를 읽고 옮기는데 드는 비용을 의미한다. CB는 희생 블록 선택 시 발생되는 비용뿐만 아니라 경과 시간도 고려하는 식 (1)을 통하여 콜드 블록도 가비지 컬렉션에 참여하도록 유도하여 마모도 평준화를 이루고 있지만, 콜드 데이터와 핫 데이터를 구별하여 저장하지 않아 오버헤드가 증가하는 단점이 있다.

CAT는 CB가 콜드 데이터만 인식하는 단점을 개선하기 위해 제안되었으며, 식 (2)를 사용하여 가장 작은 값을 가지는 블록을 희생블록으로 선정한다[5].

$$\frac{u}{1-u} \times \frac{1}{age_a} \times EC \quad (2)$$

식 (2)에서 age_a는 사용을 위해 블록이 할당된 이후 지금까지 경과된 시간을 의미하며, EC (Erase Count)는 블록의 삭제 횟수를 의미한다. CAT는 가비지 컬렉션에서의 오버헤드를 줄이기 위해 핫/콜드 데이터를 구별하여 저장하는 방법을 사용하고 있지만 여전히 콜드 데이터를 자주 이주시키는 단점이 있다. 그 외에도, CATA, CBA, FeGC (Fast and Endurant Garbage Collection), CAPi (Cost Age with Proportion of invalid page)와 같은 기법들이 제안되었으며, 최근에는 스왑 시스템 기반의 SCATA, SAGC, LFGC, NSAGC 등의 가비지 컬렉션 기법들이 제안되었다[6-13]. 이러한 동적 마모도 평준화 기법들은 오버헤드가 적은 장점이 있지만 OS와 같은 영역은 마모도 평준화에 참여하지 않아 마모도 불균형이 발생한다.

비록 동적 마모도 평준화 기법들이 가비지 컬렉션 수행 시에 발생되는 유효페이지 이주에 따른 오버헤드도 고려하여 마모도 평준화 비용을 줄이고 있지만, 블록 내의 데이터에 대해 업데이트가 이루지지 않는 완전 콜드 블록이 많은 경우 수명이 줄어드는 단점이 있다. 이는 완전 콜드 블록이 대부분의 시간 기반 동적 마모도 평준화 기법에서는 희생블록으로 선정되지 않기 때문이다.

정적 마모도 평준화 기법은 낸드 플래시내의 모든 블록들에 대하여 마모도 평준화를 수행한다. 동적 마모도 기법과 다르게 완전 콜드 블록도 마모도 평준화 대상으로 선택되기 때문에 정적 마모도 평준화 기법은 동적 마모도 기법에 비해 수명이 긴 장점이 있다. 대표적인 정적 마모도 평준화 기법에는 BET가

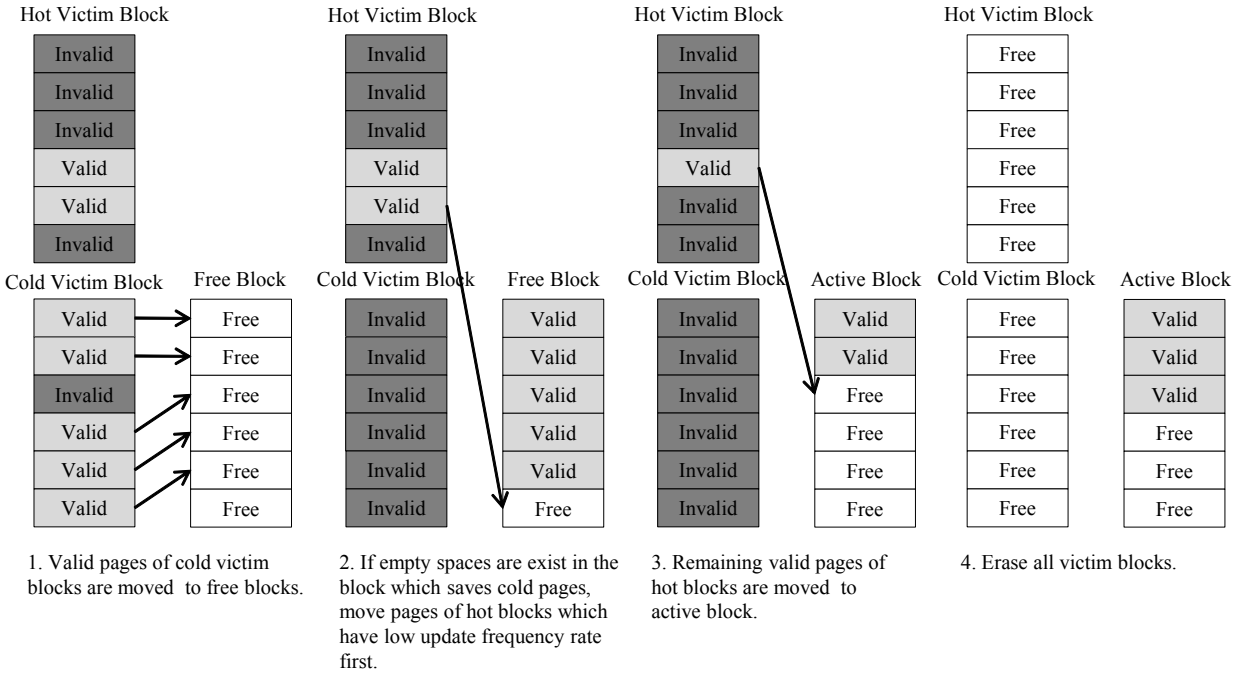

Fig. 1. The migration algorithm of valid pages of victim blocks in TAWL

있다[14]. BET는 라운드 단위로 동작하며, 해당 라운드에 가비지 컬렉션이 수행되지 않는 블록들은 강제로 가비지 컬렉션에 포함되도록 하여 모든 블록들이 골고루 사용되도록 유도하고 있다. BET는 전체 블록들을 사용하여 수명을 증가시키고 있지만, 마모도 평준화 수행 시 선택되는 블록들을 평가하지 않아 효율이 떨어지는 단점이 있다.

AWL은 핫 블록의 마모도가 특정 임계값에 도달하기 전까지 마모도 평준화를 수행하지 않는다[15]. 임계값에 도달하게 되면, AWL은 마모도가 가장 적은 블록을 콜드 블록으로 선정하여 핫 블록의 데이터를 모두 선택된 콜드 블록으로 이주시켜 블록간의 마모도 평준화를 수행한다. 비록 AWL이 임계값을 통해 마모도 평준화 수행 시점을 늦추어 콜드 블록 이주에 대한 오버헤드를 줄이고 있지만 마모 정도에 따라 콜드/핫 데이터를 판별하고 있어, 콜드/핫 블록이 교체될수록 효율이 떨어지는 단점이 있다. 그 외에도, SAW (System-Assisted Wear leveling), Dual Pool, Rejuvenator, HaWL, CrEWL (Cold data identification using raw bit Error rate in Wear Leveling), EPET-WL (Enhanced Prediction and Elapsed Time-based Wear Leveling)과 같은 마모도 평준화 기법들이 제안되었다[16-21].

이상에서 살펴본 바와 같이, 낸드 플래시 메모리 시스템의 수명을 증가시키기 위해 동적 마모도 평준화 기법과 정적 마모도 기법을 동시에 수행할 수 있다. 하지만 지금까지 제안된 알고리즘들은 두 기법의 융합에 대하여 고려되지 않았다. 본 논문에서는 동적 마모도 평준화 기법과 정적 마모도 평준화를 모두 수행하는 TAWL (Time Aware Wear Leveling)을 제안한다.

III. TAWL (Time-Aware Wear Leveling)

제안하는 TAWL은 하이브리드 마모도 평준화 기법으로, 빠르게 수명이 줄어드는 핫 블록들에 대하여 동적 마모도 평준화를 수행하는 가비지 컬렉션 모듈과 오랜 시간동안 동적 마모도 평준화 과정에 참여하지 않는 콜드 블록들에 대하여 마모도 평준화를 수행하는 정적 마모도 평준화 모듈로 이루어져있다.

1. Garbage Collection

첫 번째 모듈은 가비지 컬렉션 모듈이며, 업데이트가 빈번하게 일어나는 핫 블록들을 대상으로 마모도 평준화를 수행한다. 알고리즘 1은 TAWL의 가비지 컬렉션 모듈에서 희생 블록 선택 및 콜드 블록의 유효 페이지 이주에 대한 의사코드 (pseudo code)를 보여주고 있다. 가비지 컬렉션은 아래 식 (3)을 이용하여 가장 큰 값을 가지는 블록을 희생블록으로 선택한다.

$$value = age_a \times \frac{1-u}{(1+u)^{2-\frac{EraseCount_{avg}}{EraseCount_{MAX}}}} \qquad (3)$$

$$age_a = Time_c - Time_a$$

식 (3)에서 u는 블록 내에 유효 페이지의 비율을 의미하며, $Time_c$는 현재 시간, $Time_a$는 블록이 쓰기 연산이나 업데이트가 발생했을 때 데이터를 저장할 목적으로 활성 블록 (active block)으로 할당된 시간을 의미한다. 따라서 age_a는 블록이 할

Algorithm 1: TAWL_GarbageCollector_1

1: **for** n = 1 to N **do**
2: $blk_{value} \leftarrow (Time_c - Time_a) \times \frac{1-u}{(1+u)^2 - \frac{EraseCout\_AVG}{EraseCount\_MAX}}$
3: PushMaxHeap(blks, blk)
4: **end**
5: **while** $FreeBlock_{count} > FreeBlock_{MIN}$
6: blk ← PopMaxHeap(blks)
7: **if** blk is cold **then**
8: move all valid pages to youngest free blocks
9: **else**
10: push all valid pages to min heap structure
11: **end if**
12: **end**

Algorithm 2: TAWL_GarbageCollector_2

```
1: while pages_count > 0
2:    page ← PopMinHeap(pages)
3:    if empty spaces exist in the block
         which store cold pages then
4:       move the page to the cold block
5:    else
6:       move the page to active block
7:    end if
8: end
9: erase all blks
10: for b = 1 to all blks
11:    if ECT[block] > TH then
12:       block ← TAWL_WearLeveler(block)
13:       if exist block then
14:          Erase(block)
15:       end if
16:    end if
17: end
```

당된 이후 경과된 시간을 의미하며, age_a에 의해 유효 페이지가 많은 콜드 블록도 가비지 컬렉션에 참여하도록 유도하고 있다. $Erase\,Count_{MAX}$는 낸드 플래시 메모리의 블록이 가지는 최대 삭제 횟수를 의미하며, $Erase\,Count_{avg}$는 모든 블록들의 평균 삭제 횟수를 의미한다. 식 (3)에 의해 제안하는 가비지 컬렉션 알고리즘은 시스템 내 모든 블록들의 평균 삭제 횟수가 적을 때는 블록 내의 유효 페이지가 많은 블록들이 희생블록으로 선정되는 것을 최대한 늦추고 있다. 이를 통하여 전체 블록이 마모도에 여유가 있을 때에는 콜드 데이터를 자주 이주시키지 않아 전체 오버헤드를 줄이게 된다.

희생블록으로 선택되면 삭제 연산이 이루어지므로, 희생블록에 남아 있는 유효 페이지들은 다른 블록으로 이주되어야 한다. 이때 업데이트 빈도에 따라 데이터를 분리하여 옮기지 않으면 결국 모든 콜드 및 핫 데이터가 서로 섞이게 된다. 이로 인해 각 블록 모두에 콜드 데이터가 포함되어 결국 가비지 컬렉션 시 블록내의 유효 페이지 이주에 대한 비용이 전체적으로 높아져서 알고리즘의 효율이 떨어진다. 제안하는 TAWL은 업데이트 빈도에 따라 데이터를 분리하기 위해 가비지 컬렉션에 의해 선택된 희생블록들 중에 평균 할당 시간보다 이전에 할당된 콜드 블록의 유효 페이지들을 새로운 빈 블록들로 이주시킨다. 평균 할당 시간보다 이후에 할당된 핫 블록의 유효 페이지들은 업데이트 빈도에 따라 정렬을 수행하고 업데이트 빈도가 낮은 페이지들부터 순서대로 콜드 데이터가 저장된 블록들로 이주시키며, 한 개 블록의 페이지 수보다 적게 남은 나머지 페이지들은 활성블록으로 이주된다. 알고리즘 2는 업데이트 빈도가 높은 유효 페이지들의 이주에 대한 의사코드를 보여주고 있다. 블록 내에 모든 유효 페이지들이 이주된 희생블록들은 삭제 연산을 수행하여 빈 블록으로 만든 후, 빈 블록 풀(Free Block Pool)에 삽입된다. 그림 1은 유효 페이지 이주 순서에 대한 예를 보여주고 있다.

빈 블록들은 삭제 횟수를 기반으로 리스트를 구성하고 있으며, 요청하는 대상에 따라 다른 빈 블록을 할당한다. 기본적으로 빈 블록 할당은 마모도가 가장 적은 블록이 할당된다. 하지만 가비지 컬렉션 모듈에서 빈 블록의 할당을 요청하는 경우에는 정책에 따라 각기 다른 빈 블록을 할당한다. 복사될 유효 페이지들이 핫인 경우에는 마모도가 가장 적은 블록이 할당되며, 콜드인 경우에는 마모도가 가장 많은 블록이 할당된다. 희생블록의 할당 시점이 평균 할당 시간 이전이면 콜드 블록으로, 평균 할당 시간 이후이면 핫 블록으로 구별한다.

가비지 컬렉션에 의해 삭제가 이루어진 블록의 마모도가 특정 임계값을 초과한 경우에는 정적 마모도 평준화를 실행한다. 정적 마모도 평준화를 호출할 임계값은 식 (4)에 의해 결정된다.

$$TH = Erase\,Count_{avg} + (\alpha \times Erase\,Count_{MAX}) \qquad (4)$$

식 (4)에서 $Erase\,Count_{MAX}$는 낸드 플래시 메모리의 블록이 가지는 최대 삭제 횟수를 의미하며, $Erase\,Count_{avg}$는 모든 블록들의 평균 삭제 횟수를 의미한다. α는 평균 삭제횟수에서 추가로 허용할 삭제횟수를 결정하는 비율을 의미하는 것으로, 구체적인 값과 근거에 대해서는 4장에서 추가적으로 논의한다.

2. Static Wear Leveling

가비지 컬렉션 모듈은 핫 블록에 대하여 마모도 평준화를 수행한다. 하지만, 콜드 블록들은 마모도 평준화에 참여하지 않아 삭제 횟수에 불균형이 일어난다. 이러한 단점은 정적 마모도 평준화 기법을 동시에 적용하는 하이브리드 형태로 해결할 수 있다. 알고리즘 3은 정적 마모도 평준화 모듈의 의사코드를 보여주고 있다. 제안하는 기법의 정적 마모도 평준화 모듈은 마모도가 많은 핫 블록에 콜드 데이터를 저장하여 핫 블록의 마모도

Algorithm 3: TAWL_WearLeveler

```
Input : block, ECT, ATT
Output : block
1: cold_blk ← GetColdBlock(ATT)
2: if exist cold_blk then
3:   migrate data from cold_blk to block
4:   Erase(cold_blk)
5:   return cold_blk
6: end
7: return block
```

가 빠르게 소모되는 것을 방지하고, 콜드 블록이 마모도 평준화에 참여하도록 유도한다. 대부분 제안된 정적 마모도 평준화 기법에서는 마모도가 가장 적은 블록을 대상 블록으로 선택하지만, 이런 방법은 블록내의 데이터에 대하여 평가를 하지 않아 효율이 떨어지는 단점이 있다. 이에 반해, TAWL은 이주시킬 콜드 데이터를 판별하기 위해서 할당된 시간을 기반으로 이주 대상 블록을 선택하고 있으며, 최대한 많은 데이터를 옮겨 이들이 가비지 컬렉션에 늦게 참여할 수 있도록 블록 내에 유효 페이지의 수도 평가대상으로 한다. 따라서 TAWL은 옮길 데이터를 평가하기 위해 식 (5)를 이용한다.

$$cost = AT \times \frac{valid\,page}{total\,page} \qquad (5)$$
$$AT = Time_c - Time_i$$

콜드 데이터를 자주 이주시키는 오버헤드를 줄일 목적으로 TAWL은 가비지 컬렉션에 의해 선택되어 삭제된 블록의 마모도가 특정 임계값에 도달하기 전까지 정적 마모도 평준화를 수행하지 않는다. 블록의 마모도가 임계값을 초과하여 정적 마모도 평준화가 수행될 때 데이터를 이주해 저장될 블록은 이미 마모도가 많이 일어난 블록들이므로 식 (5)를 이용하여 가장 큰 값을 가지는 블록 즉, 가장 오랫동안 업데이트가 이루어지지 않은 블록의 페이지들이 저장되도록 한다. 페이지들이 모두 이주된 블록은 삭제 연산을 통해 빈 블록으로 만들어지고 빈 블록 풀에 삽입되어 차후 활성 블록으로 할당된다. 이를 통하여 기존의 삭제 연산이 많이 이루어진 블록들은 콜드 데이터 이주를 통해 가비지 컬렉션에 참여하는 시간을 늦추며, 콜드 데이터를 가지고 있었던 블록들은 활성블록으로 할당되어 핫 데이터를 저장하게 된다.

Table 1. Experimental environment

Chip	1 chip
Page Size	8192 + 640(Spare)byte
Block Size	2M + 160Kbyte, 256pages
Chip Size	1024 + 84(Extended)blocks
Random Read Time	60μs
Page Program Time	1500μs
Block Erase Time	5.0ms
P/E cycles	1000
Free Block trigger	5%

IV. Performance Evaluation

1. Experiment Setup

본 논문에서는 TAWL의 성능을 평가하기 위해 마이크로소프트사에서 개발한 SSD Extension for DiskSim 시뮬레이터를 사용하였다[22]. 낸드 플래시 블록의 초기 수명은 일반적인 TLC (Triple Level Cell)의 수명과 같이 1000으로 설정하였다. 낸드 플래시 시스템의 안정성을 위해 시스템에 필요한 최소 빈 블록 수는 5%로 설정하였다. 자세한 실험환경 정보는 표 1과 같다. 또한 가비지 컬렉션 수행 시에 유효 페이지 이주에 따른 오버헤드가 많은 가혹한 환경을 만들기 위해 실험에서는 전체 용량의 85%에 데이터를 저장한 상태로 진행하였으며, 저장된 데이터들의 업데이트 빈도에 따라 갱신시키는 형태로 실험을 진행하였다.

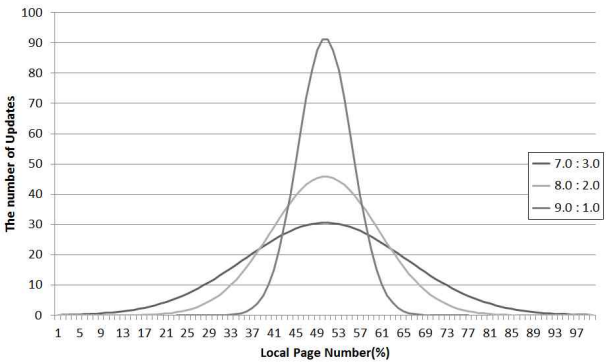

Fig. 2. Data set

실험에서는 다양한 업데이트 빈도에서의 각 알고리즘에 대한 성능을 테스트하기 위해 그림 2와 같은 콜드/핫 비율을 가지는 트레이스 파일을 사용하였다. 사용에 따른 낸드 플래시 내 블록들 간의 마모도 불균형 비율을 높이기 위해 실험에서는 데이터의 콜드/핫 비율을 7:3, 8:2, 9:1로 하였다. 성능 평가를 위해 실험에서는 기존의 CB, CAT, SAGC, FeGC, CAPi 알고리즘들을 제안한 TAWL과 수명 연장 및 오버헤드 측면에서 비교하였다.

TAWL의 정적 마모도 평준화 수행을 위한 임계 값 α는 블록들의 삭제횟수에 대한 분산을 결정하는 값으로 작은 값은 블록들의 삭제 횟수를 고르게 분포시키지만 유효 페이지가 많은 콜드 블록을 자주 이주시키는 오버헤드가 발생한다. 반대로 높은 α 값은 오버헤드를 많이 초래하는 정적 마모도 평준화 기법이 적게 수행되도록 만들지만, 콜드 블록들과 핫 블록들 간에 삭제 횟수의 편차가 심해져 결국 수명이 감소하게 된다. 그림 3은 다양한 α값에 따른 수명 변화와 정적 마모도 평준화 수행 횟수를

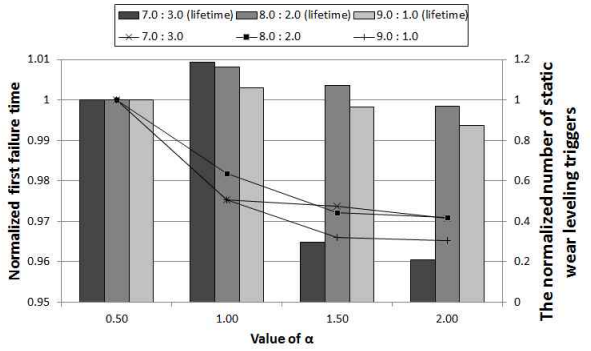

Fig. 3. Normalized first failure time and the normalized number of static wear leveling triggers according to value of α

보여주고 있다. 그림 3에서 보는 것과 같이, α 값은 1%로 설정했을 때 가장 성능이 좋으므로 본 논문의 실험에서 TAWL의 α 값은 1%로 설정한다.

2. Result and Discussion

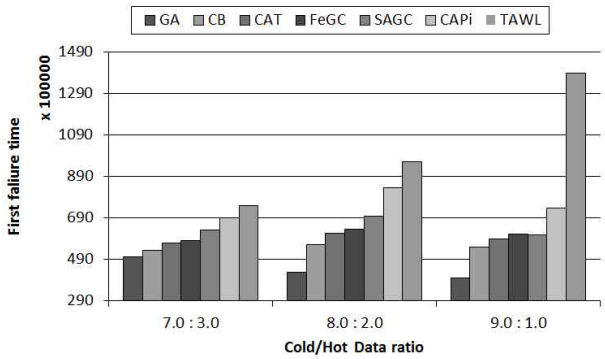

Fig. 4. First failure time

그림 4는 각 기법들의 첫 번째 배드 블록이 발생하는 시점을 나타내고 있다. 그림 4에서 보는 것과 같이 기존에 제안된 동적 마모도 평준화 기법들은 공통적으로 콜드 데이터가 많아질수록 수명이 줄어드는 것을 확인할 수 있다. 이는 동적 마모도 기법들이 공통적으로 데이터 갱신이 완전히 이루어지지 않는 콜드 블록을 가비지 컬렉션의 희생블록으로 선택하지 않기 때문에 발생한다. 따라서 이러한 단점을 해결하기 위해 제안한 기법은 시간 기반의 동적 마모도 기법에서 사용할 수 있는 정적 마모도 기법도 같이 적용하고 있다. 이를 통하여 제안한 기법은 GA에 비해 평균적으로 133% 최대 250%, CB에 비해 평균 88% 최대 160%, CAT에 비해 평균 75% 최대 131%, FeGC에 비해 평균 70% 최대 126%, SAGC에 비해 평균 60% 최대 120%, CAPi에 비해 평균 37% 최대 88% 수명이 향상되었다.

그림 5와 그림 6은 가비지 컬렉션 및 마모도 평준화 알고리즘의 오버헤드를 보여주고 있다. 그림 5는 쓰기 연산 횟수 당 삭제 연산 빈도를 보여주고 있다. 실험에서 제안한 기법은 GA

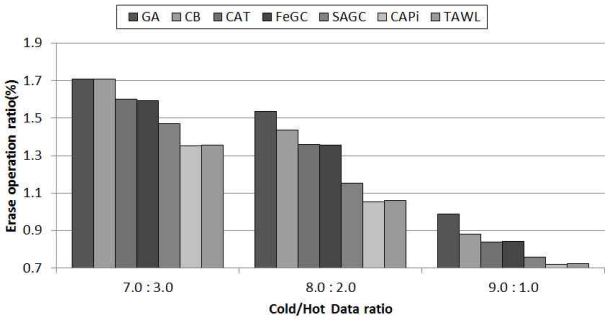

Fig. 5. Erase operation ratio per write operation

에 비해 평균적으로 34% 최대 45%, CB에 비해 평균 28% 최대 35%, CAT에 비해 평균 20% 최대 16%, FeGC에 비해 평균 19% 최대 15%, SAGC에 비해 평균 7% 최대 14% 오버헤드가 감소하였다. 다만 TAWL은 CAPi에 비해 평균 0.7%정도 오버헤드가 더 많다. 이는 가비지 컬렉션의 희생블록으로 선택되지 않는 완전 콜드 블록도 마모도 평준화에 참여시키기 위해 이주시키는 정적 마모도 평준화를 수행함에 따라 발생되는 추가적인 오버헤드이다. 결국 TAWL은 CAPi에 비해 1% 미만의 오버헤드만 추가하여 수명을 88% 늘리고 있다.

그림 6은 가비지 컬렉션 및 마모도 평준화 수행 시 발생되는 유효 페이지의 이주 연산 횟수를 보여주고 있다. GA와 CB 알고리즘을 제외한 나머지 알고리즘들은 모두 가비지 컬렉션 시에 블록 내의 유효 페이지들을 이주 시킬 때 콜드/핫 데이터를 분리하여 저장하고 있다. 이를 통하여 가비지 컬렉션 시에 발생할 수 있는 유효 페이지 복사에 대한 오버헤드를 줄이고 있으며, 콜드/핫 데이터를 분리하여 이주시키는 정책은 콜드 데이터가 많을수록 효율이 높아진다. 실험에서 TAWL은 GA에 비해 평균 17% 최대 32%, CB에 비해 평균 13% 최대 21%, CAT에 비해 평균 10% 최대 16%, FeGC에 비해 평균 9% 최대 17%, SAGC에 비해 평균 4% 최대 5% 정도 유효 페이지 이주에 따른 쓰기 연산 횟수가 감소하였으며, CAPi와는 비슷한 쓰기 연산이 발생하였다.

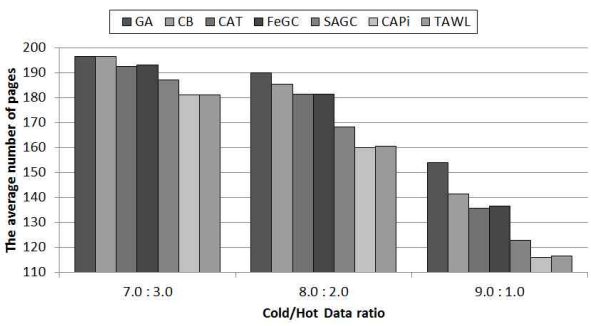

Fig. 6. The average number of page copies per erase operation

V. Conclusions

본 논문에서는 시간 기반의 새로운 하이브리드 마모도 평준화 기법을 제안하였다. 제안하는 기법은 가비지 컬렉션 수행 시 블록의 마모도를 고려하는 동적 마모도 평준화 기법과 낸드 플래시 시스템 내의 모든 블록들의 마모도를 고려하는 정적 마모도 평준화를 동시에 수행하는 하이브리드 마모도 평준화 기법이다. 또한 제안하는 기법은 가비지 컬렉션 시 콜드/핫 데이터를 분리하여 저장하며, 이를 통해 가비지 컬렉션 시 발생될 수 있는 오버헤드를 줄이고 있다. 실험을 통해 TAWL은 수명연장에서 기존의 알고리즘과 비교하여 최대 250% 향상되었음을 보였으며, 가비지 컬렉션에서 발생될 수 있는 오버헤드 또한 기존의 알고리즘에 비해 삭제 연산 횟수측면에서 최대 45%, 유효 페이지 복사 측면에서 최대 32% 감소시켰다. 다만 TAWL은 콜드/핫 데이터를 정확하게 구별하기 위해 업데이트 빈도 테이블을 사용하고 있어 추가적인 메모리 사용을 필요로 한다. 따라서 적은 메모리를 사용하여 콜드/핫 데이터 구별하는 기법에 대하여 차후 추가 연구가 필요하다.

REFERENCES

[1] Chen, Feng, David A. Koufaty, and Xiaodong Zhang. "Understanding intrinsic characteristics and system implications of flash memory based solid state drives." ACM SIGMETRICS Performance Evaluation Review. Vol. 37. No. 1. ACM, 2009.

[2] Yang, Ming-Chang, et al. "Garbage collection and wear leveling for flash memory: Past and future." Smart Computing (SMARTCOMP), 2014 International Conference on. IEEE, 2014.

[3] Wu, Michael, and Willy Zwaenepoel. "eNVy: a non-volatile, main memory storage system." ACM SigPlan Notices. Vol. 29. No. 11. ACM, 1994.

[4] Kawaguchi, Atsuo, Shingo Nishioka, and Hiroshi Motoda. "A Flash-Memory Based File System." USENIX. 1995.

[5] Chiang, M-L., and R-C. Chang. "Cleaning policies in mobile computers using flash memory." Journal of Systems and Software 48.3 (1999): 213-231.

[6] Kwon, Ohhoon, et al. "FeGC: An efficient garbage collection scheme for flash memory based storage systems." Journal of Systems and Software, vol. 84, no. 9, pp. 1507-1523, September 2011.

[7] Han, Longzhe, Yeonseung Ryu, and Keunsoo Yim. "CATA: a garbage collection scheme for flash memory file systems." Ubiquitous Intelligence and Computing. Springer Berlin Heidelberg, 2006. 103-112.

[8] Han, Long-zhe, et al. "An intelligent garbage collection algorithm for flash memory storages." Computational Science and Its Applications-ICCSA 2006. Springer Berlin Heidelberg, 2006. 1019-1027.

[9] Lin, M. W., et al. "Garbage collection policy for flash-aware Linux swap system." Electronics letters 47.22 (2011): 1218-1220.

[10] Kwon, Ohhoon, and Kern Koh. "Swap space management technique for portable consumer electronics with NAND flash memory." Consumer Electronics, IEEE Transactions on 56.3 (2010): 1524-1531.

[11] Xu, Guangxia, et al. "Garbage collection policy to improve durability for flash memory." Consumer Electronics, IEEE Transactions on 58.4 (2012): 1232-1236.

[12] Xu, Guangxia, Manman Wang, and Yanbing Liu. "Swap-aware garbage collection algorithm for NAND flash-based consumer electronics." Consumer Electronics, IEEE Transactions on 60.1 (2014): 60-65.

[13] Sang-Ho Hwang and Jong Wook Kwak, "Garbage Collection Technique for Reduction of Migration Overhead and Lifetime Prolongment of NAND Flash Memory", IEMEK Journal of Embedded Systems and Applications, Volume 11, Number 2, pp. 125~134, April 2016

[14] Chang, Yuan-Hao, Jen-Wei Hsieh, and Tei-Wei Kuo. "Improving flash wear-leveling by proactively moving static data." Computers, IEEE Transactions on Vol 59, No. 1, pp. 53-65, Jan. 2010.

[15] Liao, Jilong, et al. "Adaptive Wear-leveling in Flash-based Memory." IEEE Computer Architecture Letters, Vol 14, No. 1, pp. 1-4, Jan. 2014.

[16] Murugan, Muthukumar, and David HC Du. "Rejuvenator: A static wear leveling algorithm for NAND flash memory with minimized overhead." Mass Storage Systems and Technologies (MSST), 2011 IEEE 27th Symposium on. IEEE, pp. 1-12, May 2011.

[17] Chang, Li-Pin. "On efficient wear leveling for large-scale flash-memory storage systems." Proceedings of the 2007 ACM symposium on Applied computing, pp. 1126-1130, March 2007.

[18] Wang, Chundong, and Weng-Fai Wong. "SAW: System-assisted wear leveling on the write endurance of NAND flash devices." Design Automation Conference (DAC), pp. 1-9, May 2013.

[19] Seon Hwan Kim, Ju Hee Choi, and Jong Wook Kwak,

"HaWL: Hidden Cold Block-Aware Wear Leveling Using Bit-Set Threshold for NAND Flash Memory", IEICE Transactions on Information and Systems, Vol. E99-D, No. 4, pp. 1242~1245, April 2016

[20] Sang-Ho Hwang, et al. "Cold Data Identification using Raw Bit Error Rate in Wear Leveling for NAND Flash Memory." Journal of the Korea Society of Computer and Information, Vol. 20, No. 12, pp. 1-8, 2015.

[21] Kim, Sung Ho, et al. "EPET-WL: Enhanced Prediction and Elapsed Time-based Wear Leveling Technique for NAND Flash Memory in Portable Devices." Journal of the Korea Society of Computer and Information, Vol. 21, No. 5, pp. 1-10, 2016.

[22] V. Prabhakaran and T. Wobber, “SSD Extension for DiskSim Simulation Environment,” http://research.microsoft.com/en-us/downloads/b41019e2-1d2b-44d8-b512-ba35ab814cd4/, 2009.

Authors

Sang-Ho Hwang received the B.S. and M.S. degrees in Computer Engineering from Yeungnam University, Korea, in 2009 and 2013 respectively. His current research interests include embedded systems and non-volatile memory systems.

Jong Wook Kwak received the B.S. degree in computer engineering from Kyungpook National University, Daegu, South Korea, in 1998, and the M.S. degree in computer engineering and the Ph.D. degree in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2001 and 2006, respectively. From 2006 to 2007, he was a Senior Engineer with the system-on-chip (SoC) Research and Development Center, Samsung Electronics Company, Ltd., Suwon, South Korea. During 2011-2012, he was a Guest Researcher at the Research Institute of Advanced Computer Technology, Seoul National University. During 2012-2013, he was a Visiting Scholar at the Georgia Institute of Technology, Atlanta, GA, USA. He is currently an Associate Professor with the Department of Computer Engineering, Yeungnam University, Gyeongsan, South Korea. His current research interests include advanced processor architecture, low-power mobile embedded system design, and high-performance parallel and distributed computing.