

플래시 파일시스템에서 wear-leveling 개선을 위한

블록 할당 정책

장시웅*

*동의대학교

A Block Allocation Policy to Enhance Wear-leveling in a Flash File System

Si-woong Jang*

*Donggeui University

E-mail : swjang@deu.ac.kr

요 약

디스크에서는 데이터가 변경되면 해당 영역에 겹쳐 쓰기를 수행하나 플래시 메모리에서는 겹쳐 쓰기가 수행되지 않아 데이터가 변경될 때 새로운 영역에 데이터를 갱신한다. 따라서 이전의 데이터는 쓸모없는 데이터가 되기 때문에 가비지 컬렉션을 통해 지움 연산을 수행하여 새로운 영역을 확보하게 된다. 지움(erase) 연산을 수행할 수 있는 회수가 플래시 메모리의 특성에 의해 일정 회수로 제한을 받게 되므로 플래시 메모리의 모든 블록은 고르게 쓰여 지고 지워져야 한다.

본 논문은 지역성을 가지는 접근에서 wear-leveling을 개선하기 위한 방법으로 hot 데이터와 cold 데이터를 서로 다른 बैं크에 저장하고 시간이 흐름에 따라 일정주기로 cold बैं크와 hot बैं크를 교환하는 CB-MB(Cost Benefit between Multi Bank) 방법을 제안하고 성능을 평가하였다. CB-MB는 uniform한 작업부하에 대해서는 다른 방법들과 유사한 성능을 보이는 반면 접근 지역성을 가지는 작업부하에 대해서는 상대적으로 월등한 성능을 제공하는 것으로 분석되었다.

ABSTRACT

While disk can be overwritten on updating data, because flash memory can not be overwritten on updating data, new data are updated in new area. If data are frequently updated, garbage collection, which is achieved by erasing blocks, should be performed to reclaim new area. Hence, because the number of erase operations is limited due to characteristics of flash memory, every block should be evenly written and erased. However, if data with access locality are processed by cost benefit algorithm with separation of hot block and cold block, though the performance of processing is high, wear-leveling is not even.

In this paper, we propose CB-MB (Cost Benefit between Multi Bank) algorithm in which hot data are allocated in one bank and cold data in another bank, and in which role of hot bank and cold bank is exchanged every period. CB-MB showed that its performance was similar to that of others for uniform workload, however, the method provides much better performance than that of others for workload of access locality.

키워드

Flash File System, Wear-leveling, Garbage Collector, Database, Flash Memory

1. 서 론

많은 응용 시스템에서 플래시 메모리가 사용되고 플래시 메모리가 대형화됨에 따라 플래시 파일시스템에 관한 많은 연구가 진행되고 있다[1-5].

플래시 메모리는 디스크와는 달리 덮어쓰기가 불가능하여 데이터가 변경되면 변경된 데이터에 해당하는 이전 영역은 쓸모없게 되고, 새로운 데이터는 새로운 영역에 갱신된다. 데이터가 빈번히 변경되면 새로운 영역을 확보하기 위해 가비지

컬렉션을 통하여 새로운 영역을 확보하게 된다. 데이터를 읽고 쓰는 것은 페이지 단위로 수행하나 데이터를 지우는 연산은 블록단위로 수행된다 [6]. 데이터에 대한 가비지 컬렉션을 수행할 때 유효한 페이지는 새로운 블록으로 이동한 후 해당 데이터 블록을 지우는데, 이때 지움(erase) 연산을 수행할 수 있는 회수는 플래시 메모리의 특성에 의해 한정된 회수(약 100, 000회)로 제한을 받게 되므로 플래시 메모리의 모든 블록은 고르게 쓰여 지고 지워져서 평등한 wear-leveling이 제공되어야 한다.

데이터의 갱신, 추가, 삭제가 반복되는 동안 자유공간이 부족해지면 새로운 데이터를 추가하기 위해 자유공간을 확보하여야 하는데 이때 자유공간을 확보하기 위해 지움 연산을 수행할 블록을 선정해야한다. 지움 연산을 수행하기 위해 블록을 선정하는 전통적인 방법으로 Greedy 방법과 Cost benefit 방법이 존재하는데, Greedy 방법은 uniform한 작업 부하에 좋은 성능을 보이는 반면 접근 지역성을 가지는 작업부하에 대해서는 좋지 않은 성능을 보인다. 따라서 접근 지역성을 가지는 작업부하를 위하여 Cost benefit 방법이 사용되며, 가비지 컬렉션 수행시 cold 데이터와 hot 데이터를 분리하여 저장하면 Greedy 정책에 비해 월등한 성능을 보이는 것으로 나타났다[6].

그러나 Cost benefit 방법에서 cold 데이터와 hot 데이터를 분리하여 저장하면 wear-leveling의 편차가 커져 플래시 메모리의 수명에 큰 영향을 미치게 된다. 플래시 파일 시스템의 wear-leveling을 개선하기 위한 연구로는 [7]이 있으며, 특정한 cold 데이터가 한 블록에 오래 머무는 것을 방지하여 블록 간의 지움 연산의 회수의 차이를 최소화하는 방법을 제시하였으나 접근 지역성(access locality)을 가지는 작업부하에는 효과적이지 않다. 본 논문에서는 wear-leveling을 개선하기 위해 멀티 뱅크에서 하나의 뱅크는 cold 데이터를 다른 뱅크는 hot 데이터를 할당하고, 시간이 흐름에 따라 일정주기로 cold 뱅크와 hot 뱅크를 교환하는 방법을 제안하고 성능을 평가하였다

II. 관련 연구

플래시 메모리는 여러 개의 블록으로 구성되며 하나의 블록은 일정수의 페이지로 나뉘어 관리된다. 한 페이지의 크기는 하드 디스크의 최소 단위인 섹터의 배수로 구성된다. 플래시 메모리는 읽기와 쓰기 연산을 페이지 단위로 수행한다. 한편 쓰기 연산은 지워진 영역에만 가능하므로 데이터를 쓰기 위해서는 해당 영역을 미리 지워야하는데 지움 연산은 블록 단위로 수행된다. 플래시 메모리는 덮어쓰기(overwrite)가 되지 않으므로 데이터를 갱신하기 위해서는 갱신할 데이터에 해당하는 페이지를 무효화시킨 후 갱신된 데이터를 새로운 블록의 페이지에 써야 한다. 많은 데이터

에 대해 갱신 연산이 발생하면 무효화된 데이터가 많아져서 유효한 블록이 소진되게 된다. 따라서 자유블록의 수가 임계치 이하로 떨어지면 가비지 컬렉션 정책에 의해 특정 블록을 선정하여 유효 페이지를 새로운 페이지를 할당 받아 이동한 후 지움 연산을 수행한다.

가비지 컬렉션 정책에는 무효 페이지가 가장 많은 블록을 선정하는 Greedy 알고리즘과 블록의 age와 이용율을 고려한 Cost benefit 알고리즘이 있다. Cost benefit 알고리즘은 블록 이용율(u)과 age를 고려하여 $\frac{age(1-u)}{2u}$ 를 최대로 하는 블록을 선정[8]하여 지움 연산을 수행한다. 이 방법을 이용하면 cold 블록들은 hot 블록보다 낮은 이용율을 가짐에도 불구하고 age가 클 경우에는 지움 연산을 위한 블록으로 선정된다.

III. 다중 뱅크를 이용한 Wear-leveling 향상 기법

종래의 Cost benefit 방법은 균등한 wear-leveling을 제공하지만 접근 지역성을 가지는 작업부하에서는 Greedy 방법처럼 좋지 못한 성능을 제공한다.

3.1 CB-S (Cost Benefit with Separation)

CB-S는 hot 데이터와 cold 데이터를 분리하여 관리하는 Cost benefit 방법으로 하나의 논리적 블록에는 hot 데이터의 페이지와 cold 데이터의 페이지가 공존하는 형태로 존재한다. 그러나 논리적 페이지 데이터들을 물리적 뱅크에 할당할 때는 hot 페이지 데이터들은 hot 페이지만 포함하는 물리적 블록에 저장하고, cold 페이지 데이터들은 cold 페이지만 포함하는 물리적 블록에 저장한다.

3.2 CB-MB(Cost Benefit between Multi Bank)

CB-MB는 hot 데이터와 cold 데이터를 분리하여 비용효율적인 방법으로 관리되 hot 데이터의 페이지와 cold 데이터의 페이지를 서로 다른 물리적 뱅크에 나누어 저장하고 관리하는 방법으로 hot 데이터의 페이지와 cold 데이터의 페이지가 공존하는 형태로 존재한다. 그러나 논리적 페이지 데이터들을 물리적 뱅크에 할당할 때는 hot 페이지 데이터들은 hot 블록만 포함하는 물리적 뱅크에 저장하고, cold 페이지 데이터들은 cold 블록만 포함하는 물리적 뱅크에 저장한다.

한편, hot 블록만 포함하는 뱅크와 cold 블록만 포함하는 뱅크 모두 균등한 wear-leveling을 제공할 수 있지만 hot 블록만 포함하는 뱅크가 빨리 소진되는 현상이 나타날 것이다. 따라서 본 논문

에서는 일정한 주기로 hot बैं크와 cold बैं크의 역할을 바꾸도록 하는 방안을 CB-MB에 포함시켰다.

IV. 실험

본 논문에서 제안한 CB-MB의 성능 및 wear-leveling을 다른 기법(CB, CB-S, Greedy)과 비교하기 위해 시뮬레이션을 수행하였다.

4.1 실험 방법

실험은 2개 बैं크의 크기에 해당하는 논리적 페이지에 대한 페이지 번호를 발생시키되 발생된 홀수 페이지 번호는 번호가 하나 작은 짝수 페이지로 바꾸어 논리적인 홀수 페이지는 없도록 하여 2개의 물리적 बैं크에 할당되는 물리적인 페이지의 수는 물리적 बैं크의 절반만 차지하도록 하였다. 이때 새로운 물리적 블록이 일정개수(1/100) 이하가 되면 가비지 컬렉션이 수행되도록 하여 dirty 페이지가 많은 블록들이 클리어 되도록 하였다. 이와 같은 방법으로 작업부하를 100회 연속 발생시켜 실험을 수행하였다.

실험 수행시 사용된 파라미터는 표1과 같다. 표1의 실험 파라미터는 성능평가를 위해 통상적으로 사용되는 파라미터이다.

표 1. 실험 파라미터

특성	값	특성	값
읽기(512B)	0.125ms	페이지 크기	0.5 KBytes
쓰기(512B)	0.250ms	블록 크기	16 KBytes
지우기(16KB)	2ms	뱅크 크기	64 MBytes

본 논문에서 제안하는 CB-MB의 성능을 상대적으로 평가하기 위해 기존에 제안된 Greedy, CB, CB-S 방법들의 성능을 함께 평가하여 비교한다. 작업부하는 [6]에 있는 것과 유사한 방법으로 지역성을 가지는 작업부하와 지역성을 가지지 않는 작업부하를 4개의 등급으로 나누어 실험한다. 참조의 지역성을 표시하기 위해 x/y로 표현하는데, 이것은 y%의 데이터에 x%의 접근이 이루어지는 것을 의미한다. 따라서, 90/10의 작업부하는 10%의 데이터에 90%의 참조가 이루어지고, 90%의 데이터에는 10%의 참조가 발생하는 것을 의미한다.

4.2 성능 평가 결과

4.1절의 실험 방법으로 Greedy(GR), Cost Benefit(CB), CB-S 및 CB-MB에 대해 실험한 가비지 컬렉션 수행시간과 wear-leveling을 비교하여 분석하면 각각 그림 1 및 표 2와 같다. 접근 지역성이 없을 경우(50/50)에는 성능 측면에서 4가지 방법이 모두 유사한 결과를 보이는 반면,

wear-leveling의 측면에서는 3가지 방법(CB, CB-S, CB-MB)은 좋은 결과를 보이지만 GR은 좋지 않은 결과를 보인다. 접근 지역성이 높아질수록 GR과 CB의 성능은 급격히 나빠지는 반면 CB-S의 성능은 완만하게 떨어지고 CB-MB의 성능은 다소 좋아지는 것으로 나타났다.

CB는 접근 지역성이 높아져도 wear-leveling은 좋은 반면 성능 측면에서는 가장 나쁜 결과를 보인다. CB-S는 접근 지역성이 높아지면 성능은 양호한 반면 wear-leveling은 좋지 않은 결과를 나타낸다. 한편 본 논문에서 제안한 CB-MB 방법은 접근지역성이 증가함에 따라 성능은 더 좋아지고 wear-leveling 또한 양호한 결과를 보여 준다.

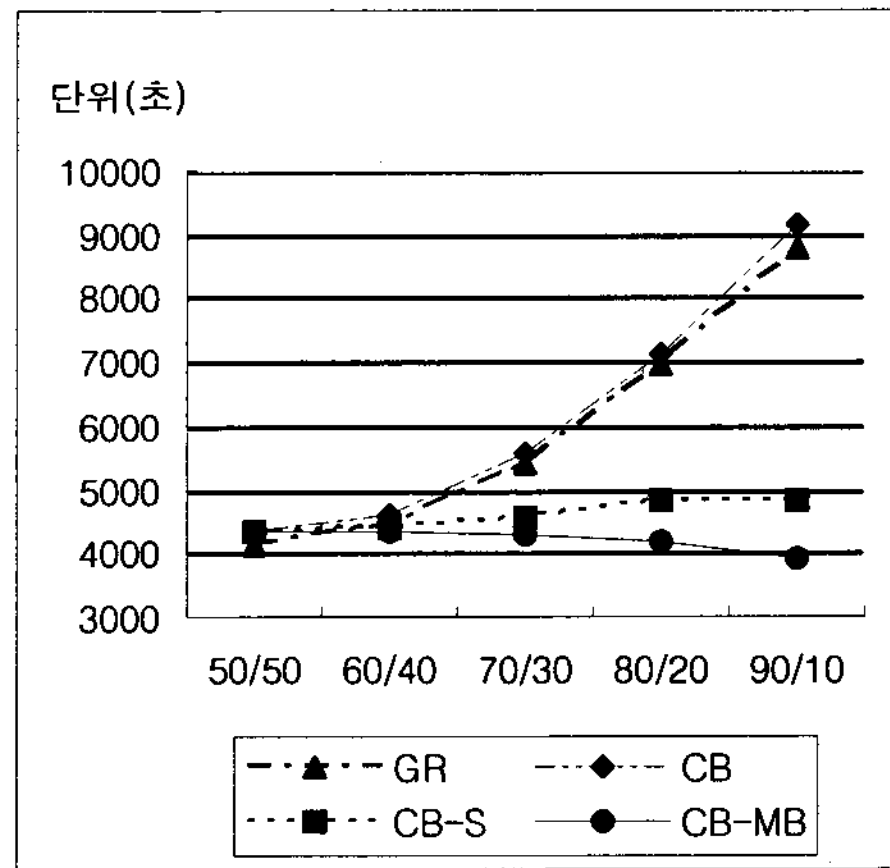


그림 1. 지역성에 따른 GC 수행시간

표 2. wear leveling의 비교

	GR	CB	CB-S	CB-MB
50/50	2.97	0.82	0.81	0.82
60/40	6.51	0.86	1.38	0.83
70/30	8.10	0.99	2.53	0.95
80/20	16.01	1.19	4.08	1.30
90/10	13.87	1.45	6.62	2.02

CB-MB의 문제점은 hot बैं크와 cold बैं크의 역할 교환에 따른 오버헤드인데, 표 3은 बैं크 역할 교환에 따른 오버헤드를 보여준다. 교환주기가 평균 쓰기 200회 이상인 경우 다른 방법에 비해 모두 우수한 성능을 보이는 것으로 판단되므로 बैं크 역할 교환 주기 200회, 100회, 50회, 25회, 12.5회인 경우의 가비지컬렉션과 wear-leveling의 상대적인 오버헤드를 비교하였다.

표 3에서 보는 것처럼 교환주기가 평균쓰기 200회인 경우에 비해 평균쓰기 100회인 경우는 성능측면과 wear-leveling 측면 모두 1% 정도 좋지 않은 것으로 나타나고, 평균쓰기 50회가 교환 주기인 경우에는 성능측면에서는 4%, wear-

leveling 측면에서 2% 좋지 않은 것으로 나타나므로 블록의 교환 주기를 평균 쓰기 50회 이상으로 하면 바람직할 것으로 판단된다.

표 3. 블록 교환의 오버헤드

교환주기	GC time	wear-leveling	블록 지움 회수 (최소, 최대)
200회	7946(1.00)	2.71	233, 250 (1.00)
100회	8048(1.01)	3.03	233, 252 (1.01)
50회	8249(1.04)	3.60	233, 255 (1.02)
25회	8653(1.09)	4.58	234, 262 (1.05)
12.5회	9527(1.20)	5.70	238, 273 (1.09)

V. 결 론

본 논문은 hot 데이터와 cold 데이터를 분리하여 저장하여 관리하는 Cost benefit 방법(CB-S)의 단점인 wear-leveling을 개선하기 위한 방법으로 hot 데이터와 cold 데이터를 서로 다른 블록에 저장하여 관리하고 일정주기로 블록의 역할을 교환하는 방법을 제안하고 성능을 평가하였다. hot 데이터와 cold 데이터를 블록단위로 분리하여 저장하여 관리하는 Cost benefit 방법은 좋은 성능을 보이지만 균등하지 않은 wear-leveling을 제공하는 것이 단점으로 지적된 반면, 본 연구에서 제안한 CB-MB 방법은 접근지역성을 보이는 작업부하에 대해 다른 방법들에 비해 좋은 성능을 보임과 동시에 균등한 wear-leveling을 보이는 것으로 나타났다. 특히 CB-MB는 uniform한 작업부하에 대해서는 다른 방법들과 유사한 성능을 보이는 반면 접근 지역성을 가지는 작업부하에 대해서는 상대적으로 월등한 성능을 제공하는 것으로 분석되었다. 이는 접근 지역성이 높아질수록 hot 데이터와 cold 데이터가 명확히 구분되어 가비지 컬렉션 시 유효 데이터의 복사가 줄어들기 때문인 것으로 분석되었다.

참고문헌

[1] Jen-Wei Hsieh, Li-Pin Chang and Tei-Wei Kuo. "Efficient On-line Identification of Hot Data for Flash-Memory Management", In SAC, pp.838-842, 2005.

[2] Y. Ryu and K. Lee. Improvement of space utilization in nand flash memory storages. Lecture Notes in Computer Science, 3820:766-775, 2005.

[3] Chang, L, Kuo, T., and Lo, S., "Real-Time Garbage Collection for Flash-Memory Storage Systems of Real-Time Embedded Systems," In

ACM Trans. on Embedded Computing Systems, November 2004, pp. 837-863

[4] Li-Pin Chang, Tei-Wei Kuo and Shi-Wu Lo. "A Real-Time Garbage Collection for Flash-Memory Storage Systems of Real-Time Embedded Systems." ACM Trans. in Embedded Computing Systems, Vol.3, No.4, pp.837-863, 2004.

[5] ERAN GAL AND, SIVAN TOLEDO, "Algorithms and Data Structures for Flash Memories", ACM Computing Surveys, Vol.37, Issue.2, pp.138-163, June, 2005.

[6] L.Z Han, Y.S Ryu, T.S Chung, M.H Lee, S.W Hong, "An Intelligent Garbage Collection Algorithm for Flash Memory Storages," Lecture Notes in Computer Science, 3980:1019-1027, 2006.

[7] Atsuo Kawaguchi, Shingo Nishioka, and Hiroshi Motoda, "Flash Memory Based File System," Proceedings of USENIX Technical Conference, New Orleans, LA, pp.155-164, 1995.