큰 페이지를 가지는 낸드 저장장치를 위한 혼합사상기반의 플래시 변환 계층 연구

이재훈°, 김명석, 이성진*, 김지홍
서울대학교 컴퓨터공학부, *인하대학교 컴퓨터정보공학과
Jhoonl89@davinci.snu.ac.kr, morssola75@davinci.snu.ac.kr, sungjin.lee@inha.ac.kr, jihong@davinci.snu.ac.kr

A Hybrid FTL for NAND Flash Storages with Large Pages

Jaehoon Lee, Myoungseok Kim, Sungjin Lee, Jihong Kim
Department of Computer Science and Engineering, Seoul National University
*Department of Computer Science and Information Engineering, Inha University

요 약

최근의 대용량 낸드 플래시 메모리는 매우 큰 물리적 페이지를 가지는데 이러한 큰 페이지를 가지는 낸드 플래시 저장장치에서는 작은 크기의 쓰기 요청을 처리하는데 많은 어려움이 발생한다. 본 논문에서는 이러한 대용량 낸드 플래시 메모리가 가지는 문제점을 효과적으로 극복할 수 있는 새로운 하이브리드 FTL을 제안한다. 제안된 하이브리드 FTL은 기존의 페이지 기반 FTL의 성능을 개선시키는 동시에 매핑 테이블의 크기를 증가시키지 않는 장점을 가지고 있다. 시뮬레이션을 통한 평가 결과 기존의 페이지 기반 FTL에 비해서 약 3% 증가 된 매핑 테이블의 크기로 18% 증가 된 성능을 얻게 되었다.

1. 서 론

낸드 플래시 기반 저장장치의 용량은 공정의 미세화와 Multi-Level Cell 기술이 발전함에 따라 점점 커지는 추세이다. 이러한 기술을 적용하여 낸드 플래시 기반 저장장치 용량을 늘리는 것은 물리적 페이지 크기의 증가를 필연적으로 수반하게 된다. 실제로 최근에 출시 된 낸드 플래시 기반 저장장치의 물리적 페이지 크기는 16KB인 경우가 대부분이다.

낸드 플래시 메모리의 읽기/쓰기 기본 단위는 물리적 페이지 단위이다. 따라서 페이지 크기의 증가는 저장장치의 대역폭 증가를 의미하며, 이는 저장장치의 성능이 향상될 것이라는 예상을 하게 된다. 하지만 예상과는 달리 페이지 크기가 큰 낸드 플래시 메모리는 빈번한 read-modify-write에 의해 성능 감소 현상을 겪게 된다. 예를 들어, 하나의 물리적 페이지의 일부가 쓰이고 뒤이어 또다른 일부가 쓰일 경우 이전에 쓰여진 페이지를 먼저 읽고 새로 쓰여질 부분과 병합한 뒤 새로운 페이지를 할당 받아 쓰는 과정을 반복하게 된다. 결국, 한번의 쓰기 연산을 위해 추가적인 읽기 연산이 발생하여 낸드 플래시 메모리의 성능을 감소시키는 결과를 낳게 되며 이러한 경향은 작은 크기의 쓰기 요청이 많은 워크로드에서 더욱 심해지게 된다.

이러한 문제를 해결하기 위한 가장 단순하고 효율적인 방법은 매핑 단위를 물리적 페이지 크기보다 작은 서브페이지 단위로 유지하는 서브페이지 단위 매핑 기법을 사용하는 것이다. 기존의 페이지 단위로 매핑 정보를 유지하는 경우 서브페이지의 위치가 고정되어 있기 때문에 read-modify-write이 발생하게 되지만, 서브페이지 단위로 매핑 정보를 유지하는 경우에는 이러한 제약조건이 사라지기 때문에 추가적인 읽기 연산이 발생하지 않게 된다.

하지만 서브페이지 단위 매핑 방식은 매핑 테이블을 유지하기 위해 필요한 메모리공간의 크기가 기존 페이지 기반 매핑 방식에 비해서 수 배나 더 커지게 된다는 단점이 존재한다. 예를 들어 물리적 페이지 크기가 16KB인 낸드 플래시 메모리에서 매핑 단위를 4KB로 유지하기 위해서는 기존 대비 4배 더 많은 메모리가 필요하게 된다.

본 연구에서는 기존의 두 기법의 단점을 제거하고 장점만을 유지하는 하이브리드 FTL을 제안한다. 페이지 단위로 매핑 정보를 유지하는 페이지 단위 매핑 방식에서 발생하던 잦은 read-modify-write 문제의 경우, 작은 크기의 쓰기 요청인 경우에만 서브페이지 단위 매핑을 사용하여 관리하는 방법으로 해결하였다. 또한 서브페이지 단위 매핑 방식에서 발생하던 매핑 테이블 크기 문제의 경우, 해시 테이블을 활용하여 서브페이지 별 매핑 엔트리를 유지하는 방법으로 해결하였다.

추가적으로 관련 연구에 의하면 작은 크기의 쓰기 요청은 자주 업데이트 되며 따라서 보유 기간(Retention time)이 짧은 것으로 밝혀진 바 있다[1][2]. 이러한 경향에 의하여 본 기법을 적용하게 되면, 작은 크기의 쓰기 요청으로 저장 된 데이터는 핫 데이터로 분류되고 그 중 가비지 컬렉션 (Garbage Collection, GC) 기간에 업데이트 되지 않은 데이터는 콜드 데이터로 분류되는 핫/콜드 데이터 분리 효과가 발생한다.

시뮬레이션 결과, 작은 크기의 쓰기 요청을 구분하여 관리함으로써 일반 페이지 매핑에서 발생하던 read-modify-write이 완전히 사라지게 되었으며, 핫/콜드 데이터를 독립적인 블록에 유지시키는 효과로 인해 가비지 컬렉션 발생 시의 유효 페이지 복사 횟수가 최대 42% 줄어들게 되었다. 또한 작은 크기의 쓰기 요청이 많은 워크로드의 경우, 유지되는 해시 테이블 엔트리의 개수가 일반 페이지 매핑 엔트리와 비교하여 약 3% 정도로 유지 되기 때문에 추가적인 매핑 정보 유지에 따른 메모리 사용 오버헤드는 거의 존재하지 않았다.

본 논문의 구성은 다음과 같다. 2장에서는 제안 된

하이브리드 FTL의 구조 및 동작 원리에 대하여 설명한다. 그리고 3장에서 서브페이지 단위 FTL, 페이지 단위 FTL과 제안 된 하이브리드 FTL을 비교 분석한 후, 4장에서 결론을 맺는다.

2. 하이브리드 FTL의 구조 및 동작

제안하는 하이브리드 FTL은 물리적 페이지 크기보다 작은 크기의 쓰기 요청이 빈번하게 발생하는 최근의 워크로드를 타겟으로 설계하였으며, 오픈 플래시 개발 플랫폼인 host-level FTL에서 구현되었다[3]. FTL이 관리하는 타겟 SSD의 물리적 페이지 크기는 16KB이다. 아래 그림 1은 하이브리드 FTL의 대략적인 구조를 보여준다.

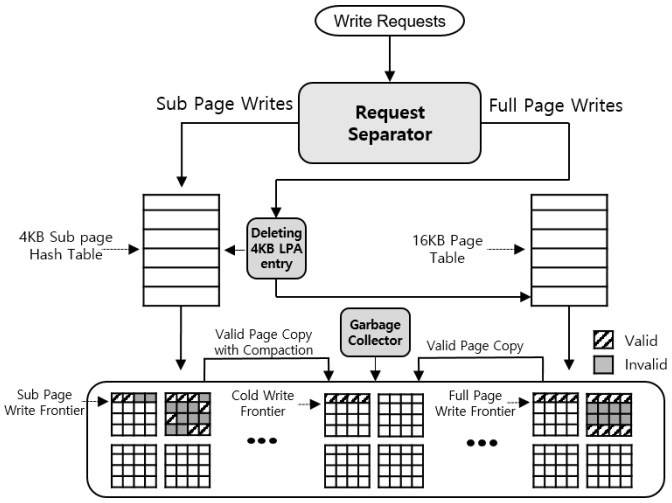

그림 1 하이브리드 FTL의 구조

쓰기 요청 처리는 짧은 보유 기간 속성을 가진 작은 크기의 경우 서브페이지 블록에 할당하고, 물리적 페이지 크기인 경우 일반 블록에 할당하며 진행한다. 두 종류의 쓰기 요청은 서로 다른 활성 블록들의 물리적 페이지 주소를 할당 받게 되며, 이에 대한 매핑 정보를 유지하기 위해 기존 16KB 매핑 테이블에 더하여 해시 테이블로 관리되는 4KB 매핑 테이블을 추가적으로 유지한다.

2.1. 쓰기 요청의 재구성 및 하이브리드 매핑 기법

하이브리드 FTL은 쓰기 요청이 들어오는 순간, 쓰기 요청의 크기를 기준으로 요청을 분류한다. Request Separator 모듈이 이러한 기능을 수행하게 되는데 만약 쓰기 요청의 크기가 물리적 페이지 크기와 동일하다면 일반 쓰기 요청으로 분류하고, 그보다 작다면 read-modify-write의 발생을 막기 위해 작은 크기의 쓰기 요청으로 분류한다. 구분 된 쓰기 요청은 서로 다른 과정을 거치면서 쓰기 작업이 이루어지는데, 그 과정은 다음과 같다.

작은 크기의 쓰기 요청은 최초 서브페이지 활성 블록에서 새로운 물리적 페이지를 할당 받게 되고 해시 테이블에서 이에 대한 매핑 작업을 수행한다. 4KB LPA를 키 값으로 하는 엔트리가 기존에 존재하고 있다면, 서브페이지 오프셋을 포함하는 PPA만을 수정하고 없다면 해시 엔트리를 생성한다.

일반 쓰기 요청은 Request Separator에 의해 16KB 단위의 LPA와 4KB 단위의 LPA 4개를 모두 유지하고 있으며, 각각의 4KB LPA가 기존에 해시 테이블에 존재하는지 검색하고 존재한다면 해시 엔트리를 삭제한 후 16KB 페이지 매핑 테이블을 갱신한다.

이러한 두가지 방식의 매핑 방식으로 페이지 엔트리를 관리하게 되면, 해시 테이블에는 항상 가장 최근의 쓰여진 4KB LPA가 유지되게 되고 해시 테이블에 존재하지 않는 LPA의 경우에는 쓰여지지 않았거나 기존 16KB 페이지 매핑 테이블에서 보통의 쓰기 방식으로 기록 되어 있음을 의미한다.

2.2. 혼합사상기반의 가비지 컬렉션

하이브리드 FTL은 greedy정책으로 희생 블록(Victim block)을 선택하며, 선택 된 희생 블록에 존재하는 유효 페이지를 자주 업데이트 되지 않는 콜드 데이터로 간주한다. 따라서 가비지 컬렉터(Garbage Collector)는 유효 페이지에 기록 된 데이터를 모두 읽은 후, 예약 된 콜드 블록에 쓰기를 진행한다. 희생 블록은 서브페이지 단위로 매핑 된 작은 크기의 쓰기가 진행 된 블록도 가능하며, 물리적 페이지 크기로 매핑 된 일반 쓰기가 진행 된 블록 또한 가능하다. 제안 된 FTL은 채널 단위의 병렬성을 지원하기 때문에 각 채널당 희생 블록을 선택하며, 이 때의 희생 블록은 두 종류의 블록이 혼합되는 경우를 허용한다. 그림 2는 채널 당 희생 블록을 선택하고 유효 페이지를 읽은 후, 이를 예약 된 콜드 블록에 쓰는 과정을 보여준다.

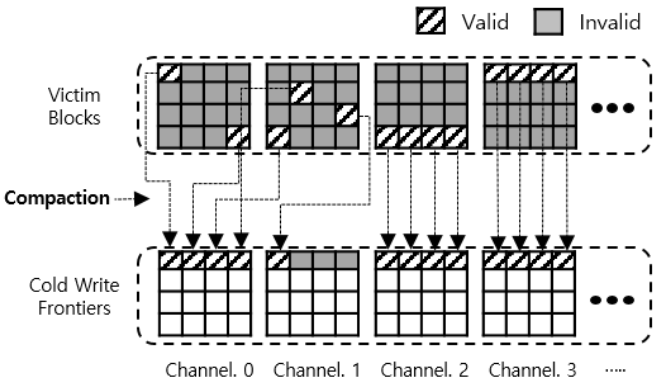

그림 2 콜드 데이터를 위한 가비지 컬렉션

하이브리드 FTL은 OOB(Out-of-band) 영역에 기록 된 LPA 정보를 이용하여 해당 물리적 페이지의 매핑 상태를 구분할 수 있다. 따라서 희생 블록에 존재하는 유효 페이지들을 읽어드린 후 각 페이지의 매핑 타입을 파악하며, 서브페이지 단위로 매핑 된 작은 크기의 데이터를 물리적 페이지 크기만큼 압축하여 예약 된 콜드 블록에 쓰기를 진행한다. 물리적 페이지 단위로 매핑 된 경우, 압축없이 그대로 콜드 블록에 쓰여지게 된다. 본 기법을 적용하게 되면, 서브페이지 단위로 매핑을 유지하는 서브페이지 단위 기법의 단점 중 하나인 물리적 페이지의 내부 단편화 현상을 부분적으로 해결할 수 있다. 또한 호스트로부터 넘어온 작은 크기의 쓰기 요청을 구분하여 처리하는 방식에 의해 핫 데이터를 그룹화

하는 효과와[1][2], 가비지 컬렉션 기간에 선택 된 모든 콜드 데이터를 한 장소에 모아두는 콜드 데이터 그룹화 효과가 발생한다.

3. 실험결과

실험은 16GB SSD에서 평가 하였으며[3], 워크로드의 경우 업데이트와 read-modify-write 비율이 5:5인 YCSB 벤치마크를 대상으로 결과를 측정하였다. 그림 3은 YCSB 벤치마크에 대하여 기존의 두 기법과 제안 된 기법의 IOPS를 비교한 실험 결과이다.

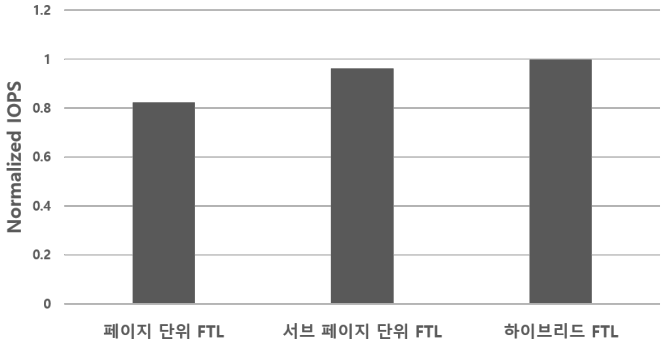

그림 3 YCSB 벤치마크 수행에 따른 IOPS

결과에서 볼 수 있듯이, 제안 된 하이브리드 FTL에서는 페이지 단위 FTL의 잦은 read-modify-write 문제를 해결함으로써 기존 페이지 매핑 FTL 대비 18% 향상된 성능을 보여준다. 또한 서브페이지 단위 FTL과 비교하여 매핑 테이블 사이즈를 4배 가까이 줄임과 동시에 성능은 그대로인 수준을 유지하였다. 다음 그림 4는 세 종류의 FTL에서 유지되는 매핑 테이블의 크기를 보여준다.

FTL 종류	엔트리 개수	테이블 크기
페이지 단위 FTL	1,048,576	32 MB
서브페이지 단위 FTL	4,194,304	128 MB
하이브리드 FTL	1,085,753	36.12 MB

그림 4 YCSB 벤치마크 수행 시 필요한 매핑 테이블 크기

16GB SSD에서 유지되어야 할 페이지 별 엔트리의 수는 1,048,576개며 각 엔트리의 크기는 32bytes이다. 해시 테이블 엔트리를 유지하기 위해 56bytes가 추가적으로 사용된다. YCSB benchmark의 경우 해시 엔트리의 개수는 37,177개 이므로 제안 된 하이브리드 FTL은 기존 페이지 단위 FTL에 비교하여 약 3.4%의 메모리 공간만을 추가적으로 사용하게 된다.

다음 그림 5는 YCSB 벤치마크의 가비지 컬렉션 부하에 대한 결과이다. 핫 콜드 데이터 분리 현상으로 인해 하이브리드 FTL의 경우 서브페이지 매핑 FTL과 비교하여 GC호출 횟수는 16% 줄어들게 되었고 GC 호출 중에 선택 된 희생 블록의 유효 페이지 복사 횟수는 42%가 감소하였다.

추가적으로 16KB 쓰기 요청이 많은 TPC-C 벤치마크의 경우에도 IOPS 및 메모리 사용량이 YCSB 벤치마크와 비슷한 양상을 보였다. 다만 작은 크기의 쓰기 요청이 적은 TPC-C의 경우 성능향상의 폭은 높지 않았다.

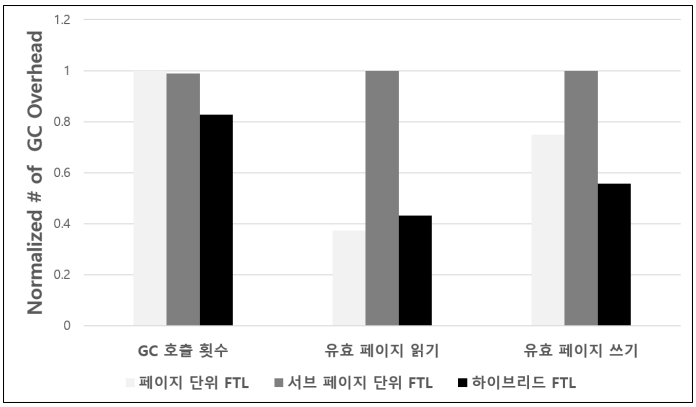

그림 5 YCSB 벤치마크 수행에 따른 GC 부하

4. 결론 및 향후 연구

본 연구에서는 물리적 페이지의 크기가 커지는 최근의 낸드 플래시 저장장치를 운용하는데 발생하는 기존의 FTL 매핑 기법들의 문제점들을 해결하는 하이브리드 FTL을 개발하였다. 결과적으로 페이지 매핑 FTL에서 발생하는 read-modify-write이 전혀 발생하지 않게 되었고, 이에 따라 IOPS가 서브페이지 단위 FTL과 동등한 수준으로 향상되었다. 또한 서브페이지 단위 FTL의 단점인 매핑 테이블 사이즈는 페이지 단위 FTL 수준으로 줄일 수 있었다. 또한 핫 콜드 데이터 분리 현상으로 가비지 컬렉션 발생 횟수 및 유효 페이지 복사 횟수가 서브페이지 단위 FTL과 비교하여 각각 16%, 42% 감소시켰다.

하지만 낸드 플래시 메모리의 모든 논리적 주소에 대하여 작은 크기의 쓰기 요청이 반복되면 최악의 경우 서브페이지 단위 FTL 이상의 매핑 테이블 사이즈가 된다. 이러한 현상을 해결하는 것이 향후 연구할 과제이다.

감사의 글

이 연구를 위해 연구장비를 지원하고 공간을 제공한 서울대학교 컴퓨터연구소에 감사드립니다. 이 논문은 2016년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2015M3C4A7065645).
(교신저자: 이성진)

참 고 문 헌

[1] L.-P. Chang et al. Hybrid Solid-State Disks: Combining Heterogeneous NAND Flash in Large SSDs. In Proc. Design Automation Conf., 2008.

[2] S. Lee et al. LAST: Locality-Aware Sector Translation for NAND Flash Memory-Based Storage Systems. In Proceedings of the International Workshop on Storage and I/O Virtualization, Performance, Energy, Evaluation and Dependability (SPEED2008), Feburary 2008.

[3] S.-W. Jun et al. BlueDBM: An Appliance for Big Data Analytics. In Proc. Int. Symp. Comput. Archit., 2015.