MADE(Min-hash Assisted Delta-compression Engine): 델타 압축 기반의 NAND 플래시 저장장치의 내구성 향상 기법

권혁준 ○, 김도현, 박지성, 김지홍

서울대학교 컴퓨터공학부

kwonhjs@snu.ac.kr, dohyun21@snu.ac.kr, jspark@davinci.snu.ac.kr, jihong@davinci.snu.ac.kr

Improving the Lifetime of NAND Flash-based Storages by Min-hash Assisted Delta-compression Engine(MADE)

Hyoukjun Kwon, Dohyun Kim, Jisung Park, Jihong Kim

Department of Computer Science and Engineering

Seoul National University

요 약

본 연구에서는 쓰기 데이터양 감소를 통해 낸드 플래시 기반 저장장치의 수명향상을 도모할 수 있는 MADE(Min-hash Assisted Delta-compression Engine)을 제안한다. MADE 모듈은 델타압축기법을 통해 중복되는 데이터 패턴을 최소화하여 실제 낸드 플래시에 인가되는 쓰기 명령 횟수를 획기적으로 줄일 수 있으며, 중복제거기법 및 무손실압축기법을 통합한 것과 유사한 효과를 보이도록 설계되었다. 또한 델타압축기법 과정 중 필요한 참조 페이지 탐색 및 압축 기법을 최적화하여, 저장되는 데이터양을 최대한 줄이는 동시에 부가적인 오버헤드를 줄이는 데 중점을 두었다. 시뮬레이션 결과, MADE가 적용된 플래시 변환 계층(Flash Translation Layer, FTL)은 실제 낸드 플래시 칩에 저장되는 데이터를 최소 50% 줄일 수 있었으며, 순차적인 중복제거기법과 무손실압축기법을 적용한 경우에 비해 추가적으로 12%의 쓰기 데이터양을 감소시킬 수 있었다.

1. 서 론

낸드 플래시 메모리는 여러 장점들로 인해 여러 컴퓨팅 영역에서 하드디스크를 대체하고 있지만, 공정의 미세화와 Multi-Level Cell 기술의 적용으로 인해 내구성이 갈수록 악화되고 있는 추세이다. 초기 낸드 플래시 기반 저장장치의 수명향상을 위한 연구로는 물리적 페이지의 할당, 가비지 컬렉션(Garbage Collection, GC), 그리고 쓰기 평준화(wear-leveling)와 같은 간접적인 방법들이 주를 이루었지만, 최근 낸드 플래시 메모리의 내구성 악화 정도가 심해짐에 따라, 저장되는 데이터양을 직접적으로 줄이는 다양한 방법들이 제안되고 있다.

이미 하드디스크 기반 저장장치 시스템에서 공간 절약의 목적으로 이루어져 온 중복제거기법 (deduplication), 무손실압축기법(lossless-compression), 그리고 델타압축기법(delta-compression)들을 FTL에 적용하는 연구들이 진행되었으며, 해당 기법들을 통해 저장장치에 인가되는 실제 데이터양을 감소시켜 주목할 만한 수명 향상을 이룰 수 있었다[1][2][3]. 이와 같이 쓰기 데이터양 감소를 위한 개별적 기법은 다양하게 제시되었지만, 더 큰 수명 향상을 위해 여러 기법들을 통합하는 연구는 상대적으로 충분히 이뤄지지 않은 실정이다. 중복제거기법, 무손실압축기법, 그리고 델타압축기법은 대상으로 하는 데이터의 범위와 기법의 효과 측면에서 차이점을 갖기 때문에, 세 기법을 통합하여 적용하면 저장되는 데이터의 양을 크게 줄일 수 있다. 하지만 압축을 위한 기법에 요구되는 성능 및 자원 오버헤드가 매우 크기 때문에 실시간 저장장치에 이를 통합 적용하기 위해서는 효과적이고 최적화된 기법의 통합 설계가 필요하다.

본 연구에서는 다양한 쓰기 데이터양 감소 기법들을 통합적으로 고려한 새로운 Min-hash Assisted Delta-compression Engine (MADE)을 제안한다. MADE 모듈은 기본적으로 새로 저장할 데이터와 유사한 데이터를 가지는 참조 페이지들을 이용하여 압축하는 델타압축기법을 통해 쓰기 데이터양을 획기적으로 줄일 수 있다. 또한, 압축 메커니즘의 최적화를 통해 중복제거기법 및 무손실압축기법의 효과를 보일 수 있도록 설계되었으며,

동시에 모듈이 SSD에 적용될 경우 발생할 수 있는 추가적인 오버헤드를 최적화하여 FTL상에서 시간 및 공간적인 제약 사항을 완화하는 효과를 역시 얻을 수 있었다. 시뮬레이션 결과, MADE가 적용된 FTL은 실제 낸드 플래시 칩에 저장되는 데이터 양을 기본적인 페이지 레벨 FTL보다 최대 95%까지 감소시킬 수 있었으며, 중복 제거기법과 무손실압축기법을 순차적으로 적용한 FTL과 비교했을 때도 훨씬 적은 오버헤드로 평균 35%의 쓰기 횟수를 추가적으로 줄일 수 있었다.

2. 작동 알고리즘 및 구조

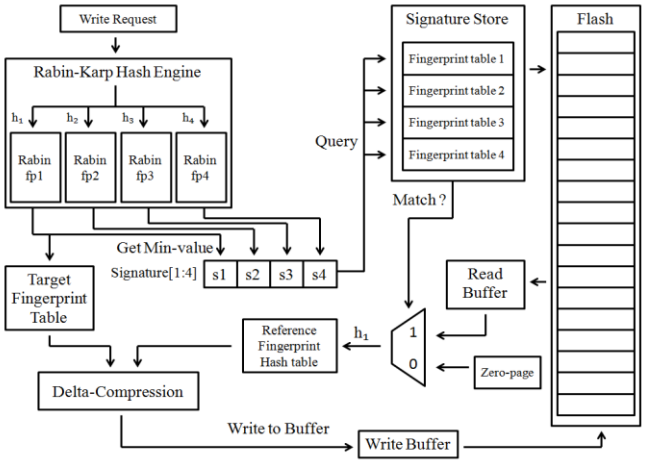

그림 1 MADE 모듈이 적용된 FTL의 구조

그림 1은 MADE 모듈이 적용된 FTL의 대략적인 구조를 보여준다. MADE 모듈은 크게 페이지의 signature를 계산하는 모듈(Rabin-Karp hash engine), signature를 저장하고 탐색하는 모듈(Signature store), 그리고 델타압축 모듈로 구성된다. MADE 모듈이 적용된 FTL에서 쓰기 명령의 처리과정은 다음과 같이 이뤄진다. 쓰기 명령이 인가되면 우선 참조 데이터를 검색하기 위해 데이터의 유사도 힌트인 signature를 계산한다. 이미 저장된 데이터들의 signature들과의 비교를 통해 요청된 데이터와 유사한 데이터를 갖는 참조 페이지를 찾아내면, 델타압축 모듈을 이용하여 데이터를 압축한다. 만약 참조 페이지를 찾지 못했을 경우, 모든 데이터가 '0'인 페이지를 참조 데이터로 삼아 압축을 시도한다. 압축된 결과 데이터는 쓰기 버퍼(Write Buffer)에 저장되며, 쓰기 버퍼에 저장된 데이터 크기가 한 물리 페이지 이상이 되면 낸드 플래시 칩에 저장된다.

일반적인 델타압축기법에서와 같이, 참조데이터의 탐색 및 델타압축 시 데이터 인코딩의 방법이 MADE 모듈의 성능 및 데이터 압축률에 영향을 주는 주요 요인이며, 본 연구에서는 이들을 낸드 플래시 기반 저장장치의 특성을 고려하여 최적화함으로써, 실제 낸드 플래시 칩에 저장되는 데이터양을 줄임과 동시에 기법 적용에 따른 오버헤드를 최소화 하였다.

2.1 유사한 데이터를 갖는 참조 페이지의 탐색

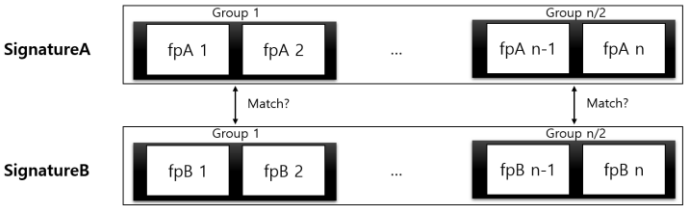

그림 2 Signature의 비교방식

MADE 모듈에서는, 델타압축기법 시 필요한 유사 데이터를 갖는 참조 페이지 탐색을 위해 Rabin-Karp fingerprint 기반의 min-hash 기법을 사용한다. 쓰기 요청된 페이지는 Rabin-Karp 해시함수를 통해 모든 부분 문자열(substring)에 대한 해시 값이 구해지게 되는데, 이중 최소값, 즉 min-hash만을 fingerprint로 삼게 된다. 이 경우 페이지가 비슷할 수록 해시 값이 일치할 확률이 높아지는 LSH(Locality Sensitive Hash)의 특성을 갖는다[5].

MADE 모듈에서는 참조 데이터의 유사성 정도를 조절 가능하게 하기 위해, 다수의 Rabin-Karp 해시함수를 사용하여 한 페이지의 fingerprint를 여러 개 계산하고, 이를 조합하여 signature를 구성한다. 그림 2는 복수의 min-hash를 통해 signature를 생성하는 과정을 예로 보여준다. 이 때 사용한 min-hash의 개수 n과 한 그룹에 들어있는 fingerprint의 개수b를 변화시킴으로써 유사도 s와 비슷하다고 판정할 확률 p의 관계를 아래와 같이 조절할 수 있다.

$$p = 1 - (1 - (1 - s)^b)^{n/b}$$

MADE에서는 시스템 상의 오버헤드를 고려해 네 종류의 min-hash값을 두 개씩 한 쌍으로 묶어 signature로 삼아, 비슷한 페이지로 판정할 확률을 유사도 30% 이하일 때 20% 이하, 유사도 60% 이상일 때 60% 이상이 되도록 설정하였다.

2.2 델타압축 시 데이터 인코딩 최적화

MADE 모듈에서는 델타압축을 위한 압축 알고리즘으로 높은 압축률과 구현 상의 오버헤드가 상대적으로 적은 VCDIFF[6]를 선정하였다. 표준 VCDIFF 인코딩 방식에서는 코드에 기록하는 동일 패턴 반복횟수 및 동일 문자열 길이의 최대값을 제한하고 있다. 이런 방식은 반복횟수와 문자열의 길이를 기록하지 않고 정의된 명령어코드만으로 짧게 기록할 수 있다는 장점이 있지만 동일한 데이터가 길게 나타날 경우, 하나의 코드로 표현할 수 있는 데이터를 여러 개의 코드로 쪼개서 표현해야 하는 문제점이 있다.

일반적으로 저장장치에 쓰기 요청을 통해 들어오는 데이터의 상당 부분이 동일한 데이터 패턴을 갖고 있으므로, 위와 같이 제한된 명령어세트를 사용하는 방식은

비효율적으로 작동할 가능성이 높다[2]. 따라서 MADE 모듈에서는 가변적으로 동일 데이터의 길이나 횟수를 명시할 수 있는 방식을 사용하였다. 이를 통하여 동일 패턴의 반복횟수와 동일 문자열 길이의 제한을 최대 32KB의 낸드 플래시 페이지에 대응할 수 있도록 연장할 수 있었다.

2.3 Min-hash와 델타압축기법 의 연산

MADE의 델타압축기법은 sliding window방식을 기반으로 하는 압축 기법으로, 압축 알고리즘 상 현재 window 내의 부분문자열과 일치하는 부분문자열을 찾는 것이 핵심이다[4]. 이를 보다 빠르게 하기 위해 데이터의 각 부분문자열에 대한 해시 값을 계산하게 되는데, 이 과정에서 참조 데이터 검색을 위해 사용했던 Rabin-Karp fingerprint를 사용하면, signature를 계산하는 과정에서 계산했던 해시 값들을 추가 계산 없이 재사용할 수 있다.

즉, MADE에서는 min-hash의 계산과정과 델타압축기법에서의 전처리 과정을 통합하였다. 이를 통하여 페이지에 대한 한 번의 처리만으로 두 과정에 필요한 처리의 상당부분을 수행할 수 있다. 이러한 처리의 통합은 각각의 처리를 독립적으로 수행할 때 발생하는 시간 및 공간적 오버헤드를 효과적으로 줄일 수 있다.

3. 실험결과 및 토의

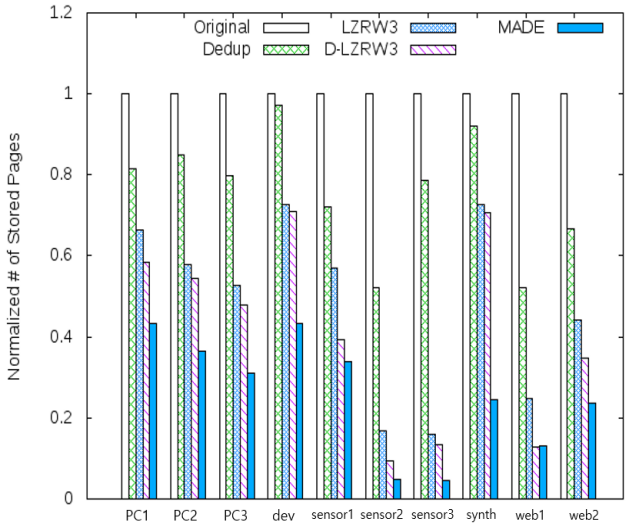

그림 3. 각 기법 별 실행된 쓰기 명령 횟수 비교

실험은 실제로 수집된 I/O trace 10개를 대상으로 이뤄졌다. 일반적인 PC환경에서 추출한 데이터(PC1-3), 반도체 생산 공정의 신호처리 데이터(SENSOR1-2), 하드웨어 합성과정에서 발생하는 데이터 SYNTH, 그리고 웹브라우저 사용시에 발생하는 데이터들을 취합한 기록 등(Web 1-2)을 사용하였다.

그림 3의 실험 결과에서 볼 수 있듯이, MADE 시스템은 대부분의 I/O trace에 대해 중복제거기법과 LZRW3 알고리즘을 이용한 무손실압축기법, 그리고 둘을 순차적으로 조합한 모듈보다 적은 쓰기 데이터 양을 기록하였다. 특히 SYNTH trace와 같은 경우, 중복제거기법 및 무손실압축기법이 함께 적용됐을 때보다 50%이상의 데이터를 추가로 줄임으로써, 각 기법이 효율적으로 처리하지 못하는 데이터를 델타압축의 효율적 적용으로 인해 크게 압축할 수 있는 것을 보여주었다.

4. 결론 및 향후 연구

본 연구에서는 플래시 저장장치에 인가되는 데이터양 감소 기법에 주로 사용되는 중복제거기법, 델타압축기법, 무손실압축기법을 효율적으로 조합하였다. 그 결과 어떠한 조합보다도 더 많은 쓰기 데이터양을 줄일 수 있었으며, 기존 기법과 비슷한 오버헤드를 유지할 수 있었다.

이처럼 시뮬레이션을 통해 MADE 모듈이 효과적으로 쓰기 데이터양을 줄일 수 있음을 확인하였으나 설계된 FTL을 실제 하드웨어로 구현하여 실제 환경 하에서 시간적인 오버헤드가 얼마나 줄어들었는지 확인하는 것은 향후 연구할 과제이다.

감사의 글

이 연구를 위해 실험데이터와 시뮬레이션 툴을 제공해 주신 서울대학교 임베디드시스템 연구실에 감사 드립니다. 이 논문은 2014년도 서울대학교 학부생 연구지원사업의 지원을 받아 수행된 연구임. 이 논문은 2014년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2013R1A2A2A01068260).

참 고 문 헌

[1] Chen, Feng, et al., "CAFTL: A Content-Aware Flash Translation Layer Enhancing the Lifespan of Flash Memory based Solid State Drives," In proceedings of the 9th Conference on File and Storage Technologies (FAST), 2011.

[2] Wu, Guanying, and He, Xubin, "Delta-FTL: improving SSD lifetime via exploiting content locality," In proceedings of the 7th ACM European Conference on Computer Systems (EuroSys), pp. 253-266, 2012.

[3] Lee, Sungjin, et al., "Improving performance and lifetime of solid-state drives using hardware-accelerated compression," Consumer Electronics, IEEE Transactions on, volume 57, issue 4, pp. 1732-1739, 2011.

[4] Bentley, Jon, and Douglas, McIlroy, "Data compression using long common strings," In proceeding of Data Compression Conference. IEEE, pp.287-295, 1999.

[5] Rajaraman, Anand, and Ullman, "Mining of massive datasets," Cambridge University Press, pp.71-126, 2011.

[6] Korn, D, Macdonald, J, and Mogul, J, "RFC 3284: The vcdiff generic differencing and compression data format," Internet Engineering Task Force (IETF), 2002.