Flash Translation Layer의 정량적 성능 예측 모델링

*임경일, 박상훈, 원삼규, 방관후 정의영
연세대학교 전기전자공학과
e-mail : kyungil.im@dtl.yonsei.ac.kr, soskhong@dtl.yonsei.ac.kr, refim@yonsei.ac.kr,
khbang@dtl.yonsei.ac.kr eychung@yonsei.ac.kr

Quantitative Performance Prediction Modeling for FTLs

*Kyungil Im, SangHoon Park, Samkyu Won, Kwanhu Bang,
Eui-Young Chung
School of Electrical and Electronic Engineering
Yonsei University

Abstract

Recently, various flash translation layers(FTL) have been studied and introduced to improve the performance of SSDs by mitigate unique characteristics of NAND flash memories which is the main data storage of SSDs. However, evaluation of FTLs has been totally based on simulation that requires actual implementation of the corresponding FTL algorithm.

Therefore, in this paper, we proposed the modeling method that predict performance of FTLs easily and fast. We may know quantitative performance of FTL simply and accurately by parameterization of FTL and input logical access patterns. By comparison to real simulation results, we prove that our modeling method shows very simple and accurate prediction results with only 12% of error.

I. 서론

최근 낸드 플래시 메모리 기반 솔리드 스테이트 디스크(SSD)들은 기존 하드 디스크 드라이브(HDD) 대비 뛰어난 성능과 안정성으로 인하여 점점 그 수요를 늘려가며 개인용 컴퓨터 뿐 아니라 많은 휴대용 기기들의 저장 매체로 각광을 받고 있다. [x]

하지만 낸드 플래시 메모리는 덮어쓰기가 불가능하고 쓰기 연산 이전에 지우기 연산이 선행되어야 하며, 불균형의 쓰기/읽기 접근 시간, 그리고 매우 큰 단위의 지우기 연산만을 지원하는 독특한 특성을 가졌기 때문에 기존 HDD와 사용하던 파일 시스템과 함께 사용할 시 낸드 플래시 기반 SSD들은 심각한 성능 저하를 겪게 된다.

따라서 이를 해결하고 보다 효율적으로 낸드 플래시를 관리하기 위하여 그림 1과 같이 SSD 내부의 캐시 버퍼와 SSD의 호스트 시스템에서의 논리적 주소는 소프트웨어 layer 인 flash translation layer(FTL)을 통해 물리적 주소로 변환되고, 이 FTL의 효율성에 따라 SSD 전체의 성능은 큰 영향을 받게 된다.

지금까지 수많은 FTL 관련 연구들이 여러 알고리즘을 주장하며 등장하였고 그 성능 또한 발전하고 있다. BAST, FAST 그리고 Superblock[1][2][3]은 이러한 기존 FTL들의 성능 평가 방식은 대부분 소프트웨어와 접근 트레이스를 기반으로 한 시뮬레이션인데, 이는 하드웨어 모델링을 포함한 시뮬레이션 보다는 속도 면에서는 이득이 있지만 알고리즘의 구현이 끝난 상태에서만 평가가 가능하다는 단점이 있다. 즉, 알고리즘의 구상단계에서 FTL의 성능을 정확하게 평가하기 어렵다는 단점이 있는 것이다.

따라서 본 논문에서는 FTL의 성능을 구현과 별개로 간단하면서도 정확하게 평가해볼 수 있는 정량정 성능

모델링을 제시하고 이 모델링을 실제 시뮬레이션 결과와 비교해 보려 한다.

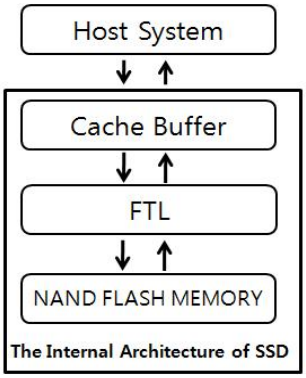

그림 1 Internal Architecture of SSD

II. 본론

그림 1은 일반적인 SSD의 내부 구조를 보여주고 있다. 호스트 시스템으로 부터 들어온 데이터는 캐시 버퍼에 쓰여 지고 낸드 플래시에 저장될 시에는 FTL의 알고리즘에 따라 논리적 주소를 물리적 주소로 변환하여 처리된다. 이러한 주소 변환을 위하여 FTL은 맵핑 테이블을 사용하게 되는데 이 테이블의 기본 단위에 따라 크게 블록 수준 맵핑 FTL과 페이지 수준 맵핑 FTL로 나뉘게 된다. 일반적으로는 보다 작은 단위인 페이지 수준 맵핑 FTL이 많은 메모리를 사용하여 좋은 성능을 내는 것으로 알려져 있다.

또한 FTL은 낸드 플래시의 특성에 기인하여 호스트 시스템으로부터 쓰기 명령이 들어올 경우 업데이트가 유발되거나 새로운 플래시 공간 확보가 필요하게 되고, 이에 따라 필연적으로 지우기나 페이지 복사 등의 연산을 포함하는 가비지 컬렉션을 수행하게 되는데 이 연산들이 성능 저하를 유발하게 된다.

최근에는 이러한 추가 연산 등을 백그라운드로 실행하는 등의 [hydra][4] 노력을 하고 있으나, 기본적으로 FTL의 알고리즘은 맵핑 수준과 그에 따른 가비지 컬렉션 알고리즘의 효율성으로 그 성능을 판단할 수 있다.

마지막으로 FTL의 성능에 영향을 주는 가장 큰 요인 중 하나는 업데이트를 위한 추가 공간의 크기인데, 이를 로그 버퍼라고 부르며 어떤 수준의 맵핑을 사용하는가와 관계없이 가비지 컬렉션의 수행시점을 뒤로 미루고 업데이트에 의한 성능저하를 방지하는 역할을 하게 된다. 특히 블록 수준 맵핑 FTL들의 경우 이러한 로그 버퍼들의 논리적 associativity에 따라 발전해왔고 많은 성능 변화를 보이고 있다.

FTL과 더불어 논리적 접근 패턴 또한 SSD 성능에 큰 영향을 주는 부분 중 하나이다. 특히 접근의 지역성이나 데이터의 길이 등은 일반적으로 SSD 성능에 영향을 주는 것으로 알려져 왔다. 다만 기존의 시뮬레이션 기반 성능 평가에서는 간단한 분석으로만 트레이스의 특성을 나타내었고 실험적으로 그 결과를 보여주는데 그쳐왔다.

따라서 본 논문에서는 FTL과 입력 패턴을 파라미터화 하고 이를 입력으로 가지는 모델링을 통하여 FTL의 성능을 분석하고자 한다.

III. 구현

3.1 모델링 요약

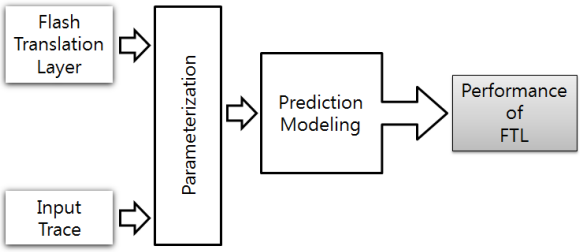

그림 2 제시하는 모델의 성능 예측 과정

본 논문에서는 그림 2와 같은 과정을 통해 FTL의 성능을 예측한다. 먼저 입력 패턴과 FTL의 파라미터화를 통하여 각각의 특성을 대표하도록 한다. 특히 FTL을 대표할 수 있는 파라미터들은 페이지 및 블록 수준 맵핑을 모두 대표할 수 있도록 하고 앞서 언급된 FTL의 성능에 관여하는 요인들을 포함하도록 하였다. 그리고 입력 패턴은 실험할 입력 패턴의 처음부터 마지막까지 모든 정보를 종합하여 특성을 대표하도록 하였다.

그 후 실제 성능평가를 위한 모델링은 앞서 구해진 파라미터들과 기본적인 낸드 플래시 메모리의 타이밍 정보 및 스펙을 입력으로 하여 간단한 수식을 거쳐 단일 낸드 플래시 메모리가 사용되었을 시 주어진 입력 패턴에 대한 throughput을 구한다. 이는 SSD의 하드웨어 효과를 배제함으로서 입력 패턴 및 FTL 간의 성능차이를 분석할 수 있도록 하기 위함이며, 만약 SSD의 하드웨어가 다중 채널이나 인터리빙을 통한 성능향상을 꾀하고 있는 경우 모델링의 조금의 수정을 통하여 하드웨어 효과 또한 적용시켜 볼 수 있다는 장점이 있다.

3.2 파라미터 및 모델링

본 논문에서 제시하고 있는 모델은 FTL을 대표하기 위한 파라미터로는 크게 네 가지를 사용한다. 전체 낸드 플래시 메모리 크기 대비 로그 버퍼 크기의 비율 R_L, 낸드 플래시의 페이지 크기 대비 FTL의 맵핑 수준의 크기 S_M, 각 맵핑 수준과 업데이트를 위한 여유 공간간의 associativity N_A, 마지막으로 가비지 컬렉션 시 지워지는 블록의 평균 개수 N_E이다. 대표적인 블록 수준 맵핑 FTL인 BAST[1] 예를 들면 한 블록당 한 개의 다른 블록을 업데이트를 위한 로그 버퍼로 사용하고 총 10%의 로그 버퍼를 가지며 한 블록당 64개의 페이지를 가지는 낸드 플래시가 사용될 경우 각각의 값은, $R_L = 0.1$, $S_M = 64$, $N_A = 1$이 된다. 그리고 가비지 컬렉션 시 1개 또는 2개의 브록이 지워지므로 $N_E = 1.5$가 된다.

입력 패턴의 파라미터 들에 대한 설명은 아래 표에 종합되어 있다. 모든 파라미터들의 페이지 및 블록은 낸드 플래시 메모리의 페이지와 블록과 같은 크기를 가지는 논리적 페이지와 블록에 대응된다.

D_{total} (byte)	총 쓰여진 페이지 크기
N_{ALP}	블록당 평균 live 페이지
L_{WR}	호스트 요청당 평균 쓰여지는 페이지 갯수
N_{UP}	총 업데이트된 페이지 갯수
N_{UB}	총 업데이트된 블록 갯수

표 1 입력 패턴의 파라미터들

위와 같은 파라미터를 가지고 성능 예측 모델은 앞서 설명된 것과 같이 최종적으로 단일 낸드 플래시 메모리를 사용했을 시의 throughput을 출력해준다.

$$Throughput(MB/s) = \frac{D_{total}}{T_{total}}$$

여기서 T_{total}은 다음과 같이 순수한 낸드 페이지 쓰기 시간과 그에 의한 부가적 연산 수행시간으로 나뉜다.

$$T_{total} = T_{WR} + T_{EX}$$

T_{WR} 의 경우 낸드 플래시의 타이밍 정보 D_{total}의 곱으로 쉽게 계산이 가능하며 T_{EX}의 경우 다음과 같이 나타내어진다.

$$T_{EX} = (D_{total}/S_P) \times P_E \times (T_{CP} \times N_{ALP} + T_E \times (N_E))$$

여기서 T_{CP}와 T_E는 각각 낸드 플래시의 한 페이지 복사 시간과 한 블록이 지워지는데 걸리는 고정된 타이밍 정보이고, S_P는 낸드 플래시 한 페이지의 크기(byte)이다.

고정된 값들을 제외하고 T_{EX}에 큰 영향을 주는 P_E의 경우

$$P_E = \alpha \times \frac{\frac{R_L}{N_L/N_A} \times \frac{N_{UP}}{D_{total}/S_P} \times N_{UB}}{L_{WR}}$$

로 나타내어지고, 여기서 α는 낸드 플래시의 초기에 얼마만큼의 데이터가 쓰여져 있느냐를 나타내주는 변수로 트레이스로부터 추출해낼 수 있으며, N_L의 경우 총 로그 버퍼를 블록 개수로 나타낸 것이다.

결론적으로 P_E는 한 페이지 쓰기 요청시마다 각 파라미터들을 통하여 부가적 연산이 수행될 확률을 구하고, 최종적인 모델에서는 이를 총 쓰여진 페이지 개수와 함께 각 부가적 연산이 필요로 하는 타이밍 정보를 통하여 T_{EX}를 구할 수 있게 된다.

Ⅳ. 실험 결과

본 논문에서는 BAST와 다섯가지의 trace를 이용하여 소프트웨어 기반 시뮬레이션 결과와 모델링으로부터의 결과를 비교해 보았다. 먼저 다섯가지의 입력 패턴으로부터 추출한 파라미터는 다음과 같다. 지면의 부족으로 D_{total}의 단위를 MB로 표시하였다.

	1	2	3	4	5
D_{total}	230	109	1546	665	6268
N_{ALP}	16.39	2.46	63.21	49.04	61.81
L_{WR}	2.36	1.12	15.84	6.81	64.19
N_{UB}	2275	15332	10381	2891	16139
N_{UP}	4.6e4	1.5e4	1.3e5	9.7e4	2.2e6

사용된 FTL이 BAST이므로 FTL의 파라미터는 3장에서 설명된 것과 같은 파라미터를 사용했으며 가정한 낸드 플래쉬의 타이밍 및 사이즈 정보는 다음과 같다.

S_P	2KB	N_P	64
T_W	$250\mu s$	T_{CP}	$325\mu s$
T_E	$2ms$	N_B	32786

여기서 N_P, N_B는 각각 블록당 페이지 개수 및 전체 블록의 개수를 의미한다.

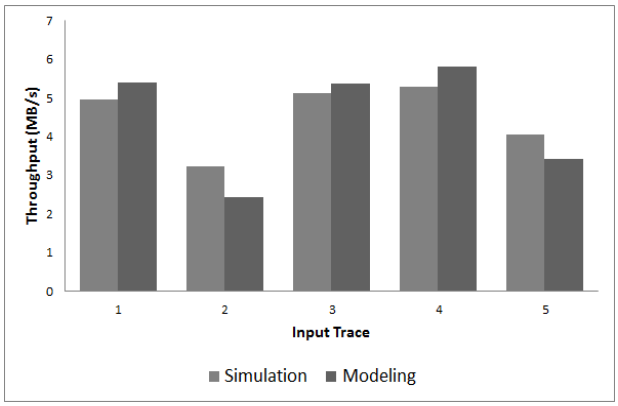

그림 3 시뮬레이션과 모델링의 결과 비교

그림 3은 최종적으로 시뮬레이션과 모델링의 값을 비교하고 있다. 시뮬레이션 대비 모델링 값의 평균 에러는 약 12%이지만, 전체적인 경향은 잘 따라가는 모습을 보여주고 있다. 이는 실제 BAST 알고리즘으로 시뮬레이션 시 50000개의 요청을 가지는 각각의 trace를 실험하는데 수 분이 소요되는 반면, 본 논문에서 제시하는 방법으로 입력 패턴의 파라미터화 및 결과값 출력에는 FTL 알고리즘의 구현 없이도 수 초 이내에 완료되므로 제시하는 모델의 목적에 부합한다고 할 수 있다.

V. 결론 및 향후 연구 방향

본 논문에서 제안한 FTL의 정량적 성능 예측 모델링은 FTL의 구현 없이도 시뮬레이션 대비 평균 12%의 에러로 대략적인 성능을 손쉽게 파악할 수 있다. 따라서 새로운 FTL 알고리즘 등의 평가에 손쉽게 사용할 수 있을 것으로 기대된다.

향후 보다 정확도를 향상하기 위하여 FTL의 파라미터를 다양화 하여 FTL을 대표할 수 있도록 모델링을 수정하여 보다 최적화된 FTL의 개발을 가능토록 할 것이다.

Acknowledgement : 이 논문은 2008년 정부(교육과학기술부)의 재원으로 한국학술진흥재단 의 지원 및 IDEC 의 지원을 받아 수행된 연구임.

참고문헌

[1] J. Kim, J. M. Kim, S. H. Noh, S. L. Min, and Y. Cho. "A space-efficient flash translation layer for compact flash systems," IEEE Transaction on Consumer Electronics, Vol. 48, No.2, pp.3666-375, 2002

[2] S.-W. Lee, D.-J. Park, T.-S. Chung, D-H. Lee, S. Park, and H.-J. Song. A log buffer-based flash transactions on, 48(2):336-375 May 2002.

[3] J.-U. Kang, H. Jo, J.-S. Kim, and J. Lee. A superblock-based flash translation layer for nand flash memory. In Proceedings of the 6th ACM & IEEE International conference on Embedded software (EMSOFT '06). ACM, 2006.

[4] Yoon Jae Seong, Eyee Hyun Nam, Jin Hyuk Yoon, Hongseok Kim, Jin-yong Choi, Sookwan Lee, Young Hyun Bae, Jaejin Lee, Yookun Cho, Sang Lyul Min, "Hydra: A Block-Mapped Parallel Flash Memory Solid-State Disk Architecture," IEEE Transactions on Computers, 02 Mar.

[5] NAND Flash Datasheet : http://www.hynix.com/