플래시 변환 계층에 대한 TPC-C 벤치마크를 통한 성능분석*

박성환° 장주연 서영주 박원주 박상원
한국외국어대학교 컴퓨터및정보통신공학과
shpark@dislab.hufs.ac.kr, jyjang@dislab.hufs.ac.kr, yjsuh@dislab.hufs.ac.kr,
wjpark@dislab.hufs.ac.kr, swpark@hufs.ac.kr

Performance Analysis of Flash Translation Layer by TPC-C Benchmark

Sung-Hwan Park° Ju-Yeon Jang Young Ju Suh Won Joo Park, and Sangwon Park
Computer Science & Information Communication Engineering Div., Hankuk University of Foreign Studies

1. 서론

최근 디지털 카메라, 휴대폰, MP3 플레이어, PDA 등 이동성을 중요시하고 플래시 메모리를 탑재한 여러 제품들이 등장하였다. 플래시 메모리에서는 데이터베이스 시스템을 통하여 보다 정보를 효율적으로 관리해야 할 필요가 있다. 그러나 플래시 메모리는 아직 데이터베이스를 충분히 지원하지 못하고 있다. 그 이유는 플래시 메모리는 각 블록에 대한 소거 횟수(보통 10만 번)가 제한되어 있고 특정 블록에 쓰기 연산을 하기 위해서는 반드시 그 블록이 먼저 소거되어 있어야 하기 때문이다. 또한, 하드디스크와는 다르게 읽기, 쓰기와 소거 연산 수행에 있어 약 80:200:1500 정도의 속도 차이를 보인다. 이러한 플래시 메모리의 단점들을 보완하기 위해 FTL(Flash Translation Layer) 소프트웨어가 플래시 메모리 시스템에 탑재되어 있다[1]. 그러나 여전히 데이터베이스 시스템을 지원하기에는 매우 부족하다. 대표적인 FTL 알고리즘으로 M-System사의 FMAX[2], Mitsubishi사의 여유영역기법[3]과 로그 블록 기법(log block scheme)[4]이 있다. 그러나 이러한 FTL 알고리즘은 많은 정보를 다루는 데이터베이스 시스템에서는 기대 이하의 성능을 보인다. 본 논문은 이미 제안된 세 가지 FTL 알고리즘들이 데이터베이스 시스템에서 어떠한 성능을 보이는지를 알기위해 실험을 하였고, TPC-C 벤치마크[5]를 수행하였을 때 만들어지는 로그를 사용하였다. 본 논문은 새로운 FTL 알고리즘이 필요한 이유와 그 방향성을 제시한다. 최근의 플래시 메모리는 그 성능을 개선하기 위해 내부복사(copy-back)와 칩 인터리빙(chip-interleaving) 기술을 사용한다. FTL 알고리즘의 연산 횟수를 통해 총 수행 시간, 데이터베이스 페이지 쓰기 연산에 대한 성능 비와 평균적인 응답 시간을 계산하였다. 이를 통해 각 알고리즘의 성능을 평가할 수 있다.

2. 성능 분석

첫 번째로, 플래시 메모리에서 데이터베이스 I/O를 처리하는데 걸린 총 수행 시간을 계산하였다. 이는 해당 알고리즘이 주어진 I/O를 얼마나 빠르게 처리할 수 있는지의 기준이 될 수 있다.

$$T_{total} = n_r C_r + n_w C_w + n_e C_e \qquad (1)$$

식 1은 총 수행 시간을 계산하는 식이다. 이는 각 FTL 알고리즘이 데이터베이스의 I/O를 처리하는데 발생한 읽기, 쓰기와 소거 연산의 횟수에 각각의 비용을 곱해서 모두 더한 값이다. n_r은 플래시 페이지 읽기 횟수, n_w은 플래시 페이지 쓰기 횟수, n_e은 플래시 블록 소거 횟수, C_r은 플래시 페이지 읽기 비용, C_w은 플래시 페이지 쓰기 비용, C_e은 플래시 블록 소거 비용을 나타낸다.

표 1. 플래시 메모리에서의 총 수행 시간(초)

	CIL 1	CIL 2	CIL 4
FMAX	1480.33	822.66	483.30
Mitsubishi	1899.10	1439.54	1560.92
LOG	13856.44	14366.57	15928.86

표 2. NAND 플래시 메모리의 칩 인터리빙 시 접근 속도

	읽기	쓰기	소거
CIL(1)	320μs (8KB)	800μs (8KB)	1.5ms (128KB)
CIL(2)	160μs (8KB)	400μs (8KB)	1.5ms (256KB)
CIL(4)	80μs (8KB)	200μs (8KB)	1.5ms (512KB)

표 1은 플래시 메모리의 크기가 2G바이트 일 때 각 FTL 알고리즘에 대해서 칩 인터리빙에 따른 실험 결과이다. 표 1은 표 2의 칩 인터리빙 시 접근 속도를 기준으로 계산된 값이다. 전체적인 결과를 보면 로그 블록 기법의 총 수행 시간이 가장 느린 것을 알 수 있다. 데이터베이스 I/O는 완전히 임의적인 디스크 I/O를 발생시킨다. 때문에 로그 블록 기법은 로그 블록의 페이지를 많이 활용하지 못하고 계속해서 합병 연산이 발생하게 되어, 수행 시간이 가장 느리다. FMAX의 복사블록과 Mitsubishi의 여유영역에 변동 섹터 방식으로 데이터를 기록하기 때문에 복사블록이나 여유영역의 활용도가 증대하여 로그 블록 기법보다는 더 유연하게 I/O를 처리하며, 더 적은 합병연산이 유발된다. 하지만 FMAX와 Mitsubishi는 FMAX의 복사블록과 Mitsubishi의 여유영역을 위해 두 배의 메모리가 필요하기 때문에 성능이 좋다고 볼 수 없다. 플래시 메모리의 크기가 2G바이트보다 큰 경우에도 총 수행 시간은 2G바이트인 경우의 총 수행 시간과 비슷했다. 이로서 플래시 메모리의 크기가 커져도 성능에는 영향을 미치지 않는 것을 알 수 있다.

두 번째로, 표 1에서 구한 결과를 한 개의 데이터베이스 페이지 쓰기 연산에 대한 성능 비로 나타내었다. 하드 디스크에서는 데이터베이스의 쓰기 연산 하나당 실제로 디스크에 쓰기 연산이 하나가 발생한다. 하드 디스크는 데이터를 고정 섹터 방식으로 기록하기 때문에 실제로는 쓰기 연산 한번만 발생한다. 하지만, 플래시 메모리에서는 데이터베이스의 쓰기 연산 하나당 읽기, 쓰기와

* 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업[2006-S-040-01, Flash Memory 기반 임베디드 멀티미디어 소프트웨어 기술 개발]과 과학기술부 및 대구경북과학기술연구원의 연구개발사업의 일환으로 수행하였음.

소거 연산이 발생될 수 있다. 그러므로 데이터베이스에서 쓰기 연산이 하나 발생했을 때 플래시 메모리에서 얼마나 부가적인 연산이 발생되는지가 중요하다. 때문에 본 논문에서는 데이터 페이지 하나를 저장하는데 필요한 비용에 대한 실제 비용의 비(a)를 계산하여 각 FTL 알고리즘의 성능을 평가하였다.

$$k = \frac{S_p}{0.5m} \quad (2)$$

데이터베이스의 한 페이지를 기록할 때 실제 플래시 메모리에 요청되는 페이지 개수 k는 식 2와 같다. S_p는 데이터베이스의 한 페이지 크기 (K바이트), 0.5는 플래시 메모리의 한 섹터의 크기 (K바이트), m은 플래시 메모리가 한 번에 읽기와 쓰기 연산을 하는 섹터의 수를 나타낸다. 식 3은 데이터베이스의 한 페이지를 기록할 때 플래시 메모리에서 소요되는 시간(p_w)을 나타낸다.

$$p_w = kC_w \quad (3)$$

$$P_w = \sum_{i=1}^{n} p_{w_i} = k \cdot n_d \cdot C_w \quad (4)$$

$$P'_w = n_r C_r + n_w C_w + n_e C_e \quad (5)$$

$$\alpha = \frac{P'_w}{P_w} = \frac{n_r C_r + n_w C_w + n_e C_e}{n_d \cdot p_w}$$

$$= \left(\frac{n_r}{n_d} C_r + \frac{n_w}{n_d} C_w + \frac{n_e}{n_d} C_e\right) \Big/ p_w \quad (6)$$

식 6에서와 같이 최적 비용에 대한 실제 비용의 비(a)를 구하여, 플래시 메모리에 대한 성능을 평가할 수 있다. 즉, a는 데이터베이스의 페이지 쓰기 연산을 처리하는데 있어서 실제로 발생한 비용을 예상 비용으로 나누어 그 비율을 나타낸 것이다. 이때 a는 p_w의 영향을 받는다. p_w값은 칩 인터리빙에 따라 달라지는 값으로, 하드웨어 성능을 고려한 값이 된다. 그러므로 a는 하드웨어 성능을 반영한 값이다. a는 데이터베이스의 한 페이지를 기록하는데 플래시 메모리에서 드는 비용을 비율로 나타낸 값이다.

표 3. 플래시 메모리에서의 a

	CIL 1	CIL 2	CIL 4
FMAX	2.09	2.32	2.72
Mitsubishi	2.68	4.06	8.80
LOG	19.52	40.48	89.76

표 4. 플래시 메모리에서의 평균 응답 시간

	CIL 1	CIL 2	CIL 4
FMAX	1.67	0.93	0.55
Mitsubishi	2.14	1.62	1.76
LOG	15.62	16.19	17.95

표 3은 칩 인터리빙에 따른 각 FTL의 a를 나타낸 것이다. 여기서 칩 인터리빙이 커질수록 a값이 증가한다. 이것은 한 번에 읽거나 쓸 수 있는 섹터의 개수가 증가하여 성능을 증진시킬 수 있으나, 로그 블록, 복사블록과 여유영역 등의 활용도가 낮고, 논리적 블록이 커질수록 페이지를 찾는데 걸리는 시간이 길어지기 때문이다.

세 번째로, 평균적인 응답 시간을 측정하였다. 플래시 메모리는 데이터베이스에서 I/O를 요청하게 되면 빠른 시간 내에 결과를 반환해야 한다. 또한, 데이터베이스에서 일정량의 I/O를 요청하면 일정한 시간 내에 결과를 반환해야 한다. 데이터베이스의 한 I/O를 처리하는데 걸리는 평균적인 시간이 작을수록 효율적으로 처리한다고 할 수 있다. 표 4에서 보면 FMAX와 Mitsubishi의 평균 응답 시간은 칩 인터리빙이 커질수록 평균 응답 시간이 줄어드는 반면, 로그 블록 기법의 평균 응답 시간은 커짐을 알 수 있다. 그 이유는 칩 인터리빙이 커지면 논리적 블록의 크기가 커져 원하는 섹터를 찾기위해 탐색해야 하는 섹터의 개수가 증가하기 때문이다.

3. 결론

현재 플래시 메모리에 탑재되어 있는 FTL 알고리즘들은 순차적이거나 몇 개의 블록을 동시에 참조할 때는 좋은 성능을 발휘한다. 이러한 패턴을 보이는 윈도우즈 파일 시스템에서 몇 FTL 알고리즘은 좋은 성능을 보였다[6]. 하지만, 데이터베이스와 같은 임의의 위치에 쓰기 연산을 수행하는 경우에는 나쁜 성능을 보인다. 대표적인 FTL 알고리즘 중에 성능이 매우 좋은 로그 블록 기법은 데이터베이스 시스템에서 가장 안 좋은 성능을 보였다. 또한, 대표적인 FTL 알고리즘들은 칩 인터리빙과 같은 하드웨어의 기능을 잘 이용하지 못하는 경향을 보였다. 이는 데이터베이스 시스템에 맞는 새로운 저장 기법이 필요함을 의미한다. 새로운 저장 기법은 하드웨어 특성을 잘 활용하고, 플래시 메모리의 단점을 효과적으로 극복할 수 있는 기법이어야만 한다. 데이터베이스 시스템에 맞는 새로운 저장 기법이 개발되어야만 휴대용 저장 장치로서 플래시 메모리가 진정한 가치를 발휘할 것이다.

참고 문헌

[1]Eran Gal and Sivan Toledo. Algorithms and Data Structures for Flash Memories. ACM Computing Surveys. Volume 37, Issue 2. June 2005. Pages: 138-163. : 2005.

[2]Amir Ban. Flash file system optimized for page-mode flash technologies, 1999. United States Patent, no. 5,937,425.

[3]Takayuki Shinohara. Flash memory card with block memory address arrangement, 1999. United States Patent, no. 5,905,993.

[4]Jesung Kim, Jong Min Kim, Sam H. Noh, Sang Lyul Min, and Yookun Cho. A space-efficient flash translation layer for compact flash systems. IEEE Transactions on Consumer Electronics, 48(2), 2002

[5]Sang-Won Lee, Bongki Moon. Design of Flash-Based DBMS: An In-Page Logging Approach. ACM SIGMOD International Conference on Management of Data, Beijing, China, June, 2007.

[6]박원주, 유현석, 박성환, 김도윤, 박상원. 윈도우즈 파일 시스템에서 플래시 메모리의 FTL 알고리즘 성능 분석. 한국정보과학회 학술대회, 2005. 7.