부분 가비지 컬렉션을 이용한 로그 구조 파일시스템의 쓰기 성능 개선

곽현호°, 신동군
성균관대학교 정보통신공학부
gusghrhkr@gmail.com, dongkun@skku.edu

Partial Garbage Collection Technique for Improving Write Performance of Log-Structured File Systems

Hyunho Gwak°, Dongkun Shin
School of Information and Communication Engineering, Sungkyunkwan University

요 약

최근 플래시 저장장치의 사용이 대중화되면서 플래시 저장장치의 특성에 맞는 로그 구조 파일시스템에 대한 관심도 높아지고 있다. 로그 구조 파일시스템은 임의 쓰기 성능이 순차 쓰기 성능보다 낮은 저장장치에서 특히 유리하다. 최근 로그 구조 파일 시스템은 가비지 컬렉션 비용을 줄이기 위해 SSR (Slack Space Recycling)을 사용하는데, 이 기법은 임의 쓰기를 유발하므로 임의 쓰기 성능이 낮은 저장장치에서 쓰기 성능을 감소시키는 문제가 있다. 이 논문에서 제시하는 부분 가비지 컬렉션은 SSR을 사용할 때 유효한 블록들을 일부만 복사하여 옮김으로써 임의 쓰기 수를 줄여서 쓰기 성능을 증가시키는 기법으로, SD 카드에서 쓰기 성능을 최대 40% 증가시켰다.

1. 서 론

플래시 저장장치는 저전력, 작은 크기, 높은 내구성, 높은 임의 접근 성능 등의 특성을 가지고 있어서 휴대전화 같은 임베디드 시스템에서 많이 사용되어 왔다. 최근에는 SSD나 SD 카드 등의 가격이 낮아지면서 플래시 저장장치가 점점 대중화되는 중이다. 플래시 저장장치에 적합한 파일시스템인 로그 구조 파일시스템[1]에 대한 관심 또한 높아지고 있다.

로그 구조 파일시스템은 데이터를 업데이트 할 때 원래 위치에 엎어쓰는 것이 아니라, 전에 기록한 데이터는 무효화하고 새로 쓰여지는 데이터는 다른 곳에 기록하며 순차적으로 쓴다. 그런데, 무효화된 데이터들이 남아있으므로 저장장치의 용량을 다 사용하기 전에 순차 쓰기를 할 수 있는 연속적인 공간이 없어지게 된다. 그래서, 순차 쓰기를 할 연속된 여유 공간을 확보하기 위해, 무효화된 데이터들을 모아서 연속된 여유 공간을 만드는 가비지 컬렉션 (Garbage Collection)을 파일시스템에서 해주어야 한다. 그런데, 저장장치가 바쁜 상태에서 가비지 컬렉션을 하면 유효한 데이터들을 복사하는 오버헤드로 인해 쓰기가 지연되어 쓰기 성능이 감소한다. 이 문제를 해결하기 위해 SSR (Slack Space Recycling)[2]이 제시되었다. SSR은 연속된 여유 공간이 부족할 때 가비지 컬렉션을 하지 않고 데이터를 무효화된 데이터 자리에 기록함으로써 가비지 컬렉션을 지연시키는 기법이다. 하지만, SSR을 사용하면 남아있는 유효한 데이터들로 인해 쓰기 요청이 나누어져서 임의 쓰기를 유발하게 된다. 이것은 특히 임의 쓰기 성능이 순차 쓰기 성능에 비해 낮은 저장장치에서 큰 성능저하를 유발할 수 있기 때문에, 복사할 유효한 데이터들이 적어 가비지 컬렉션의 오버헤드가 작은 경우에는, 가비지 컬렉션을 하는 것이 오히려 쓰기 성능에 더 좋을 수 있다.

이 논문에서 제시하는 부분 가비지 컬렉션은 이 점에 착안하여 SSR을 사용할 때 저장장치의 순차, 임의 쓰기 성능과 가비지 컬렉션의 오버헤드를 비교하여, 가비지 컬렉션 오버헤드보다 순차 쓰기로 인한 성능 증가가 더 크다면 해당 쓰기 공간에 부분 가비지 컬렉션을 적용하여 쓰기 성능을 증가시키는 기법이다.

2. 관련 연구

2.1 플래시 저장장치

플래시 저장장치는 임의 쓰기를 사용하면 플래시 칩 사이의 병렬처리가 되지 않으므로 임의 쓰기 성능이 순차 쓰기 성능보다 낮다. 그래서, 사용자가 임의 쓰기를 해도 파일시스템에서 순차 쓰기를 하는 로그 구조 파일시스템은 플래시 저장장치에서 효과적으로 사용될 수 있다. 이 논문은 로그 구조 파일시스템의 쓰기 성능을 개선시키기 위한 것이므로 로그 구조

이 논문은 2013년도 정부(교육부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임 (2013R1A1A2A10013598)

표 1. 순차 쓰기와 임의 쓰기 성능

	SD card A	SD card B	SD card C
2MB Sequential Write (MB/s)	8.968	10.961	16.829
4KB Random Write (MB/s)	0.882	0.989	1.140

파일시스템이 적합한 플래시 저장장치를 사용하여 실험하였다. 사용한 플래시 저장장치는 클래스 2 성능을 가지는 SD 카드 A 제품과 클래스 10 성능을 가지는 SD 카드 B와 C 제품이다. 표 1은 사용한 저장장치의 순차 쓰기 성능과 임의 쓰기 성능을 측정한 표이다. 표에서 확인할 수 있는 것처럼 순차 쓰기에 비해 임의 쓰기 성능이 매우 낮으므로 로그 구조 파일시스템을 사용하는 것이 유리하다.

2.2 F2FS (Flash Friendly File System)

이 논문에서는 로그 구조 파일시스템 중 하나인 F2FS[3]를 사용한다. F2FS는 플래시 메모리의 특성에 맞추어 설계된 파일시스템으로 플래시 저장장치에서 높은 성능을 보인다. F2FS의 기본 쓰기 단위는 4KB 블록이고, 연속된 512개의 블록의 집합인 세그먼트를 할당 받아서 순차 쓰기를 한다. 데이터는 업데이트 빈도가 많은 순서대로 hot, warm, cold 세그먼트에 분리되어 기록된다. 가비지 컬렉션 시에 복사되는 유효한 블록들은 거의 업데이트 되지 않는다고 판단되어서 cold 세그먼트에 쓰인다.

할당 받은 세그먼트의 블록들이 다 소진 되면 유효한 블록이 없는 프리 세그먼트를 새로 할당 받는데, 이때 프리 세그먼트의 수가 미리 정의된 값보다 작으면 가비지 컬렉션이 시작된다. 가비지 컬렉션은 유효한 블록들이 남아 있는 더티 세그먼트에서 유효한 블록들을 모두 cold 세그먼트로 이동시켜 프리 세그먼트를 만든다. 그런데, 가비지 컬렉션을 사용할 때 유효한 블록들을 복사하는 오버헤드로 인해 쓰기 성능이 감소하므로, F2FS는 가비지 컬렉션을 적용하기 전에 SSR을 우선 사용한다. SSR은 프리 세그먼트 대신 더티 세그먼트를 할당 받아 무효한 블록 자리에 새로운 데이터를 엎어쓰는 것으로, 가비지 컬렉션을 지연시킬 수 있다. 그림 1(a)와 (b)는 A, B, C, D, E 다섯 개 블록을 프리 세그먼트에 쓸 때와 SSR을 사용하여 더티 세그먼트에 쓸 때 세그먼트의 상태를 나타낸 것이다. 프리 세그먼트에 쓸 때 순차적으로 처리되는 하나의 쓰기 요청이 SSR 세그먼트에서는 남아있는 유효 블록에 의해 여러 개로 나누어지므로 임의 쓰기가 유발된다.

3. 부분 가비지 컬렉션

3.1 SSR과 부분 가비지 컬렉션

SSR은 가비지 컬렉션에 의한 블록복사 오버헤드를 크게 감소시키지만, 임의쓰기를 유발하는 단점이 있다. 본 논문에서는 가비지 컬렉션의 블록복사 오버헤드를 감소시키면서도 작은 크기의 임의쓰기를 막기 위해서

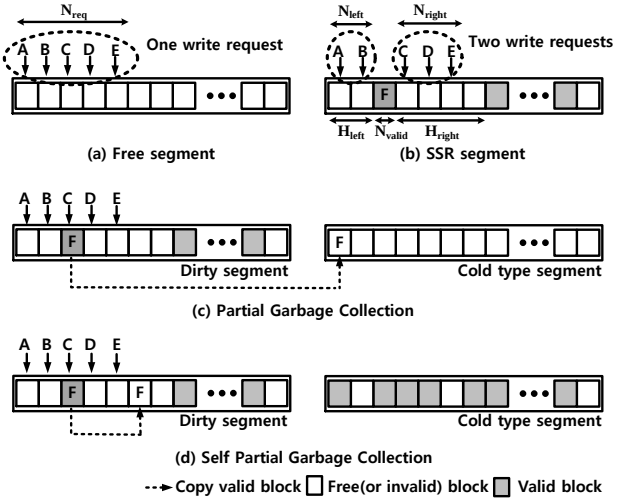

그림 1. SSR과 부분 가비지 컬렉션

*부분 가비지 컬렉션*이라는 기법을 제시한다. 부분 가비지 컬렉션은 세그먼트 전체에 가비지 컬렉션을 적용하는 것이 아니라, 쓰기 요청을 처리할 수 있는 일부 블록들에 대해서만 부분적으로 가비지 컬렉션을 적용하는 것으로, 쓰기 성능이 SSR보다 더 좋아질 수 있다. 예를 들어, 그림 1(c)처럼 SSR 세그먼트의 한 개의 유효한 블록 F만 cold 세그먼트로 복사하면 주어진 다섯 개의 블록을 순차적으로 기록하여 쓰기 성능을 증가시킬 수 있다. 그런데, 그림 1(d)와 같이 cold 세그먼트에 남아있는 유효한 블록들이 많아서 유효한 블록들을 cold 세그먼트에 복사할 때에도 작은 크기의 임의 쓰기가 발생할 때에는 유효한 블록들을 cold 세그먼트가 아니라 자기 세그먼트에 복사하는 *셀프 가비지 컬렉션*을 통해 하나의 쓰기 요청으로 합치는 것이 쓰기 성능이 더 좋을 수 있다.

3.2 선택적 부분 가비지 컬렉션

부분 가비지 컬렉션은 기존 SSR보다 쓰기 성능을 더 증가시킬 수 있는 경우에만 선택적으로 사용되어야 한다. 쓰기 요청 사이의 유효한 블록들에 대해 각 기법을 사용했을 때 쓰기 성능을 계산하기 위해서 그림 1처럼 SSR 세그먼트에서 쓰기 요청의 길이를 N_{req}, 유효한 블록들의 개수를 N_{valid}, 양 옆 연속된 무효한 블록 개수를 H_{left}와 H_{right}로 정의한다. 그러면, N_{req}크기의 쓰기요청은 두 개의 빈 공간에 N_{left}, N_{right}의 크기를 가지는 두 개의 쓰기 요청을 만들어 내는데, $N_{left} = H_{left}$이며, $N_{right} = \min(H_{right}, N_{req} - H_{left})$이다.

저장장치에서 각 임의 쓰기 크기마다 쓰기 성능(MB/s)을 측정하여 임의 쓰기 크기 별 한 블록을 쓰는데 소요되는 시간(s/Block)인 함수 T_{Block}을 모델링 할 수 있다. 그러면, n개의 블록들을 순차적으로 쓸 때 소요되는 시간은 $n \cdot T_{Block}(n)$가 된다. 플래시 메모리의 특성상 읽기 성능이 쓰기 성능보다 매우 높으므로 쓰기에 소요되는 시간만 고려하여, cold 세그먼트의 연속된 무효한 블록들의 평균 개수를 H_{cold}라고 할 때

m개의 유효한 블록을 cold 세그먼트에 복사하는 오버헤드의 평균값을 $m \cdot T_{Block}(H_{cold})$로 모델링 할 수 있다. 이 값들을 이용하여 각각 SSR, 부분 가비지 컬렉션, 셀프 가비지 컬렉션을 사용하여 쓰기 요청을 처리하는 시간인 T_{SSR}, T_{PGC}, T_{SPGC}를 다음과 같이 모델링할 수 있다.

$T_{SSR}(H_{left}, H_{right}, N_{req}) = H_{left} \cdot T_{Block}(H_{left}) + \min(H_{right}, N_{req} - H_{left}) \cdot T_{Block}(\min(H_{right}, N_{req} - H_{left}))$

$T_{PGC}(N_{req}, N_{valid}, H_{cold}) = N_{req} \cdot T_{Block}(N_{req}) + N_{valid} \cdot T_{Block}(H_{cold})$

$T_{SPGC}(N_{req}, N_{valid}) = (N_{req} + N_{valid}) \cdot T_{Block}(N_{req} + N_{valid})$

예를 들어, 그림 1(b)의 SSR 세그먼트에서 세 가지 기법을 사용하여 쓰기에 소요되는 시간은 각각 $2T_{Block}(2) + 3T_{Block}(3)$, $5T_{Block}(5) + T_{Block}(H_{cold})$, $6T_{Block}(6)$으로 계산할 수 있다. 이처럼 무효한 공간 사이에 위치한 연속된 유효한 블록구간마다 세 가지 기법의 시간을 계산하여, 가장 적은 시간이 소요되는 기법을 선택한다.

4. 실험과 분석

4.1 실험 환경

실험은 안드로이드 4.2.2, 리눅스 커널 3.4.5 버전인 갤럭시 S4에 표 1의 세 가지 SD 카드 제품을 장착하여 같은 크기의 파티션에서 진행하였다. 부분 가비지 컬렉션은 SSR을 사용할 때 적용되는 것이므로, Tiobench를 사용하여 F2FS에서 SSR을 사용하도록 같은 크기의 warm 파일과 cold 파일로 저장장치의 공간을 전부 채우고 실험하였다. 이때 각 파일들을 무작위로 업데이트하여 사용량을 50%로 맞추었다. 그리고, 다시 Tiobench를 사용하여 같은 크기의 warm 파일을 순차 쓰기 했을 때, 기존 SSR만을 사용한 경우와 선택적 부분 가비지 컬렉션을 적용한 경우의 쓰기 성능을 비교하였다. 표 2는 이때 H_{left}, N_{valid}, N_{right}, H_{cold}의 평균값을 측정한 것이다. 세그먼트 안의 유효한 블록들이 고르게 퍼져있으므로 H_{left}, N_{valid}, N_{right}의 평균값이 비슷한 것을 확인할 수 있다.

4.2 실험 결과

그림 2(a) 그래프는 SSR, 부분 가비지 컬렉션, 셀프 부분 가비지 컬렉션을 사용한 비율을 나타낸다. SD 카드 A와 C에서는 SSR이 거의 사용되지 않았지만, SD 카드 B는 임의 쓰기 크기 별 성능 차이가 크지 않으므로 SSR이 주로 사용되었다. 그림 2(b) 그래프는 선택적 부분 가비지 컬렉션을 사용했을 때 쓰기량과 쓰기 요청 수를 나타낸다. 부분 가비지 컬렉션을 사용하면 유효한 블록을 cold 세그먼트로 복사하기 때문에 쓰기량이 각 저장장치에서 26%, 27%, 27% 증가하였다. 하지만, 쓰기 요청들이 합쳐져서 쓰기 요청

표 2. 평균 블록 개수

	H_{left}	N_{valid}	H_{right}	N_{right}	N_{req}	H_{cold}
Average	3.0	3.0	2.5	2.5	1079.0	5.1

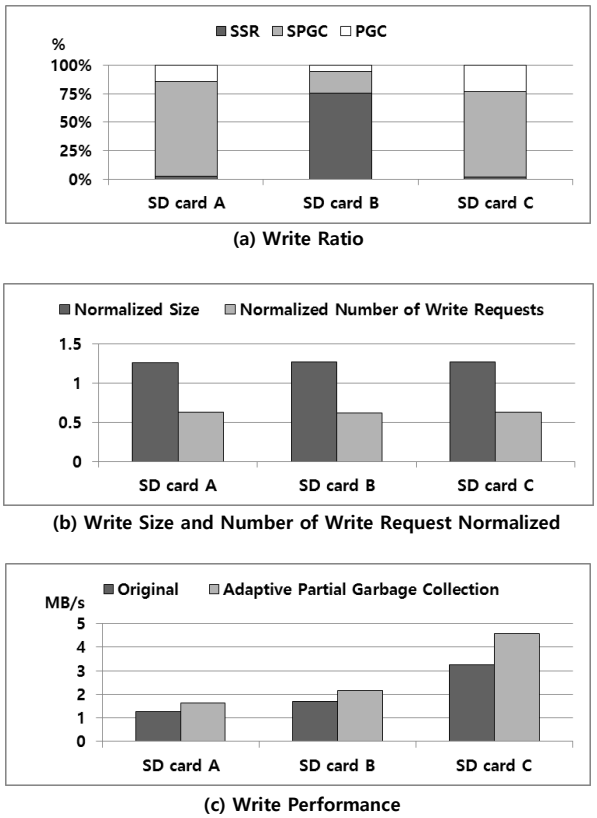

그림 2. 선택적 부분 가비지 컬렉션 실험 결과

수가 63%, 62%, 62% 감소하였다. 그림 2(c) 그래프는 쓰기 성능을 비교한 것으로, 부분 가비지 컬렉션의 오버헤드로 감소하는 쓰기 성능보다 쓰기 요청이 합쳐져서 증가하는 쓰기 성능이 더 크므로 각 저장장치에서 쓰기 성능이 29%, 27%, 40% 증가하였다.

5. 결론

이 논문에서는 F2FS에서 SSR모드를 사용할 때 일부 유효 블록들을 가비지 컬렉션 함으로써 임의 쓰기 수를 줄여 쓰기 성능을 높이는 부분 가비지 컬렉션 기법을 제시했다. 각 저장장치 마다 임의 쓰기 성능을 측정하여 가비지 컬렉션 오버헤드와 줄어드는 임의 쓰기로 인한 이득을 계산하여 가장 효율적인 기법을 사용한다. 실험 결과 SD 카드에서 쓰기 성능이 최대 40% 증가하였다.

Reference

[1] M. Rosenblum and J. Ousterhout, “The design and implementaion of a log-structured file system,” ACM Transactions on Computer Systems, vol. 10, no. 1, pp. 26-52, 1992.

[2] Yongseok. Oh, et al. “Optimizations of LFS with slack space recycling and lazy indirect block update,” Proceedings of the 3rd Annual Haifa Experimental Systems Conference, p 2, 2010.

[3] An f2fs teardown, http://lwn.net/Articles/518988