로그 구조 파일시스템의 쓰기 성능 개선을 위한 부분 가비지 컬렉션

*권용선, 강윤지, 신동군
성균관대학교 정보통신대학
e-mail : *tea984811@skku.edu, oso41@skku.edu, dongkun@skku.edu*

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

*Young-Sun Kwon, Yun-ji Kang, Dong-Kun Sin
School of Information and Communication Engineering
Sungkyunkwan University

Abstract

Log-structured file systems are suitable for flash memory-based storage systems. Recent log-structured file systems use the slack space recycling technique under high utilizations to reduce the garbage collection overhead. However, SSR degrades the write performance by generating a large number of small write requests. In this paper, we propose the partial garbage collection technique to solve the performance problem of SSR.

I. 서론

스마트폰에 사용되는 eMMC와 PC에 사용되는 SSD 등 플래시 메모리 기반의 저장장치가 대중화되고 있다. 플래시 메모리는 빠른 속도와 저 전력으로 동작하는 장점을 지니고 있지만, 덮어 쓰기가 불가능하고 쓰기 단위와 지우기 단위가 다르며 지우기 연산 횟수가 제한되어 있다는 단점을 지니고 있다.

이러한 플래시 메모리에는 지정한 위치에 업데이트(in-place update)를 하는 전통적인 파일시스템이 부적합하다. 전통적인 파일시스템에서 임의 쓰기와 덮어쓰기가 빈번하게 일어나기 때문에 플래시 메모리의 쓰기 성능과 수명에 효율적이지 못하다.

그에 반해 로그 구조 파일시스템[1]은 모든 데이터를 순차 쓰기로 기록하는 구조로 플래시 메모리의 성능을 향상시키고 수명을 증가시켜 플래시 메모리에 적합하다.

하지만 로그 구조 파일시스템의 가장 큰 단점으로 순차적으로 쓰는 도중에 데이터를 저장할 공간이 부족하면 수행하는 가비지 컬렉션이 있다. 가비지 컬렉션은 무효 데이터와 유효 데이터가 섞여 있는 공간에서 유효 데이터만을 다른 공간으로 복사하여 여유 공간을 확보하는 과정이다. 가비지 컬렉션을 수행하는 동안 복사해야하는 유효 데이터가 많을수록 큰 비용이 발행하며 사용자가 느끼는 응답시간 또한 증가한다.

이를 완화시키기 위하여 복사 비용이 없는 Slack Space Recycling[2]이 제안되었다. 이 기법은 쓰기 요청을 처리할 공간이 부족할 경우에 가비지 컬렉션을 수행한 빈 공간에 쓰기 연산을 수행하는 것이 아니라 무효 데이터가 쓰인 공간에 쓰기 연산을 수행하는 기법이다. 복사 비용이 없다는 장점이 있지만 데이터를 순차 쓰기가 아닌 임의 쓰기로 처리하기 때문에 시스템의 쓰기 성능을 저하시킨다.

본 논문에서는 이를 개선하기 위하여 적은 복사 비용으로 많은 순차 쓰기 공간을 확보하는 부분 가비지 컬렉션 기법을 제안하고자 한다.

II. 관련 연구

2.1 플래시 메모리(Flash Memory)

플래시 메모리는 하드디스크에 비해 빠른 처리 속도와 빠른 접근 속도를 지니고 있다. 또한, 비휘발성, 저전력 소모, 가벼움 등의 장점을 지니고 있어, 스마트 기기와 같은 모바일 임베디드 시스템에 널리 사용되고 있다.

플래시 메모리가 위와 같은 이점을 가졌지만 하드웨어 구조 상 몇 가지 제약 사항이 존재한다. 기존의 하드디스크가 덮어 쓰기가 가능하지만 플래시 메모리는 지우기 연산 후 쓰기 연산을 해야 한다. 하지만 쓰기에 사용하는 연산 단위(페이지, page)와 지우기에 사용하는 연산 단위(블록, block)가 서로 달라 새로운 데이터를 쓰기 위해서는 블록 전체를 지운 후 해당 페이지에 쓰거나, 매핑 테이블을 관리하여 새로운 페이지에 데이터를 써야한다. 또한, 각 블록마다 지우기 횟수가 제한되어 있어, 여러 블록에 걸쳐 사용하는 것이 효율적이다.

이러한 플래시 메모리의 특징을 효율적으로 살리기 위해서 쓰기 평준화(wear leveling)나 가비지 컬렉션(garbage collection)등의 기법을 요구한다.

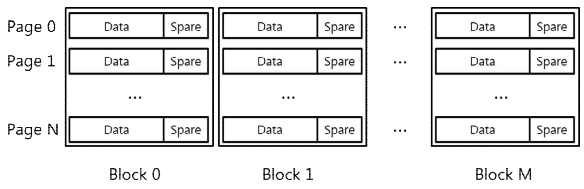


그림 1 플래시 메모리의 구조

2.2 로그 구조 파일시스템

로그 구조 파일시스템은 데이터를 로그에 순차적으로 기록하여 관리하기 때문에 쓰기 성능과 복구에 뛰어나다. 기존 파일 시스템이 메타 데이터 영역과 유저 데이터 영역을 나누어 관리하였던 것이 비해, 로그 구조 파일시스템은 각 데이터를 모두 로그에 기록한다.

업데이트 되는 데이터는 새로운 로그에 기록하고 이전 데이터는 무효 데이터로 처리한다. 사용하지 않는 로그를 제거하고, 새로운 로그를 쓸 공간을 마련하기 위하여 가비지 컬렉션을 한다.

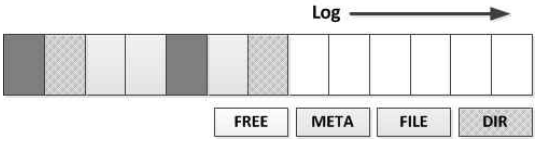


그림 2 로그 구조 파일시스템

본 논문의 실험에서 사용하는 로그 구조 파일시스템인 F2FS(Flash-Friendly File System)[3]는 플래시 메모리를 위한 파일 시스템으로 데이터를 관리하는 임의 쓰기 영역과 실제 데이터가 저장되는 순차 쓰기 영역으로 나누어 사용한다.

F2FS는 데이터와 데이터의 위치를 가리키는 노드를 각각 3개의 종류(Hot/Warm/Cold)로 나누어 종류 별로 순차쓰기를 수행한다.

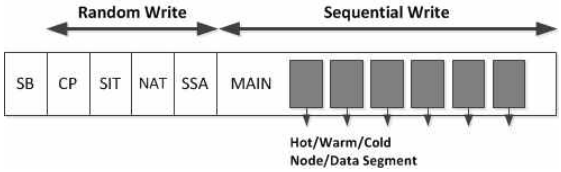


그림 3 F2FS 파일시스템 개요

2.3 가비지 컬렉션

가비지 컬렉션은 무효화된 데이터가 기록된 스토리지 공간을 관리하는 기법으로, 로그 구조 파일시스템에서 쓰레기 수집을 통해 무효 데이터를 제거하고 새로운 로그를 쓸 수 있는 빈 공간을 만든다.

로그 구조 파일시스템에서 가비지 컬렉션 시 사용하는 Copy and Compaction 기법은 희생 세그먼트를 선정하여 유효 데이터를 다른 세그먼트의 빈 공간에 새로 기록하고, 희생 세그먼트 내부의 블록을 모두 삭제하여 새로운 빈 공간으로 만드는 역할을 한다. 디바이스 사용량이 높을수록 복사 비용이 크며 사용자 응답 시간도 크게 증가한다.

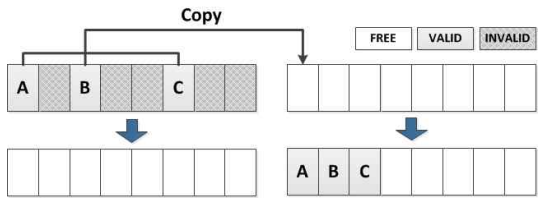


그림 4 Copy and Compaction

유효 데이터를 복사할 때 빈 공간에 기록할 수도 있지만, 무효 데이터가 존재하는 공간에 데이터를 업어 쓰기하는 Slack Space Recycling(SSR) 기법이 제안되었다. 복사 비용이 없지만 임의 쓰기를 유발하여 쓰기 성능을 감소시킨다.

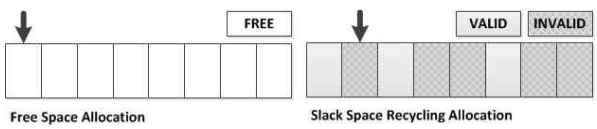


그림 5 로그 구조 파일시스템 공간 할당

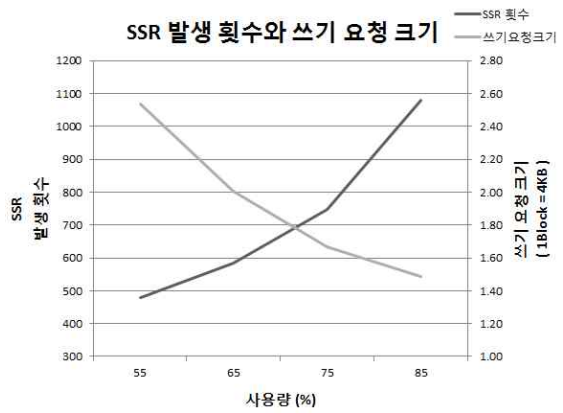
Ⅲ. 본론

3.1 필요성

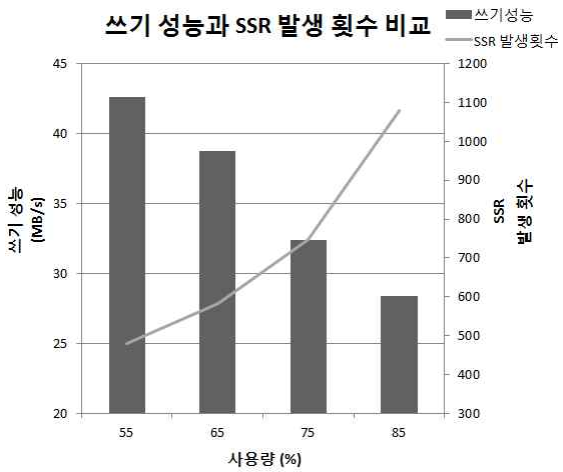
F2FS 파일시스템에서는 쓰레기 수집 시 Copy and Compaction 기법과 함께 SSR 기법을 함께 사용한다.

파일시스템의 사용량이 높아지면 늘어난 무효 공간에 SSR 기법을 이용하여 쓰기 요청을 처리한다. 이때 하나의 세그먼트에는 무효 데이터와 유효 데이터가 섞여 있기 때문에 그래프 1과 같이 쓰기 요청은 작은 크기의 임의 쓰기로 처리된다. 그래프 2에서는 SSR 발생 횟수가 증가하며 성능 저하가 나타나는 것을 보여주는데, 작은 크기의 임의 쓰기는 파일시스템의 쓰기 성능 저하의 요인이 된다.

이를 해결하기 위한 방안으로 쓰기 요청이 발생하기 전에 무효 데이터 사이에 있는 유효 데이터를 다른 공간으로 복사하여 쓰기 성능을 개선하는 부분 가비지 컬렉션을 제안하고자 한다.



그래프 1 사용량에 따른 쓰기 성능과 쓰기 요청 크기 비교



그래프 2 사용량에 따른 쓰기 성능과 SSR 발생 횟수 비교

3.2 부분 가비지 컬렉션

부분 가비지 컬렉션은 작은 크기의 쓰기 요청을 적은 복사 비용으로 많은 순차 쓰기 공간을 확보하기 위하여 고안하였다.

파일시스템의 사용량이 높아지면서 하나의 세그먼트에는 유효 데이터와 무효 데이터가 섞여 나누어진다. 이 세그먼트에 SSR 기법을 사용하여 쓰기 요청을 처리할 경우 쓰기 요청을 연속된 무효 페이지들의 크기에 맞추어 나눈다. 이를 막기 위해 쓰기 요청이 발생하기 전에 무효 데이터 사이에 있는 유효 데이터를 무효 공간으로 미리 복사하여 큰 크기의 무효 공간을 생성한다. 데이터를 복사할 때 사용되는 비용이 존재하므로 일정 크기 이하의 데이터만 복사하여 이익을 최대화한다.

다음 그림과 같이 부분 가비지 컬렉션은 한 세그먼트에 쓰기 요청이 발생할 경우 쓰기 요청을 나누지 않고 한 번에 처리할 수 있다.

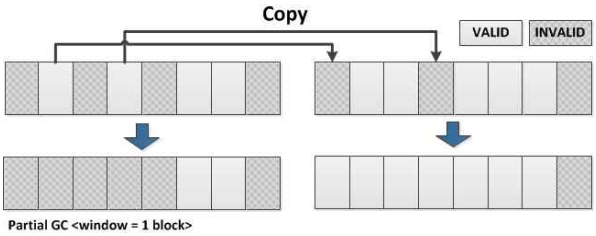


그림 6 부분 가비지 컬렉션

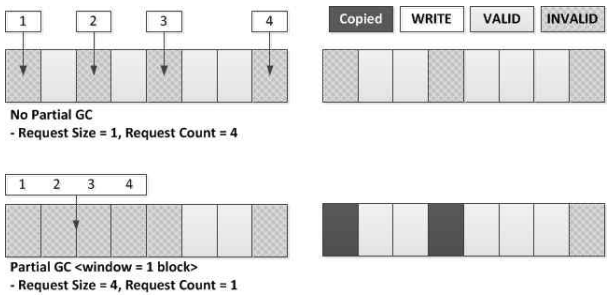


그림 7 쓰기 요청 처리 비용과 쓰기 요청 크기

Ⅳ. 실험 결과

4.1 실험 환경

본 논문에서는 성능 측정을 위하여 상용 스마트폰(갤럭시 S4)의 커널에 F2FS 파일 시스템을 적용하였고, 2400MB 크기의 캐시 파티션에 마운트하여 실험하였다. SSR을 수행하는 환경을 위해 tiobench를 사용하여 1200MB 크기의 파일을 생성하였고 파일 시스템의 사용량을 50%로 맞추었다. 그 다음 4KB 크기의 임의쓰

기를 발생시켜 총 1100MB의 무효 데이터를 생성하여 여유 공간을 5%미만으로 만들었다.

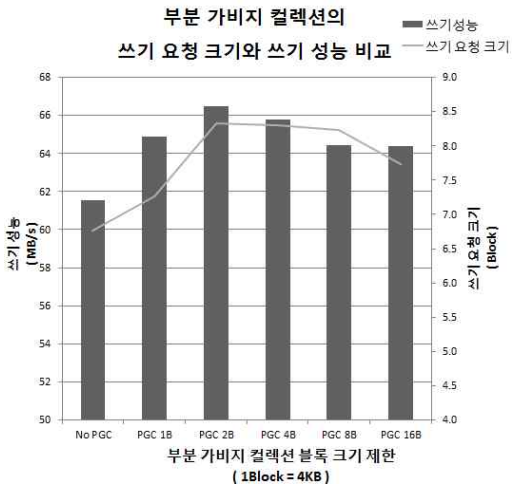
실험은 부분 가비지 컬렉션의 블록 크기 제한을 다르게 하며, tiobench를 이용하여 1MB 크기의 쓰기 요청을 발생시켜 성능을 측정하였다.

또한, 부분 가비지 컬렉션을 사용하기 전의 데이터와 부분 가비지 컬렉션을 사용할 때 유효 데이터를 복사하는 크기 제한을 다르게 하여 얻은 데이터를 비교하였다.

4.2 실험 결과

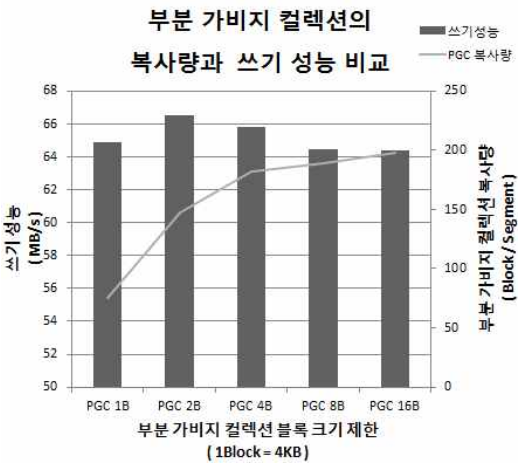
아래의 그래프 3을 살펴보면 부분 가비지 컬렉션을 수행 할 때의 쓰기 성능이 일반 쓰기 성능보다 높은 것을 확인할 수 있다. 또한 부분 가비지 컬렉션을 수행한 후의 쓰기 요청 크기가 일반 쓰기 요청 크기보다 크다는 것을 알 수 있다.

또한 부분 가비지 컬렉션을 통해 쓰기 요청 크기를 증가시키고 이를 통해 쓰기 성능이 개선되는 것을 관찰하였다.



그래프 3 부분 가비지 컬렉션 블록 크기 제한에 따른 쓰기 요청 크기와 쓰기 성능 비교

부분 가비지 컬렉션을 수행하면 수행하지 않을 때보다 쓰기 성능을 개선시킬 수 있지만 블록 크기의 제한에 따라 성능이 달라지는 것을 확인할 수 있다. 이는 부분 가비지 컬렉션에서 유효 데이터를 복사하는 과정에서 발생하는 비용이 존재하기 때문이다. 그래프 4를 살펴보면 블록 크기 제한이 커지면서 세그먼트 당 복사하는 유효 데이터 블록 수가 증가하는 것을 확인할 수 있다. 이로 인해 아래와 같은 실험 조건 중 블록 크기 제한이 2 블록일 때 성능이 가장 좋은 것을 확인할 수 있다.



그래프4 부분 가비지 컬렉션 블록 크기 제한에 따른 세그먼트 당 복사량과 쓰기 성능 비교

V. 결론 및 향후 연구 방향

로그 구조 파일시스템에서 큰 비용이 드는 가비지 컬렉션을 보완하기 위해 Slack Space Recycling 기법이 제안되었지만 이 또한 파일시스템의 쓰기 성능을 악화시킨다.

본 논문에서는 Slack Space Recycling을 이용하여 쓰기 요청이 발생하기 이전에 무효 데이터 사이에 존재하는 유효 데이터를 다른 공간으로 복사하는 부분 가비지 컬렉션을 제안하였다. 이를 통해 쓰기 요청 크기를 증가시키고 결과적으로 쓰기 성능을 개선시켰다.

향후 연구로 로그 구조 파일시스템의 사용량에 따라 적합한 부분 가비지 컬렉션의 블록 크기를 제한하는 방안을 연구할 계획이다.

참고문헌

[1] Mendel Rosenblum and John K. Ousterhout, “The Design and Implementation of a Log-Structured File System”, ACM Transactions of Computer Systems, 1992.

[2] Yongseok Oh, et al. “Optimizations of LFS with slack space recycling and lazy indirect block update”, SYSTOR ,2010.

[3] Flash Memory Filesystem, Korea Linux Forum, Oct. 12, 2012, http://elinux.org/images/8/81/A_New_File_System_Designed_for_Flash_Storage_in_Mobile.pdf