데이터 저널의 멀티스트림화를 통한 MySQL/InnoDB 최적화

최소이 강동현 김경민 서성윤 이상원 엄영익
성균관대학교
{ithdli,kkangsu,lufovic77,rudobco,swlee,yieom}@skku.edu

Optimizing MySQL/InnoDB by Multi-Streamed data journal

Soyee Choi, Dong Hyun Kang, Kyung-Min Kim, Sung-Youn Seo, Sang-Won Lee, Young Ik Eom
SungKyunKwan University

요 약

멀티스트림 SSD는 데이터 별로 쓰기 수명이 다른 점을 활용하여 다른 수명의 데이터를 물리적으로 분리함으로써 SSD 내부에서의 쓰기 증폭으로 인한 성능 저하를 완화하고자 한다. 하지만 멀티스트림 SSD는 수동적인 시스템으로, SSD 내부에서 자동으로 데이터가 분리되는 것이 아니라 호스트에서 데이터의 수명에 따른 힌트를 제공받아야 한다. 본 논문에서는 파일 시스템 저널링의 특성을 이용하여서 호스트에서 별도로 수명을 파악하는 과정이 필요가 없고 응용의 종류에 의존적이지 않은 일반적인 멀티스트림 SSD 활용 방안을 제시한다. 특히 ext4 파일 시스템의 데이터 저널을 사용하여서 파일 시스템 차원에서의 데이터 일관성을 보장하면서 멀티스트림 SSD를 사용하여 성능도 고려하였다. 본 논문에서는 MySQL/InnoDB를 사용하여 성능을 평가를 하였으며 데이터 저널을 이용함으로써 데이터의 일관성을 보장하였기 때문에 MySQL/InnoDB 응용에서의 데이터 일관성 보장 기법인 Double Write Buffer (DWB) 없이 성능을 평가하여 기존의 메타데이터 저널 모드에서 DWB를 켰을 때보다 38% 정도의 쓰기증폭(Write Amplification Factor: WAF) 감소와 31%의 TpmC 개선을 확인하였다.

1. 서 론

Solid State Drive (SSD)는 빠른 읽기/쓰기 성능을 보이며 주류 저장 장치로 자리 잡아가고 있다. 전기적인 SSD는 기존의 기계적인 Hard Disk Drive(HDD)와 몇 가지 차이점을 갖는다. 먼저 SSD는 덮어쓰기를 허용하지 않는다. 이 때문에 SSD는 쓰기 공간을 확보하기 위해 Garbage Collection (GC)를 수행하지만 GC 과정에서의 병목이 SSD의 성능에 악영향을 준다. 한편 SSD는 HDD와 달리 내부 컴퓨팅 자원을 제공하여 새로운 인터페이스 지원이 가능하다. 최신 SSD 인터페이스 기술은 GC 병목의 최소화를 목표로 하며 Multi-Streamed SSD [1] (이하 MS-SSD)가 대표적인 예이다. MS-SSD는 데이터를 수명에 따라 물리적으로 분리하는 인터페이스이다. 하지만 MS-SSD는 매우 수동적이다. SSD가 직접 데이터의 수명을 파악하여 분리하는 것이 아닌 사용자가 MS-SSD에 데이터에 수명에 따라 힌트를 부여해야 한다. 따라서 사용자가 스트림을 어떻게 분리하는가는 중요한 문제이다.

본 논문에서는 MS-SSD의 데이터베이스에의 일반적인 활용 방안에 대한 연구 [2, 3] 의 일환으로 파일 시스템 저널링의 특성을 이용하였다. 특히 메타데이터만을 저널링 하는 Ordered journal (이하 OJ)과 일반 데이터도 함께 저널링하는 Data journal (이하 DJ) 가운데 성능은 다소 낮지만 더 안정적인 DJ 모드에 초점을 맞추었다. MS-SSD를 통해서 DJ의 단점이었던 낮은 성능은 보완하면서 DJ의 장점을 이용해 데이터베이스 응용 단에서의 별도 저널링 없이도 높은 일관성을 보장한다.

2. 배경 지식

2.1 멀티스트림 SSD

SSD는 페이지 단위의 읽기/쓰기를 수행하는 반면 GC과정에서의 지우기는 페이지의 모음인 블록 단위로 수행한다. 이 때문에 GC의 대상 블록에는 지워져서는 안 되는 유효한 페이지가 존재할 수 있다. 유효한 페이지는 별도의 블록에 복사되는 과정인 카피백(copyback)을 통해 보호될 수 있다. 하지만 이 과정에서 쓰기의 증폭 (Write Amplification)은 SSD의 성능 저하의 원인이 된다.

멀티스트림 SSD는 상이한 수명의 데이터를 물리적으로 분리함으로써 쓰기 증폭을 줄이는 기술이다. MS-SSD에서에서 비슷한 수명의 페이지가 동일 블록에 위치하여 해당 페이지들이 비슷한 시점에 무효화됨에 따라 GC 대상으로 선택됐을 때 카피백 해야 하는 페이지의 수가 감소한다. MS-SSD는 기술위원회 T10에서 표준화 되었다. [4]

MS-SSD에서 서로 다른 수명의 데이터는 *스트림*이라는 개념으로 물리적으로 분리된다. 이 때 스트림을 결정하는 것은 SSD의 펌웨어가 아닌 호스트이다. 호스트가 데이터를 쓰기 이전에 스트림 아이디를 부여하면 해당 아이디를 기준으로 데이터가 분리, 위치된다. 따라서 호스트에서 데이터를 분리하는 방법이 중요하다.

대표적으로 호스트에서 데이터를 분리하는 방법은 특정 응용의 데이터의 수명 패턴을 분석하여 구분하는 방법과 운영체제 레벨에서 데이터의 역할에 따른 수명 특성에 따라 스트림을 구분하는 두 가지로 나뉜다. 본 논문에서는 두 번째 방법을 채택하며 이는 응용의 영향을 받는 첫 번째 방법과 달리 일반화 될 수 있다.

2.2 파일 시스템 저널링

파일 시스템은 데이터의 일관성을 보장하기 위해 아직 커밋되지 않은 데이터를 저널 파일을 통해 관리한다. Ext4 파일 시스템에서 디폴트 저널 크기는 128MB로 저널 데이터는 라운드로빈 방식으로 저널 영역에 데이터를

쓴다. 작은 영역에 데이터를 매번 쓰기 때문에 저널 데이터는 수명이 아주 짧은 hot 데이터이다. 하지만 기존의 시스템에서는 핫한 저널 데이터와 비저널 데이터는 상이한 수명을 가짐에도 불구하고 SSD 내부에서 섞인다.

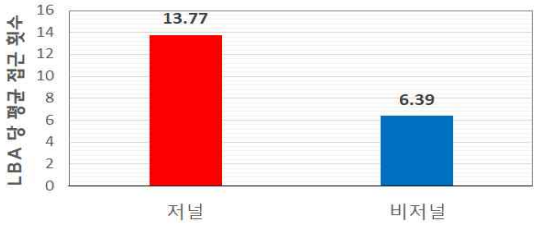


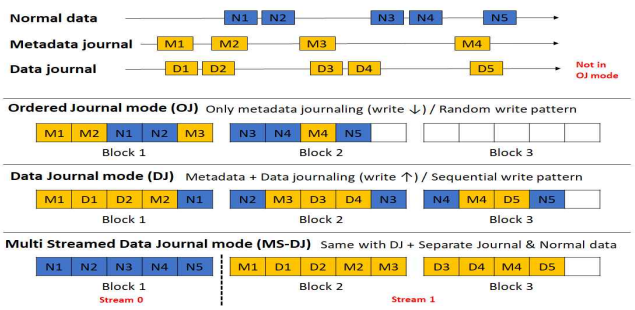
그림 1 저널/비저널 데이터 평균 접근 횟수

실제로 [그림1]은 저널데이터가 더 Hot함을 분석한 그래프이다. MySQL TPC-C를 ext4 데이터 저널모드로 수행하였을 때 LBA당 평균 접근 횟수는 전체 저널 혹은 비저널 데이터가 접근한 횟수를 LBA 범위로 나눈 값으로 하나의 LBA영역이 몇 번 접근하였는지 보여준다. 즉 LBA당 접근 횟수가 2배 정도 되는 저널 데이터가 더 hot함을 확인하였다.

파일 시스템의 저널은 크게 두 가지로 구분된다. 전통적으로 대부분의 파일 시스템은 ext4의 ordered 저널과 같이 메타데이터만을 저널 파일에 쓰는 메타데이터 저널링(Ordered journal)기법을 따랐다. 하지만 메타데이터 저널링은 메타데이터만 저장하므로 데이터의 일관성이 완전히 보장되지는 않는다. 반면 데이터 저널링 (Data journal)은 일반 데이터도 저널링 하기 때문에 높은 데이터 안정성을 보장한다. 하지만 저널 파일에 쓰는 데이터가 늘어나기 때문에 OJ에 비해 성능이 낮다. [5]

본 논문에서는 DJ 모드의 안정성은 유지하면서도 OJ 모다 좋은 성능을 보이는 Multi-Streamed Data Journal (MS-DJ) 디자인을 제시한다. 특히 본 논문에서 DJ 모드에 초점을 맞춘 이유는 다음과 같다. 첫째, MS-DJ 모드는 파일 시스템 차원에서 데이터 안정성을 보장하기 때문에 응용에서 데이터 일관성을 보장하기 위한 추가적인 작업을 수행할 필요가 없다. 일례로, MySQL/InnoDB에서의 Double Write Buffer(DWB)과 같은 응용 차원에서의 저널링의 역할을 MS-DJ가 대체할 수 있다. 둘째, 커밋을 할 때 메타데이터만 유지하고 있어 데이터를 갱신해야 하는 OJ와 달리 DJ는 데이터 갱신의 필요가 없어 연속적인 쓰기의 경향성을 보여 성능 상의 이점을 가진다.

3. Multi-Streamed Data Journal (MS-DJ)



[그림 2] OJ, DJ, MS-DJ 저널 모드에 따른 비교

본 논문에서는 2과 3 장에서의 논의를 바탕으로 새로운 저널링 방식인 'Multi-Streamed Data Journal' (MS-DJ)를 제안한다. Ext4 파일 시스템의 JBD2를 수정하여 MS-DJ 모드로 마운트 되었을 때 저널 데이터의 스트림 아이디를 0번을 가지는 일반 데이터와 달리 1번을 부여했다. [그림 2]는 본 논문에서 제시하는 디자인과 기존의 OJ 모드 DJ 모드의 차이점을 보여준다. 그림에서 알 수 있듯이 OJ모드와 DJ 모드와 달리 MS-DJ에서는 별도의 스트림 아이디를 부여하여 저널의 데이터가 별도의 블록에 분리된다. 한편 일반 저널링 기법에서는 한 블록 내에 저널 데이터가 무효화 되더라도 일반 데이터들이 유효한 상태로 위치할 가능성이 높다.

본 논문과 관련하여 멀티스트림 SSD를 활용하여 저널 데이터를 구분하고자하는 Fstream 기법이 있다. [2] MS-DJ에서는 데이터 저널의 안정성을 활용하여 데이터베이스의 저널링을 대체한다는 점에서 Fstream과 차별화된다. 또한 Fstream은 저널 자체의 최적화에 집중한 아이디어로 저널을 더욱 세부적인 스트림으로 구분하였지만 멀티스트림 SSD가 한정적인 스트림을 제공한다는 점에서 본 논문의 기법인 MS-DJ가 더욱 가치를 갖는다. 실제로 본 논문에서 사용한 멀티스트림 SSD의 경우 최대 16개의 스트림만 제공한다.

4. 성능 평가

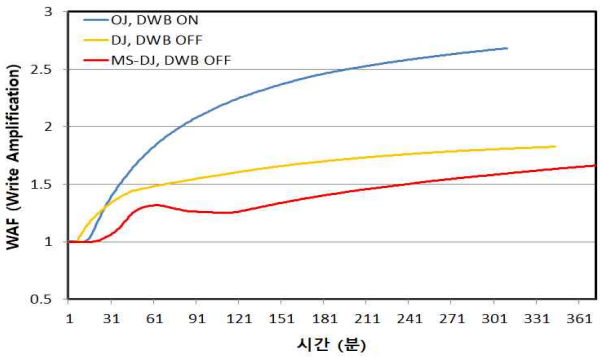
4-1 실험 환경

본 성능평가는 Ubuntu 14.04.5, kenel version 3.19에서 수행되었으며 MS-DJ 저널 옵션을 추가한 Ext4 파일 시스템을 사용하였다. SSD로는 삼성의 PM953 Multi-Stream nvme SSD를 사용하였다. 또한 본 논문에서는 DJ 모드에서 데이터 저널링을 수행하기 때문에 DJ 모드와 MS-DJ 모드에서는 MySQL/InnoDB의 Double Write Buffer를 사용하지 않았다. SSD의 크기가 크기 때문에 WAF를 발생시키기 위하여 FIO 워크로드를 통해 40GB의 영역에 aging을 수행하였다. 실제 실험을 위해서 TPC-C [6] 벤치마크를 150GB의 데이터베이스 크기로 5시간 동안 수행하였다. 본 실험에서는 SSD 내부에서의 쓰기 증폭 표지인 WAF와 MySQL TPC-C에서의 트랜잭션 처리량을 의미하는 TpmC 측면에서의 성능을 평가, 비교하였으며 Data journal mode에서의 저널 쓰기 양이 더 많기 때문에 check point 병목을 고려하여 저널의 크기를 128MB와 5GB로 실험하였다.

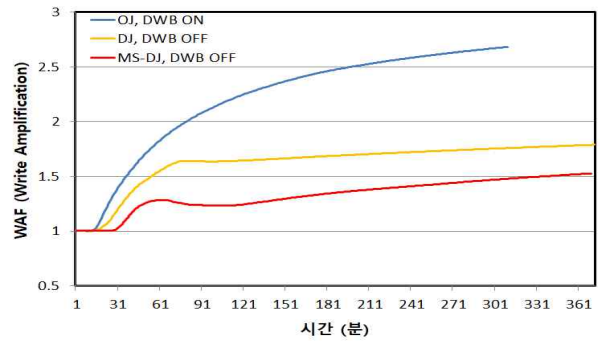
4-2 실험 결과

[표 1] 저널 크기 및 모드에 따른 WAF

	OJ mode DWB ON	DJ mode DWB OFF	MS-DJ mode DWB OFF
128MB	2.681	1.827	1.662
5 GB	-	1.789	1.525



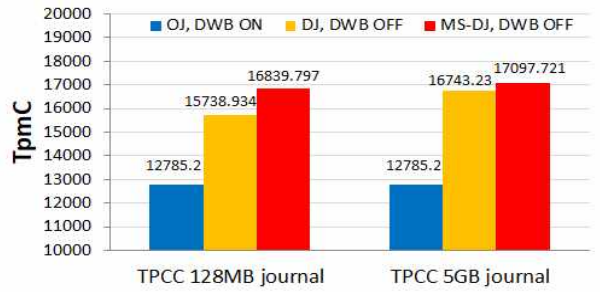
[그림 3] 저널 크기 128MB 일 때의 WAF 변화



[그림 4] 저널 크기 5GB 일 때의 WAF 변화

[표 1]는 저널 크기 및 저널 모드에 따라 5 시간 동안 MySQL TPC-C 벤치마크를 돌린 후에 최종 WAF 값이다. 결론적으로 보았을 때 MS-DJ 모드에서 가장 WAF가 적다. OJ 모드에 비해서는 40% 정도의 WAF 감소량을 보였으며, 128MB의 저널 영역을 부여하였을 때에는 DJ 모드에 비해 9%의 WAF 감소량을 확인할 수 있고 5GB 저널을 부여하였을 때에는 15% 정도 WAF가 감소하였다.

이 때 OJ모드와 큰 차이를 보이는 이유에는 DWB의 부재 역시 큰 영향을 미친다. MySQL/InnoDB 의 DWB는 쓰기 양도 방대할 뿐 아니라 매우 핫한 데이터이다. [3] 저널 데이터 뿐 아니라 DWB를 끔으로써 DWB도 일반 데이터와 분리되게 되어 성능이 크게 개선되었다. [그림 3, 4]는 시간에 따른 WAF의 변화량 그래프이다. 해당 그래프에 따르면 MS-DJ의 WAF가 항상 가장 낮을 뿐 아니라 WAF의 차이가 점차 증가하는 것을 확인할 수 있다.



[그림 5] 저널 크기 및 모드에 따른 TpmC

[그림5]는 평균 TpmC를 비교한 그래프이다. 128MB 저널을 사용하였을 때에는 DJ 모드에 비해 약 7% (약 1100TpmC) 개선을 보였고 5GB 저널에 대해서는 DJ에 비해 약2% (약 340TpmC) 개선을 보였다. 또한 OJ mode에 비해 약 30% 이상의 높은 TpmC 개선을 보였다. 이는 SSD의 WAF 개선에 의한 효과로 볼 수 있다.

5. 결론

MS-DJ는 높은 성능과 높은 데이터 일관성을 보장하는 디자인일 뿐만이 아니라 멀티스트림 SSD를 응용에 관계없이 활용하기 위한 일반적인 방향을 제시한다. 특히 데이터베이스의 저널링와 파일 시스템 저널링을 모두 수행할 필요 없이 파일 시스템 저널링 만으로 데이터의 일관성을 보장할 수 있는 디자인을 제시하였다.

성능 평가 결과 WAF와 TpmC 측면에서 유의미한 성능 개선을 확인할 수 있었다. 기존의 OJ 모드와 비슷한 성능이 아닌 월등한 성능 양상을 보였으며 일반 DJ모드를 사용했을 때보다도 성능이 개선되었다. 다만 WAF의 변화량 그래프를 보았을 때 MS-DJ와 DJ 모드의 그래프가 점차 가까워지는 양상이 보여 해당 부분에 대한 추가적인 연구가 필요할 것으로 보인다.

사사

이 논문은 2017년도 삼성전자(클러스터과제)의 지원을 받아 수행된 연구임.

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 SW컴퓨팅산업원천기술개발사업(SW스타랩)의 연구결과로 수행되었음 (IITP-2015-0-00314)

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW중심대학지원사업의 연구결과로 수행되었음.

참고 문헌

[1] Jeong-Uk Kang, Jeeseouk Hyun, Hyunjoo Maeng, Sangyeun Cho, “Multi-streamed Solid-State Drive“, Hot Strorage, 2014

[2] Eunhee Rho, Kanchan Joshi Seung-Uk Shin, Nitesh Jagadeesh Shetty, Joo-Young Hwang, Sangyeon Cho, Daniel DG Lee, Jaeheon Jeong, “FStream: Managing Flash Streams in the File System“, FAST, 2018

[3] Soyee Choi, Hyeonwoo Park, Sang-won Lee, “Don’t mix pages with different lifetimes in one stream“, edbt demo, 2018

[4] William Martin (T10 Technical editor), SCSI Block Commands-4 (SBC-4), 2015

[5] Yunji Kang, Dongkun Shin, Per-Block-Group Journaling for Imporving Fsync Response Time, UEEE ISCE, 2014

[6] tpcc-mysql benchmark, https://github.com/Percona-Lab/tpcc-mysql, 2008