MySQL/InnoDB의 innodb_flush_neighbors 옵션이 플래시SSD 기반 OLTP 데이터베이스 성능에 미치는 영향

The Effect of innodb_flush_neighbors options in MySQL/InnoDB on FlashSSD-based OLTP Database Performance

저자
(Authors)

신영찬, 박종혁, 이상원
Youngchan Shin, Jonghyeok Park, Sangwon Lee

출처
(Source)

한국정보과학회 학술발표논문집 , 2017.12, 1729-1731 (3 pages)

발행처
(Publisher)

한국정보과학회
KOREA INFORMATION SCIENCE SOCIETY

URL

http://www.dbpia.co.kr/Article/NODE07322663

APA Style

신영찬, 박종혁, 이상원 (2017). MySQL/InnoDB의 innodb_flush_neighbors 옵션이 플래시SSD 기반 OLTP 데이터베이스 성능에 미치는 영향. 한국정보과학회 학술발표논문집, 1729-1731.

MySQL/InnoDB의 innodb_flush_neighbors 옵션이 플래시SSD 기반 OLTP 데이터베이스 성능에 미치는 영향

신영찬, 박종혁, 이상원

성균관대학교 정보통신대학

knesa123@skku.edu, akindo19@gmail.com, swlee@skku.edu

The Effect of innodb_flush_neighbors options in MySQL/InnoDB on FlashSSD-based OLTP Database Performance

Youngchan Shin, Jonghyeok Park, Sangwon Lee
College of Information and Communication Engineering, Sungkyunkwan university

요 약

DBMS는 변경 사항들을 데이터베이스 파일에 기록하기 전 메모리 영역에 임시적으로 저장한다. MySQL InnoDB 스토리지는 주기적으로 buffer pool, redo log, undo log 등을 flush하는데 이 flush를 할 때 여러 option들이 주어진다. 그 중 innodb_flush_neighbors라는 옵션이 있는데 이는 buffer pool에서 페이지가 flush 될 때 같은 extent의 다른 dirty page들을 flush하는 것에 대한 설정 값이다. 본 논문에서는 빈번한 write가 성능의 저하로 이어지는 플래시SSD에서 innodb_flush_neighbors의 옵션이 OLTP 데이터베이스 성능에 얼마나 영향을 미치는지를 알아보고 이를 분석하였다.

1. 서 론

DBMS는 변경 사항들을 데이터베이스 파일에 기록하기 전, 메모리 영역에 임시적으로 저장한다. MySQL InnoDB 스토리지는 주기적으로 저장한 buffer pool, redo log, undo log 등을 flush하는데 이 flush를 할 때 여러 option들이 있다.[1] 그 중 Innodb_flush_neighbors라는 옵션이 있는데 이는 buffer pool에서 페이지가 flush 될 때 같은 범위의 다른 dirty page들을 flush하는 것에 대한 설정 값이다.

이는 HDD(Hard Disk Drive)의 데이터가 sequential한 특징과 seek time이 걸린다는 점에서 같은 extent의 dirty page들을 함께 처리하는 점에서 효율적이지만, 최근 많이 사용되는 저장매체인 SSD(Solid State Drive)는 HDD와는 상이한 동작 방식을 가지고 빈번한 쓰기가 발생할 경우에 성능 저하를 일으키게 된다.

본 논문에서는 MySQL InnoDB에서 여러 옵션 중 innodb_flush_neighbors 옵션이 플래시SSD 기반 OLTP 데이터베이스 성능에 미치는 영향을 확인 및 분석하였다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 SSD의 특성과 innodb_flush_neighbors option에 따른 차이점을 간단히 알아본다. 3장에서는 본 논문에서 수행한 성능 평가 환경과 그 결과를 보이고 이를 분석한다. 마지막으로 4장에서는 결론 및 향후 연구를 소개하고 논문을 마무리한다.

2. SSD와 innodb_flush_neighbors

2.1 SSD의 특성

HDD는 read와 write의 성능이 큰 차이가 없는 반면, SSD는 read가 write에 비해 몇 배는 더 빠르다. 또한 sequential read와 random read에서의 차이도 거의 없다. write의 경우에는 sequential보다 random한 경우가 훨씬 더 느리다.

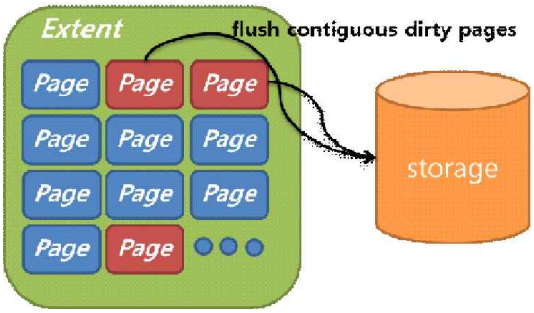
SSD는 write가 많이 발생하게 되면 성능이 떨어지게 된다. 왜냐하면 SSD는 덮어쓰기가 불가능하고 제한된 횟수 이상 쓰기/삭제 시에는 수명을 다하게 되기 때문이다. SSD는 덮어쓰기가 불가능하기에 영역을 초기화 시켜버리는 삭제 연산을 지원하지만, 이는 read나 write에 비해 수십 배에서 크게는 수백 배까지 걸린다.[2] 삭제 연산이 오래 걸리는데다가 제한된 횟수 이상으로 write와 삭제 연산이 발생하면 수명이 다하게 되기 때문에 이를 줄이는 것이 SSD의 성능 면에서는 바람직하다. 게다가 SSD는 page단위로만 read, write가 일어날 수 있는데, 약간의 수정사항이 있더라도 그 페이지 전체를 invalid 시키고 새로운 page에 내용을 write를 하게 된다. SSD에서 invalid된 page들은 나중에 삭제 연산을 통해 초기화시키게 된다.

2.2. innodb_flush_neighbors option

innodb_flush_neighbors는 InnoDB buffer pool에서 페이지가 flush 될 때 같은 범위의 다른 dirty page들을 flush

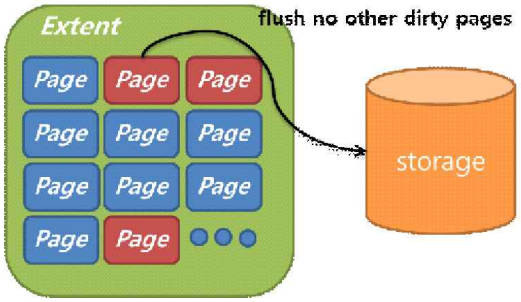
하는 것에 대해 세 가지의 옵션을 제공한다. 이 옵션은 해당 page를 flush할 때 같은 extent의 페이지들을 flush 하는 것에 대한 옵션이다.

innodb_flush_neighbors가 default값인 1이면 buffer pool에서 flush하려는 페이지와 같은 extent의 인접한 dirty page들을 flush하게 된다.[3]



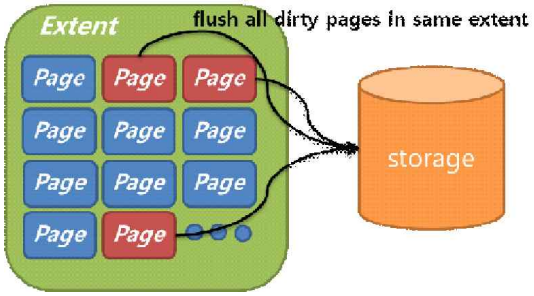
[그림 1] innodb_flush_neighbors=1 (default)

innodb_flush_neighbors의 설정 값이 0인 경우에는 innodb_flush_neighbors를 작동시키지 않는 것으로 buffer pool에서 다른 dirty page들을 flush 시키지 않는다.



[그림 2] innodb_flush_neighbors=0

반면 innodb_flush_neighbors가 2일 경우는 flush할 때 buffer pool의 같은 extent의 dirty page들을 전부 flush 시킨다.



[그림 3] innodb_flush_neighbors=2

HDD는 데이터에 접근할 때에 seek time이 걸리기 때문에 Table data가 HDD 스토리지 기기에 저장되어 있을 때에는 이러한 인접 페이지들을 하나의 작업으로 flush하는 것이 개별적으로 각각의 페이지를 flush하는 것보다 I/O 오버헤드를 줄이게 된다.

반면 SSD는 seek time이 중요한 요소가 아니고, 빈번한 쓰기 연산은 SSD 성능에 악영향을 미치기 때문에 innodb_flush_neighbors 옵션을 쓰지 않는 것이 좋다.[4]

본 논문에서는 이러한 innodb_flush_neighbors option에 따라 실제로 어떠한 성능을 보이는 지 확인하였다.

3. 성능 실험

본 논문에서 데이터 저장장치로 사용한 SSD는 Samsung 850 Pro(256GB)이다. 자세한 실험 환경은 [표 1]과 같다.

[표 1] 실험 환경

운영체제	Ubuntu 16.04.2 LTS
프로세서	Intel(R) Core™ i7-860 CPU @ 2.80GHz
메모리(RAM)	8.00GB
저장장치	Samsung 850 pro SSD
데이터베이스	MySQL 5.6.25
성능 평가 시 사용한 툴	Sysbench

데이터베이스의 크기는 1GB, 5GB, 10GB, 20GB, 50GB로 실험하였다.

innodb_flush_neighbors는 값에 따라 특정 page가 flush 될 경우에 인접 dirty page들을 같이 flush시키거나 같은 범위의 dirty page들을 전부 flush시키게 된다. 이는 HDD에서는 성능향상을 가져오지만 SSD에서는 역효과를 일으키므로, 이 옵션이 SSD에 어떠한 영향을 얼마나 미치는지의 성능을 측정하기 위해 실험은 먼저 innodb_flush_neighbors 옵션에 따라 flush가 얼마나 더 일어나게 되는지를 확인해보았다.

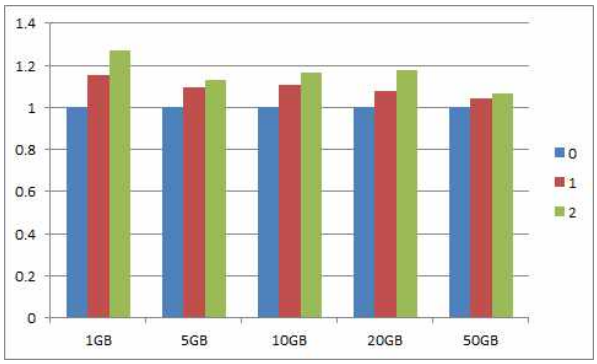
실험은 각 데이터 크기별로 max-request를 1000000으로 했을 때의 innodb_flush_neighbors를 0, 1, 2로 했을 때의 값을 확인 하는 것으로 진행했다. 그 결과, SSD에서 innodb_flush_neighbors 옵션 값에 따른 flush 횟수는 [표 2]와 같다.

[표 2] 데이터양과 옵션 값에 따른 flush 횟수

	innodb_flush_neighbors option		
	0	1	2
1GB	148505	171623	188764
5GB	193548	212482	218332
10GB	692128	765790	807061
20GB	1144276	1236026	1344994
50GB	2799107	2914321	2976420

InnoDB에서 flush가 발생하게 되면 해당 페이지의 내용들을 수정해야 하는데 위에서 언급했듯이 수정해야 하는 page 전체를 invalid시키고 write를 해야 하며, write와 삭제 연산의 비효율성과 제한된 write횟수에 따르면 flush를 자주 하는 것이 큰 성능의 저하를 야기한다.

[표 2]의 데이터를 가시적으로 확인하기 위해 0일 때의 데이터를 1로 설정하고 차이를 그래프로 나타내면 [그림 4]와 같다. 범례의 0, 1, 2는 각각 innodb_flush_neighbors의 값을 의미한다.



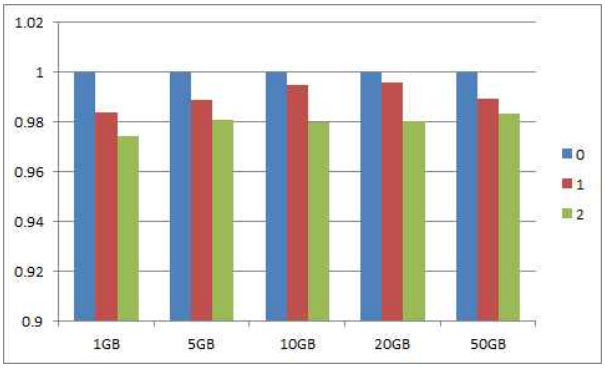
[그림 4] 옵션 0을 기준으로 나타낸 flush 횟수 비율

이러한 flush 횟수의 증가로 인해 나타나는 단위시간당 처리한 transaction 수인 TPS를 확인하기 위해 마찬가지로 max-request를 100만으로 했을 때 SSD에서 innodb_flush_neighbors 옵션 값에 따른 TPS 값을 구한 결과는 [표 3]과 같다. 마찬가지로 범례의 0, 1, 2는 각각 innodb_flush_neighbors의 값을 의미한다.

[표 3] 데이터양과 옵션 값에 따른 TPS

	innodb_flush_neighbors option		
	0	1	2
1GB	2068.68	2035.11	2015.71
5GB	2019.44	1996.82	1980.82
10GB	1315.79	1309.57	1289.55
20GB	914.72	910.79	896.64
50GB	430.02	425.54	422.92

마찬가지로, [표 3]의 데이터의 차이를 확인하기 위해 이번엔 innodb_flush_neighbors 옵션이 0일 때의 데이터를 기준으로 잡고 그래프로 나타내면 [그림 5]와 같다.



[그림 5] 옵션 0을 기준으로 나타낸 TPS 비율

성능 측정 결과 flush의 경우에는 6퍼센트에서 27퍼센트까지 차이가 발생함을 보여주었다. 처리하는 transaction 수가 일정하다보니 데이터가 커질수록 옵션에 따른 차이가 줄어드는데, 이는 데이터가 커지다보니 같은 extent에 있는 dirty page의 수가 그만큼 줄어들기 때문에 발생하는 것으로 보인다. 데이터의 크기가 커지는 만큼 transaction 수 또한 커진다면 비슷한 비율의 차이를 보일 것이라 여겨진다.

TPS의 경우에는 전체적으로는 flush 횟수와 반비례하여 성능이 떨어짐을 보여주지만, 2.5퍼센트 정도만의 성능 저하를 보여주어 flush의 차이에 비하면 예상보다 차이가 크지 않다. 그렇지만 2장에서 보았듯이 삭제연산의 경우 read나 write에 비해 수십 배에서 수백 배 걸리게 되는데 flush가 계속 일어나 삭제연산이 필요해지는 정도가 된다면 TPS 또한 큰 차이가 보일 것이다. TPS 또한 flush처럼 데이터 크기가 커질수록 차이가 줄어드는 것은 transaction수가 데이터에 비례해 커지지 않아 innodb_flush_neighbors의 영향이 줄어들어 그러한 결과가 나오는 것으로 보이며, 데이터 결과가 고르게 비례하지 않는 이유는 Sysbench 워크로드의 수행결과가 랜덤하게 일어나기 때문으로 보인다.

추가로 SSD의 특성 중 일정 횟수 이상의 write연산과 삭제연산이 반복되면 수명이 다하는 것으로 미루어보았을 때 flush 횟수의 차이는 SSD의 전체 사용으로 보았을 때 상당히 의미가 있는 값이라 생각되어진다.

5. 결론

본 논문에서는 MySQL/InnoDB의 innodb_flush_neighbors 옵션이 플래시SSD 기반 OLTP 데이터베이스 성능에 미치는 영향을 측정하였다.

성능 측정 결과, 옵션에 따라 SSD에서 데이터가 1GB일 때 flush횟수가 최대 27퍼센트까지 차이가 남을 확인해볼 수 있고, TPS의 경우는 1GB일 때에는 2.5퍼센트의 성능 저하가 생겨, flush횟수에 비하면 차이가 적어 보이지만 삭제연산이 들어가게 된다면 어느 정도 차이가 날 것으로 보인다.

현재 MySQL InnoDB에서는 SSD에서 사용하기에 문제가 되는 부분들이 있어 보인다. 따라서 향후 연구에서는 SSD에서 MySQL InnoDB insert buffer를 사용할 때 flush를 줄임으로써 성능을 높이는 연구를 할 계획이다.

사사

- 스타랩 사사: 본 연구는 미래창조과학부 및 정보통신기술진흥센터의 SW컴퓨팅산업원천기술개발사업(SW스타랩)의 연구결과로 수행되었음 (IITP-2015-0-00314).
- SW중심대학 사사: 본 연구는 미래창조과학부 및 정보통신기술진흥센터의 SW중심대학지원사업의 연구결과로 수행되었음 (2015-0-00914)

참 고 문 헌

[1] InnoDB flush
https://dev.mysql.com/doc/refman/5.6/en/glossary.html#glos_flush
[2] Hwanggyo Lee, Sang-won Lee. “SSD Write Optimization using MySQL Insert Buffer.“ 한국정보과학회 학술발표논문집, (2017.06): 319-321.
[3] InnoDB options
https://dev.mysql.com/doc/refman/5.7/en/innodb-parameters.html
[4] InnoDB flush
https://dev.mysql.com/doc/refman/5.7/en/innodb-lru-background-flushing.html