로그 디바이스 종류에 따른 동시성 제어 기법 성능 비교

전성빈 조태광 박종혁 이상원
성균관대학교 소프트웨어대학
{jsb9344, jo940114, akindo19, swlee }@skku.edu

Performance Comparison of Concurrency Control policies by Log Device type

Sungbeen Jeon Taegwang Jo Jong-Hyeok Park Sang-Won Lee
Sungkyunkwan University

요 약

바이트 단위 쓰기 접근이 가능한 비휘발성 저장장치인 NVDIMM 덕분에 트랜잭션의 가장 큰 병목이었던 로깅 오버헤드가 줄어들었고, 동시성 지원 보장을 위한 지연이 마지막 병목으로 남게 되었다. 본 논문에서는 로그 디바이스를 NVDIMM과 SSD로 설정 하여 로그 디바이스 종류에 따른 FIFO와 CATS 락 스케줄링 기법의 성능을 측정하였다. NVDIMM을 적용했을 때 로깅 오버헤드의 감소로 락 스케줄링의 성능 차이가 발생함을 확인하였다.

1. 서 론

바이트 단위 쓰기 접근이 가능한 비 휘발성 저장장치인 NVDIMM (Non Volatile Dual-Inline Memory Module)의 등장은 트랜잭션 로깅의 디자인 원칙을 근본적으로 변화시키고 있다. 데이터 쓰기가 완료되면 로그 레코드의 지속성 (Durability)이 보장되기 때문에 트랜잭션 커밋 직전에 로그를 플러시 해야 하는 *flush-before-commit* 기법은 더 이상 필요하지 않게 되었다 [1]. 비 휘발성 덕분에 트랜잭션의 가장 큰 병목이었던 로깅 오버헤드가 줄어들었으며 그 결과, 고성능 트랜잭션 지원을 위한 병목으로 동시성 지원 보장을 위한 지연이 마지막 병목으로 남게 된다.

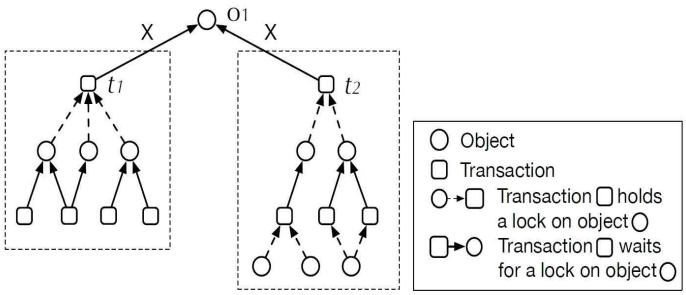
효율적인 동시성 지원 보장을 위해 최근 출시된 MySQL 8.0은 기존의 FIFO (First In First Out) 기반의 락 스케줄링 기법을 CATS (Contention Aware Transaction Scheduling) 기법으로 변경 하였다 [2]. CATS 기법은 더 높은 컨텐션을 가진 트랜잭션에게 lock을 먼저 부여함으로써 트랜잭션 처리량을 높이는 기법이다 [3].

NVDIMM을 로그 디바이스로 사용하여 로깅 병목이 줄어든 환경은 동시성 제어 기법의 성능에 큰 영향을 미칠 것이다. 본 논문에서는 MySQL 8.0에서 로그 디바이스로 각각 NVDIMM과 SSD를 사용하였을 때 FIFO와 CATS 기법의 성능을 비교분석 하고, 동시성 제어 기법의 성능 차이를 확인한다.

본 논문의 구성은 다음과 같다. 2장에서는 MySQL 8.0에 도입된 CATS 기법에 대해 알아보고, 기존의 FIFO 기법과 비교 한다. 3장에서는 로그 디바이스의 종류에 따른 CATS 기법과 FIFO 기법의 성능을 비교 한다. 마지막으로 4장에서는 결론을 제시하고 논문을 마무리한다.

2. Contention Aware Transaction Scheduling

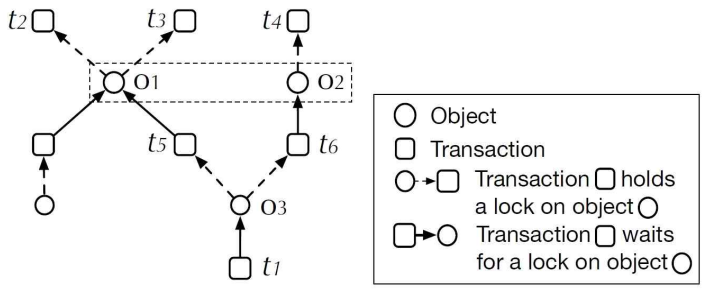
CATS는 더 높은 컨텐션을 가진 트랜잭션에 lock을 부여함으로써 트랜잭션 처리량을 높이는 락 스케줄링 기법이다. CATS는 종속되어 있는 트랜잭션이 많을수록 컨텐션이 높다고 판단한다.



[그림 1] Exclusive lock의 종속성 그래프

[그림1]에서 동일한 데이터 오브젝트에 두 개의 트랜잭션이 Exclusive lock을 요청하고 있다. 트랜잭션 *t1*에 종속되어있는 트랜잭션의 개수는 4개이고 트랜잭션 *t2*에 종 속되어있는 트랜잭션의 개수는 3개이다. CATS는 종속되어있는 트랜잭션의 개수가 많은 *t1*이 컨텐션이 높다고 판단하고 lock을 우선적으로 부여한다. 하지만 종속되어 있는 트랜잭션의 개수로만 우선순위를 부여하는 것은 항

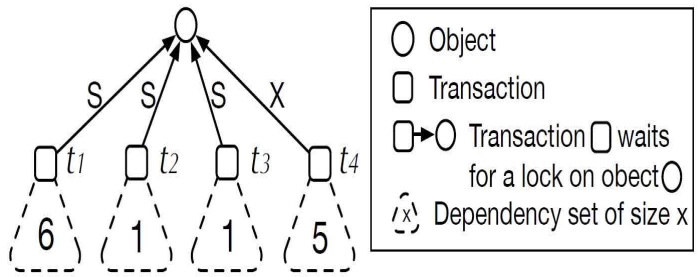
상 최적의 결과를 도출하지 않는다.



[그림 2] Shared lock의 종속성 그래프

[그림 2]에서 트랜잭션 *t1*은 트랜잭션 *t2*에 종속되어있다 라고 판단할 수 없다. 트랜잭션 *t2*가 종료되더라도 만약 트랜잭션 *t3*가 오브젝트 O1의 lock을 계속 가지고 있다면 트랜잭션 *t1*은 트랜잭션 *t2*에 종속되어있지 않는다.

따라서 CATS는 Shared lock의 경우 Shared lock을 요청하고 있는 모든 트랜잭션에 종속되어있는 트랜잭션의 개수에서 모든 트랜잭션이 전부 종료되는 예상시간을 나눈 값을 종속성의 크기로 정의한다. 종료되는 예상시간은 근사함수를 활용하여 결정한다.



[그림 3] shared lock과 exclusive lock의 종속성 그래프

[그림 3]의 경우 Shared lock의 종속성의 크기는 트랜잭션 *t1,t2,t3* 에 종속되어있는 모든 트랜잭션의 합인 8에서 트랜잭션이 종료되는 예상시간을 나눈다. 근사함수 f(3)의 값을 2라고 가정하면 트랜잭션 *t1,t2,t3*의 종속성의 크기는 8 / 2 = 4 가 된다. 트랜잭션 *t4*의 종속성의 크기는 5이므로 CATS는 트랜잭션 *t4*가 컨텐션이 높다고 판단하여 트랜잭션 *t4*에게 lock을 부여한다.

3. 성능평가 및 분석

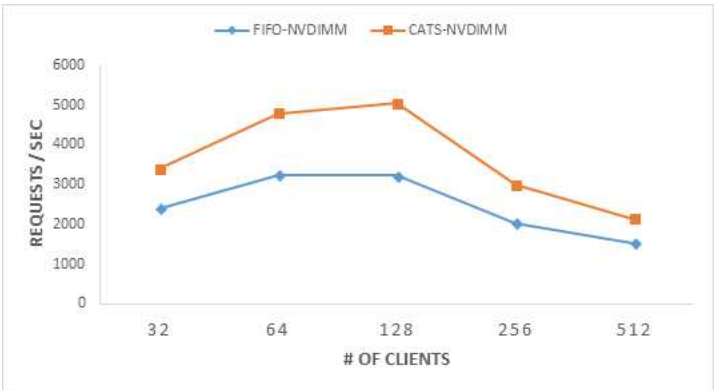
본 논문에서는 MySQL 8.0 소스 코드를 수정하여 락 스케줄링 기법을 FIFO로 설정하였고, Linkbench 벤치마크 [4]를 통해 CATS 기법과 성능을 비교하였다. 벤치마크에 사용한 워크로드는 읽기와 쓰기의 비율이 7:3인 읽기 위주의 워크로드이다. 쓰레드 별 리퀘스트 횟수는 50,000이며, 버퍼 크기는 5GB로 설정하였고, 데이터베이스 크기는 3GB이다. 로그 디바이스는 각각 Samsung SSD 850 PRO와 Netlist NVvault DDR4 NVDIMM을 사용하였고, 자세한 실험 환경은 [표 1] 과 같다.

운영체제	Ubuntu 16.04.2 LTS
프로세서	Intel® Xeon E5-2640 2.60GHz (32 Core)
메모리(RAM)	32GB
저장장치	Samsung SSD 850 Pro 256GB
NVDIMM	Netlist NVvault DDR4 16GB
벤치마크	Linkbench

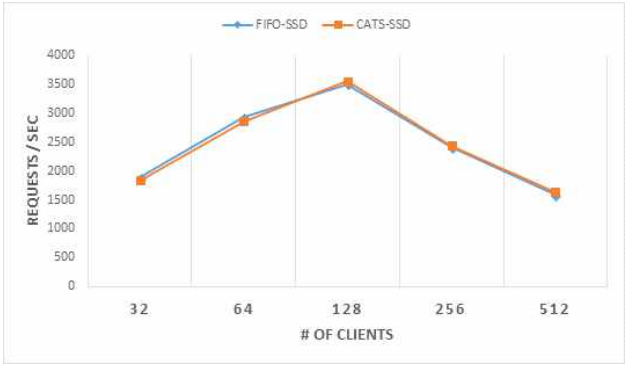
[표 1] 실험환경

본 성능 평가에서는 동시성 제어 기법의 성능 차이를 확인해보기 위하여 MySQL의 로깅 및 DWB (Double Write Buffer) 등 서버 환경 설정을 동일하게 기본 값으로 설정하였고, 동시성 제어 기법만 달리하여 실험을 수행하였다. 또한, 컨텐션을 조절하기 위해 클라이언트 개수를 32부터 512까지 설정하여 CATS와 FIFO 기법의 성능 차이를 확인하였다.

성능 측정 결과, 로그 디바이스를 NVDIMM으로 설정했을 때 초당 리퀘스트 횟수 (Requests/Sec)가 높게 나왔으며, [그림 4]에서 알 수 있듯이, 128 클라이언트일 때, CATS 기법의 성능이 FIFO 기법보다 최대 1.6배 성능 차이가 있음을 알 수 있다. 또한 컨텐션이 낮은 경우에도 최대 1.4배 성능의 차이가 있었으며, 컨텐션이 높은 512 클라이언트일 때도 최대 1.3배 성능 차이를 확인할 수 있었다.



[그림 4] NVDIMM을 로그 디바이스로 사용하였을 경우 초당 리퀘스트 횟수



[그림 5] SSD를 로그 디바이스로 사용하였을 경우 초당 리퀘스트 횟수

반면, 로그 디바이스를 SSD로 설정하였을 경우 **[그림5]**에서 볼 수 있듯이, CATS 기법과 FIFO 기법의 성능차이는 없었다. 컨텐션이 적은 경우에서는 오히려 FIFO 기법보다 성능이 감소했다. 왜냐하면 CATS 기법에서 락 스케줄링 기법에 사용되는 근사함수 계산의 오버헤드가 FIFO 기법보다 크기 때문이다. 컨텐션이 높은 경우, NVDIMM을 로그디바이스로 설정하였을 경우에는 FIFO 기법보다 성능향상이 있었지만, SSD에서는 FIFO기법과 동일하게 성능이 감소하는 것을 알 수 있다.

4. 결론 및 향후 연구

본 논문에서는 로그 디바이스를 NVDIMM과 SSD로 설정 하여 로그 디바이스 종류에 따른 FIFO와 CATS 락 스케줄링 기법의 성능을 비교 하였다. 실험 결과, 로그 디바이스가 NVDIMM으로 설정하였을 경우, 최대 1.5배까지 성능 차이가 있으며 컨텐션이 높은 경우에도 1.3배 까지 성능 차이가 있었다. 반면, SSD를 로그 디바이스로 설정할 경우, FIFO 기법과 큰 성능 차이가 없었다.

NVDIMM을 로그 디바이스로 설정하여 로깅 오버헤드가 줄어든 경우, 동시성 제어 기법의 차이가 전체 트랜잭션 성능에 큰 영향을 미친 것을 확인 하였다.

향후 연구로는 NVDIMM이 장착된 컴퓨팅 환경에서 효율적인 동시성 제어 기법에 대한 연구를 진행할 것이다.

사사

이 성과는 2018년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2018R1A2B2005502).

이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 한국연구재단-차세대정보·컴퓨팅기술개발사업의 지원을 받아 수행된 연구임 (No. NRF-2015M3C4A7065696).

이 논문은 2018년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (2017R1D1A1B03028426).

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

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW중심대학지원사업의 연구결과로 수행되었음 (2015-0-00914)

5. 참고문헌

[1] Wang, Tianzheng, and Ryan Johnson. "Scalable logging through emerging non-volatile memory" Proceeding of the VLDB Endowment 7.10 (2014): 856-876.

[2] MySQL 8.0 Release Notes. (2017). "Changes in MySQL 8.0.3 (2017-09-21. Release Candidate)" https://dev.mysql.com/doc/relnotes/mysql/8.0/en/news-8-0-3.html (2018-4-20 방문)

[3] Tian, Boyu, et al. "Contention-aware lock scheduling for transactional databases." Proceedings of the VLDB Endowment 11.5 (2016): 648-662.

[4] Armstrong, Timothy G., et al. "LinkBench: a database benchmark based on the Facebook social graph." *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. ACM, 2013.