로그 저장장치에 따른 메인 메모리 DBMS 동시성 제어기법 성능 평가

김경민 ○ 박종혁 이상원
성균관대학교
{lufovic77, akindo19, swlee}@skku.edu

Performance evaluation on concurrency control techniques of main-memory DBMS based on log device

Kyungmin Kim○ Jonghyeok Park Sang-Won Lee
Sungkyunkwan University

요 약

멀티 코어, 비휘발성 메모리 등 컴퓨팅 환경의 급격한 변화와 함께 응용에서 생성하는 데이터의 종류와 양은 기하급수적으로 늘어나고 있다. 멀티 코어 환경에서는 대용량 데이터를 효율적으로 처리하는 트랜잭션 처리가 중요하다. 한편, 비휘발성 메모리를 로그 저장장치로 활용하면 로그 쓰기 지연을 제거할 수 있다. 그 결과 트랜잭션 수행에 있어 최종적으로 남는 지연은 동시성 제어 기법에 의해 발생하는 지연이다. 본 논문에서는 실제 상용 비휘발성 메모리인 DCPMM을 사용하여 로그 저장장치의 종류에 따른 동시성 제어 기법의 성능을 비교한다.

1. 서 론

최근 컴퓨팅 환경은 단일 칩에 수십 개의 코어가 있는 매니코어 (Many Core) 시스템이 지배적이고, 높은 네트워크 대역폭을 바탕으로 대용량의 데이터를 다루는 빅데이터 응용에서는 고성능 트랜잭션 처리의 필요성이 대두되고 있다. 한편, DRAM과 비슷한 성능을 제공하고 영속성을 보장하는 비휘발성 메모리를 저장장치로 사용할 수 있게 되었다.

비휘발성 메모리에 로그 레코드 쓰기가 완료되면 지속성 (Durability)이 보장되기 때문에 트랜잭션이 커밋 (Commit) 이전에 로그를 플러시 (Flush) 하지 않아도 된다[1]. 그 결과, 트랜잭션 수행에 있어 최종적으로 남는 지연은 동시성 제어 기법에 의해 발생하는 지연이다. 따라서 로깅 병목이 사라진 트랜잭션 환경이 동시성 제어 기법들의 트랜잭션 처리 성능에 미치는 영향을 파악할 필요가 있다.

선행연구[2]는 상용 비휘발성 메모리가 아닌 DRAM 에뮬레이션을 기반으로 성능평가를 수행하였다. 본 논문에서는 에뮬레이션 기반이 아닌 실제 상용 비휘발성 메모리를 로그 저장장치로 활용하여 로그 저장장치에 따른 동시성 제어 기법의 성능을 평가 및 분석한다. 사용하는 비휘발성 메모리는 Intel에서 출시한 Optane DC Persistent Memory Module (DCPMM)[3]로 DRAM 대비 읽기 지연은 최대 2.5배, 쓰기 지연은 최대 5배[4]를 가진다.

* 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2022R1A2C2008225).

본 논문의 구성은 다음과 같다. 2장은 DCPMM에 대해 설명하고 3장은 동시성 제어기법을 다룬다. 4장에서는 TPC-C 실험 환경과 결과를 설명하고 5장에서는 성능 차이가 나는 이유를 분석해본다. 마지막으로 6장에서는 결론과 향후연구를 제시하고 논문을 마무리한다.

2. Intel® Optane™ DC Persistent Memory (DCPMM)

DCPMM은 Intel에서 출시한 비휘발성 메모리로 DRAM과 유사한 성능을 가지고 최대 512GB까지 영속성을 제공하는 장치이다[4]. DCPMM은 DIMM 인터페이스상의 메모리 버스에서 load() 및 store() 연산을 통해 영속성을 보장한다. DCPMM을 저장장치로 사용하는 경우, DBMS는 낮은 지연과 높은 트랜잭션 처리량을 보장할 수 있다.

DCPMM은 세가지 모드를 제공한다: Memory, App Direct, 그리고 Dual 모드[3]. Memory 모드는 DCPMM을 L4 캐시로써 활용한다. DRAM에서 우선 데이터를 찾고 cache miss가 발생하는 경우 DCPMM을 탐색하는 방식으로 동작한다. App Direct 모드는 Memory 모드와 달리 영속성을 보장하고 사용자에게 영속성을 보장하기 위한 동작 (clflush 및 메모리 연산 순서 보장 연산)이 추가로 필요하다. Dual 모드는 Memory와 App direct 모드를 다양한 비율로 혼합해서 사용할 수 있는 모드이다.

3. 동시성 제어 기법

본 장에서는 동시성 기법들 중 락킹 (Locking)을 기반으로 하는 2PL (Two-Phase Locking) 기법의 NO_WAIT (Non-waiting Deadlock Prevention) 버전과 타임스탬프 (Timestamp)를 기반으로 하는 OCC (Optimistic Concurrency Control) 기법의 구현 중 하나인 Silo의 특징과 동작과정에 대해 설명한다.

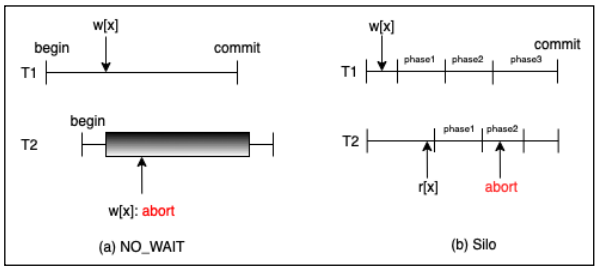

그림 1 동시성 제어 기법 동작 비교

3-1 NO_WAIT

2PL 기법은 교착상태 (Deadlock) 발생 시 트랜잭션의 처리 방법에 따라 세부적인 구현 버전이 존재하는데, 그중 NO_WAIT 버전은 교착상태가 발생하는 상황 자체를 만들지 않으려 한다. 2PL 기법은 읽기와 쓰기 시도 시 항상 락을 얻어야 하는데 NO_WAIT은 락을 얻으려는 트랜잭션의 요청이 실패하면 해당 트랜잭션을 곧바로 중지 (Abort) 시켜 락을 기다리지 않게 한다. 그림1 (a)과 같이 T1이 오브젝트 x에 대한 락을 소유하고 있는 경우, T2가 x에 대한 락을 요청한다면 즉시 트랜잭션 T2을 중지 시킨다.

3-2 Silo

Silo 기법은 낙관적 동시성 기법인 OCC 기법을 구현하였기 때문에 충돌이 발생하지 않는다면 정상적으로 커밋 되지만, 충돌이 발생하는 경우 이를 해결하기 위해 epoch과 TID (Transaction ID)을 사용한다. Epoch은 40ms 가량의 짧은 시간 단위로, 모든 쓰레드가 접근할 수 있는 전역 epoch과 각 워커 쓰레드가 가지고 있는 지역 epoch이 있다. TID는 트랜잭션과 레코드의 버전을 나타내어 현재 가지고 있는 상태로부터 수정이 있었는지를 확인할 수 있다[5]. 이 정보들을 가지고 트랜잭션의 순서를 결정하며 세 커밋 단계를 거쳐 트랜잭션을 반영할지 중지할지 결정한다. 그림 1 (b)와 같이 T1이 오브젝트 x에 대한 쓰기를 시도하면 phase1에서 x에 대한 락을 취득한다. 이후 T2가 x를 읽으려는 시도를 할 때 phase2에서는 트랜잭션의 read set에 있는 레코드들의 TID를 검사하여 수정이 있었는지 여부를 검사하기 때문에, 트랜잭션은 중지된다.

4. TPC-C 실험 결과

4-1 실험 환경

본 논문에서는 오픈소스 메인 메모리 DBMS인 DBx1000[6]에서 TPC-C 벤치마크를 수행하였으며, 로그 디바이스로 SSD와 비휘발성 메모리 (RAMDISK와 DCPMM)를 활용하였다. RAMDISK는 선행연구[2]에서 비휘발성 메모리를 DRAM 에뮬레이션 한 것이다. 데이터베이스의 크기는 500 warehouse이고 클라이언트 수는 48로 설정하였다. DCPMM은 각 소켓마다 128GB DIMM을 장착하였다. 자세한 실험 환경은 표 1와 같다.

표 1 실험 환경

OS	Ubuntu 20.04.3 LTS
Processor	Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz
Memory	187 GB
SSD	Intel SSDSC2KB24 (240 GB)
DCPMM	Intel Optane DCPMM 100 series (128 GB * 2)
Benchmark	TPC-C (500 warehouse, 50GB)

4-2 실험 결과

그림 2와 3은 TPC-C 벤치마크 수행 후 락킹 기반의 동시성 제어 기법인 NO_WAIT와 낙관적 동시성 제어 기법인 Silo의 트랜잭션 처리량이다.

그림 2은 로깅을 비활성화 시켰을 때 (No Logging) 트랜잭션 처리량이다. Silo 기법의 성능이 NO_WAIT보다 약 18% 높은 것을 확인할 수 있다.

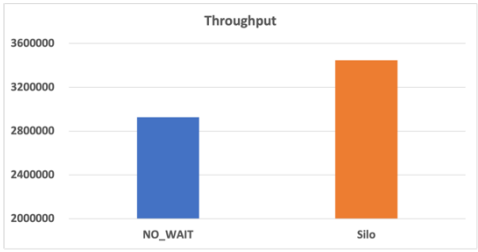

그림 2 No Logging 트랜잭션 처리량

그림 3은 로깅을 활성화한 결과이며 메인 메모리 DBMS 기반 로깅 기법인 Taurus[2]을 사용하였다. NO_WAIT과 Silo 모두 로그 디바이스로 SSD보다 비휘발성 메모리를 사용하였을 때 월등히 높은 트랜잭션 처리량을 가지는 것을 확인할 수 있다.

한편, 로그 디바이스로 SSD를 사용하였을 경우 트랜잭션 처리량은 NO_WAIT과 Silo가 큰 차이가 없음을 확인할 수 있다. 반면 비휘발성 메모리를 사용할 경우 NO_WAIT 기법이 Silo에 비해 RAMDISK와 DCPMM 각각 2.4, 1.5배 더 높은 트랜잭션 처리량을 제공하는 것을 볼 수 있다.

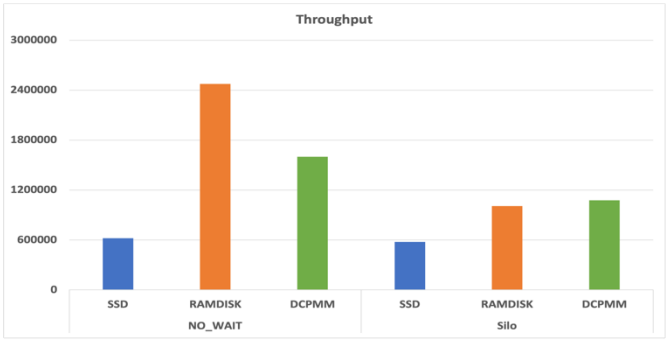

그림 3 Taurus Logging 트랜잭션 처리량

표 2 Abort ratio

	NO_WAIT	Silo
No Logging	0.00929	0.03085
SSD	0.00050	1.46235
RAMDISK	0.00429	0.01273
DCPMM	0.00738	0.01209

표 2는 위에서 수행한 TPC-C 벤치마크에서 로깅 여부와 로깅 장치, 그리고 동시성 기법에 따라 기록한 abort 비율을 나타낸다. Abort 비율은 중지된 트랜잭션 수를 커밋 된 트랜잭션 수로 나누어 구하였다. 모든 경우에서 락킹을 기반으로 하는 NO_WAIT 기법이 Silo 기법에 비해 낮은 abort 비율을 보이고 있다. 트랜잭션은 커밋 되기 전에 여러 번 중지될 수 있기 때문에 값은 1을 넘어갈 수 있다.

5. 성능평가 결과분석

그림 3에서 알 수 있듯이 로그 디바이스로 RAMDISK와 DCPMM을 활용했을 때 SSD에 비해 트랜잭션 처리량이 최소 약 2배, 최대 약 4배 증가하는 것을 알 수 있다. 성능 향상의 주된 원인은 비휘발성 메모리가 SSD보다 낮은 쓰기 지연을 갖기 때문이다.

SSD는 구조상 영속성을 보장하려면 높은 쓰기 지연이 발생하기 때문에 트랜잭션 처리량은 동시성 제어 기법과 상관 없이 로그 쓰기 성능에 의해 결정된다. 반면, 비휘발성 메모리는 영속성도 보장하는 동시에 DRAM과 유사한 대역폭과 쓰기 지연 시간을 지원하기 때문에 동시성 제어 기법이 트랜잭션 처리 성능에 미치는 영향이 커진다.

그림 2처럼 로깅을 비활성화하면 트랜잭션 abort 시에도 로깅 지연이 전혀 발생하지 않는다. 따라서 커밋 직전 충돌 발견 시 트랜잭션을 중지시키고 재시작 하는 Silo 기법이 표 2처럼 높은 중지 비율을 가짐에도 락킹을 기반으로 하는 NO_WAIT에 비해 높은 트랜잭션 처리량을 제공한다. 그림 3처럼 로깅을 활성화 하였을 때는 비휘발성 메모리를 사용했음에도 불구하고 예상과 다르게 남아있는 로그 쓰기 지연으로 인해 abort 비율이 높은 Silo의 성능이 NO_WAIT보다 낮은 것을 볼 수 있다. 이는 잦은 트랜잭션 중지와 이때 발생하는 로깅 지연이 CPU 사이클 낭비를 유발하고 결국 전체 트랜잭션 성능 저하로 이어지기 때문이다.

특히, DRAM 에뮬레이션을 바탕으로 한 RAMDISK의 결과는 NO_WAIT가 Silo보다 최대 2.4배 더 좋은 성능을 가진다. 하지만 이는 잘못된 쓰기 지연 시간을 가정한 에뮬레이션의 결과이다. 실제 상용 비휘발성 메모리인 DCPMM을 사용한 결과 최대 1.5배 성능 향상을 가지는 것을 확인할 수 있다. 이는 DCPMM이 가지는 최대 900ns의 쓰기 지연과 하드웨어적 특성(256B의 XPBuffer[3] 등)이 트랜잭션 처리 성능에 영향을 미쳤기 때문이다.

6. 결론

본 논문에서는 DCPMM을 활용하여 동시성 제어 기법이 트랜잭션 처리에 미치는 영향을 알아보기 위해 로그 저장장치에 따른 동시성 제어 기법의 성능 평가를 수행하였다. 로그 저장장치로 비휘발성 메모리를 사용하였을 때 SSD에 비해 로깅 병목이 줄어들기 때문에 동시성 제어 기법이 트랜잭션 성능을 결정하는 것을 확인하였다. 특히, 로그의 쓰기 지연과 병목을 줄여주는 비휘발성 메모리를 사용하더라도 결국 abort 발생 비율이 적은 락킹 기반의 NO_WAIT 기법의 성능이 더 높은 것을 확인하였다. 또한, DRAM 기반의 에뮬레이션이 아닌 상용 DCPMM을 활용한 실험을 통해 실제 비휘발성 메모리의 쓰기 지연을 고려한 동시성 제어 기법의 필요성을 확인하였다.

향후연구로는 DCPMM 뿐만 아니라 차세대 비휘발성 메모리를 고려한, 효율적인 새로운 동시성 제어 기법에 대한 연구를 수행할 것이다.

참고 문헌

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

[2] Xia, Yu, et al. "Taurus: Lightweight parallel logging for in-memory database management systems." Proceedings of the VLDB Endowment 14.2 (2020): 189-201.

[3] Intel® Optane™ DC Persistent Memory Quick Start Guide. https://www.intel.com/content/dam/support/us/en/documents/memory-and-storage/data-center-persistent-mem/Intel-Optane-DC-Persistent-Memory-Quick-Start-Guide.pdf. (2022-08-28 방문)

[4] Benson, Lawrence, Hendrik Makait, and Tilmann Rabl. "Viper: an efficient hybrid PMem-DRAM key-value store." Proceedings of the VLDB Endowment 14.9 (2021): 1544-1556.

[5] Tu, Stephen, et al. "Speedy transactions in multicore in-memory databases." Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles. 2013.

[6] Yu, Xiangyao, et al. "Staring into the abyss: An evaluation of concurrency control with one thousand cores." (2014).