X-FTL 을 활용한 SQLite 다중버전 동시성 제어

이종백, 오기환, 이상원
성균관대학교 전자전기컴퓨터공학과

e-mail : {hundredbag, wurikiji, swlee}@skku.edu

SQLite Multi-version Concurrency Control using X-FTL

Jong-Baeg Lee, Gi-Hwan Oh, Sang-Won Lee
Dept. of Electrical and Computer Engineering, Sungkyunkwan University

요 약

스마트 디바이스의 사용량 증가와 더불어 각종 가전기기의 스마트화로 인하여 임베디드 시스템에서 주로 사용되는 SQLite 데이터베이스에 대한 동시적 접근 제어의 중요성이 증가하였다. 플래시 메모리 저장장치 단계에서 트랜잭션의 원자성을 제공하는 X-FTL 은 SQLite 의 저널링 모드에서 발생하는 쓰기 연산으로 인한 성능 저하를 해결하였다. 또한 페이지 단위로 트랜잭션의 원자성을 관리하는 X-FTL 의 특징을 이용한다면 동시성 제어 측면의 성능 향상을 기대할 수 있다. 본 논문에서는 X-FTL 을 사용할 때 발생할 수 있는 동시성 제어 성능의 한계를 밝히고, X-FTL 의 X-L2P 테이블에 SCN 을 추가하여 SQLite 의 동시성 제어 성능을 향상할 수 있는 새로운 구조를 제안한다.

1. 서론

스마트폰, 태블릿 PC 등 모바일 기기의 사용량은 계속해서 증가하는 추세이다. 에릭슨 모빌리티 리포트에 따르면 2014 년 전 세계 스마트폰 가입자 수는 26 억 명이며, 5 년 후에는 이 수치가 56 억 명까지 증가할 것으로 전망된다. 모바일 디바이스의 사용량 증가와 더불어 자동차와 각종 가전기기의 스마트화도 빠르게 진행 중이다. 임베디드 환경에서 주로 사용되는 플래시 메모리와 SQLite DBMS 의 중요성이 증가하는 추세에서, 다양한 애플리케이션의 데이터베이스에 대한 동시적 접근 제어의 중요성 또한 매우 증가하였다.

SQLite DBMS 는 롤백 모드, WAL 모드로 나누어지는 두 가지 저널링 모드를 통해 트랜잭션의 원자성과 지속성을 제공한다. SQLite 가 롤백 모드로 동작하는 동안 데이터 변경이 발생하면 원본 데이터를 별도의 파일에 저장함으로써 원자성을 제공하며, SQLite 가 WAL 모드로 동작하는 경우에는 롤백 모드와는 반대로 변경된 내용을 별도의 파일에 저장함으로써 원자성을 제공한다.

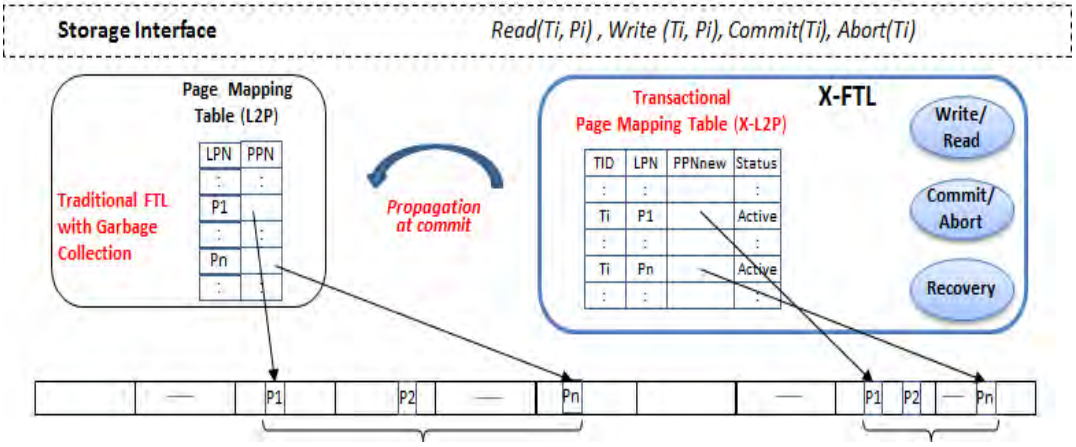
대부분의 모바일 기기는 데이터를 영구적으로 저장하기 위한 장치로 플래시 메모리를 사용한다. 동일 페이지에 대한 덮어쓰기가 불가능하다는 특성 때문에 플래시 메모리에서는 페이지 수정이 발생했을 때 기존의 페이지가 아닌 새로운 페이지에 변경된 데이터를 쓰는 Copy-on-write (COW) 전략을 이용한다.

Copy-on-write 전략을 사용하는 플래시 메모리의 특징과 원본 데이터를 유지하는 SQLite 의 저널링 모드의 유사성에서 착안한 X-FTL 은 저장장치 단계에서 트랜잭션의 원자성과 지속성을 제공한다. 이를 바탕으로 SQLite 의 성능 저하의 주된 원인으로 분석되는 저널링 모드에서의 추가적인 쓰기, 동기화 연산을 줄일 수 있다. 또한, 페이지 단위로 쓰기 연산을 관리하는 X-FTL 의 특징을 활용하면 기존의 SQLite 보다 향상된 동시성 제어를 제공할 것으로 기대된다.

이를 바탕으로 본 논문에서는 X-FTL 을 활용할 경우 SQLite 의 동시성 제어에서 발생할 수 있는 문제를 밝히고, 이를 해결하기 위한 확장된 X-FTL 의 구조에 대한 설계를 제안한다.

본 논문은 다음과 같이 구성되어 있다. 2 장에서는 본 논문의 기초가 된 SQLite 에서 제공하는 동시성 제어와 X-FTL 에 관하여 설명한다. 3 장에서는 X-FTL 을 활용할 경우 발생하는 동시성 제어 측면의 문제를 제시하고, X-FTL 의 기능을 확장하여 SQLite 에서의 동시성 제어 기능을 향상할 새로운 구조를 설계하여 제

1) 이 논문은 2012 년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(2012R1A1A2A10044300)
2) 본 연구는 지식경제부 및 한국산업기술평가관리원의 산업융합원천기술개발사업(정보통신)의 일환으로 수행하였음. [10041244, 스마트 TV 2.0 소프트웨어 플랫폼]
3) 이 논문은 2014 년도 정부(교육부)의 재원으로 한국연구재단-글로벌 박사 펠로우십사업의 지원을 받아 수행된 연구임 (2014H1A2A1019559)



(그림 1) X-FTL의 구조[1]

안한다. 마지막으로 4 장에서는 결론과 앞으로의 연구 방향을 제시하면서 본 논문을 마무리한다.

2. 관련연구

2-1. SQLite 의 동시성 제어

SQLite 는 트랜잭션의 원자성과 지속성을 제공하기 위해 데이터베이스 파일과는 별도의 저널파일을 생성하여 이용하며, 이 방법은 크게 두 가지 저널 모드로 나눌 수 있다. 각각은 롤백 모드, WAL 모드이며 SQLite 가 각 모드로 동작하는 동안 다른 방식의 동시성 제어를 제공한다.

SQLite 가 롤백 모드로 동작하는 동안 트랜잭션에 의한 변경이 발생하면 변경사항은 데이터베이스 파일에 적용되며, 원본 내용은 별도로 생성되는 롤백 저널 파일에 저장이 되어 트랜잭션의 원자성을 제공한다. SQLite 가 WAL 모드로 동작하는 경우에는 롤백모드와는 반대로 동작한다. WAL 모드에서 트랜잭션에 의한 변경이 발생하면 변경된 내용은 별도로 생성되는 WAL 파일에 적용이 되며, 원본 내용은 데이터베이스 파일에 보존하는 방법으로 트랜잭션의 원자성을 제공한다.

롤백 모드에서는 데이터베이스 파일의 락킹을 통해 동시성을 제어한다. 롤백 모드에서는 변경 사항이 데이터베이스 파일에 적용되므로 읽기 트랜잭션에서 완전하지 않은 데이터를 읽는 것을 방지하기 위해 쓰기 작업이 진행되는 동안 데이터베이스 파일에 배타적 락을 걸어 다른 트랜잭션의 동작을 막는다. 즉, 쓰기 트랜잭션에 의한 변경사항이 메모리에만 적용되고 아직 디스크에 기록되지 않았다면 이 트랜잭션은 동시에 여러 읽기 트랜잭션과 동작할 수 있지만, 변경사항을 디스크로 기록하는 동안에는 하나의 쓰기 트랜잭션만 동작한다.

WAL 모드에서는 읽기 트랜잭션과 쓰기 트랜잭션이 동시에 동작할 수 있으며, 스냅샷 격리 수준의 고립성을 제공한다. 변경사항이 WAL 파일에 기록되므로 쓰기 작업이 진행되는 동안에 원본 데이터베이스 파일을 읽음으로써 일관된 데이터를 얻을 수 있기 때문이다. 읽기 트랜잭션은 계속해서 트랜잭션 시작 시의 데이터베이스 스냅샷을 보기 때문에 쓰기 트랜잭션이 커밋 하더라도 변경된 내용을 볼 수 없다.

2-2. X-FTL

덮어쓰기를 허용하지 않는 플래시 메모리의 특징으로 인하여 FTL 에서는 데이터 변경이 발생할 때 새로운 페이지에 변경 내용을 기록하는 Copy-on-write (COW) 전략을 사용한다. 변경된 페이지와 이전 페이지를 유지하는 COW 의 특징이 SQLite 의 저널링 모드와 유사하게 동작하는 것으로부터 착안하여 저장장치 단계에서 트랜잭션의 원자성을 제공하는 X-FTL 이 제안되었다[1].

X-FTL 에서는 트랜잭션의 원자성을 제공하기 위해 기존의 페이지 매핑 테이블인 L2P 테이블을 유지하면서 추가로 트랜잭션 매핑 테이블인 X-L2P 테이블을 추가하였고, 각각은 (그림 1)의 왼쪽과 오른쪽에 나타난다. (그림 1)에서 볼 수 있듯이 X-L2P 테이블에서는 트랜잭션의 원자성을 위해 트랜잭션 ID, 새로운 페이지의 물리적 주소, 트랜잭션의 상태 정보를 저장한다.

호스트로부터 쓰기 요청이 들어오면 새로운 페이지에 쓰기 작업이 이루어진다. 기존의 FTL 에서 새로운 매핑 정보를 L2P 테이블에 적용하던 것과 달리 X-FTL 에서는 트랜잭션이 커밋 또는 롤백 될 때까지 새로운 매핑 정보를 X-L2P 테이블에서 관리한다. 이전 페이지의 물리 주소는 L2P 테이블에서 유지되며, 트랜잭션이 롤백 되어야 할 때 이전 페이지 주소를 사용하여 롤백 되기 때문에 이전 페이지는 가비지 컬렉션 되지 않는다.

여러 페이지에 대한 쓰기를 요청하는 트랜잭션의 원자성을 제공하기 위해 X-FTL 에서는 X-L2P 테이블을 이용하여 기존의 FTL 과는 다른 방식의 커밋 과정을 보여준다. 커밋 명령을 받으면 트랜잭션에 의해 변경된 페이지는 우선 플래시 메모리에 영구적으로 기록된다. 그 후 X-L2P 테이블의 트랜잭션 상태의 값을 Committed 로 변경한 뒤 변경된 X-L2P 테이블이 저장될 새로운 위치를 플래시 메타 블록에 기록한다. 이 시점에서 커밋은 완료된 것으로 판단되며, 그 후에 새로운 페이지 주소를 X-L2P 테이블에서 L2P 테

이블로 적용하는 것으로 모든 커밋 과정이 완료된다.

3. X-FTL 을 활용한 SQLite 다중 버전 동시성 제어

3-1. X-FTL 의 동시성 제어

SQLite 에서 사용되는 롤백 모드와 WAL 모드에서는 기본적으로 SERIALIZABLE 단계의 트랜잭션 고립성을 제공한다. 이를 위해 롤백모드와 WAL 모드에서는 동시에 하나의 쓰기 트랜잭션만을 허용한다. 또한, 쓰기 트랜잭션과 읽기 트랜잭션이 동시에 동작할 수 있는 WAL 모드와는 다르게, 롤백 모드에서는 쓰기 트랜잭션이 디스크에 변경된 내용을 기록하는 동안 다른 읽기 트랜잭션이 동작하는 것을 막는다.

SQLite 에서 두 가지 저널링 모드를 대신하여 X-FTL 을 사용한다면 저장소 단계에서 페이지 단위로 트랜잭션의 원자성을 관리하기 때문에 기존에 파일 단위로 동시성을 관리하던 것에 비해 향상된 동시성 제어를 기대할 수 있다. 그렇지만 기존의 X-FTL 을 그대로 사용하여 SERIALIZABLE 단계의 고립성을 보여주고자 할 경우 페이지 단위의 동시성 제어를 제공할 수 없는 문제가 발생한다.

X-FTL 환경에서 SQLite 가 동작하는 경우 기존의 저널링 모드에서 동작하는 것과 마찬가지로 여러 읽기 트랜잭션의 동시 동작은 허용된다. 그렇지만 쓰기 트랜잭션의 경우 동시에 여러 트랜잭션이 같은 페이지에 쓰기를 시도할 때, 아직 커밋되지 않은 데이터 페이지를 다른 트랜잭션이 변경할 수 있으므로 하나의 쓰기 트랜잭션만이 동작되어야 한다.

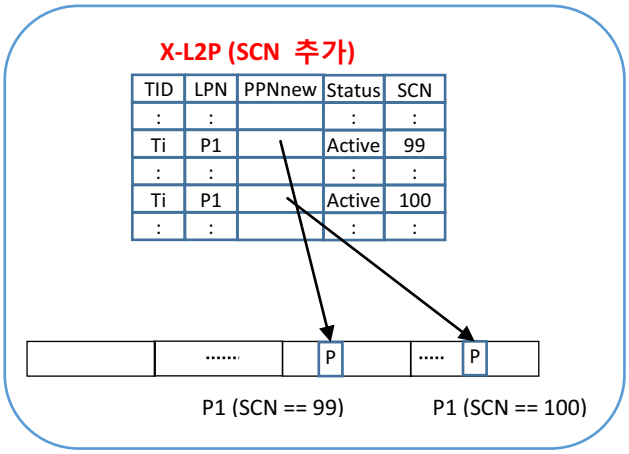
또한, X-FTL 환경에서는 쓰기 트랜잭션이 데이터를 저장장치에 기록하는 동안 읽기 트랜잭션의 동작을 막아야 한다. X-FTL 에서는 쓰기 트랜잭션이 변경을 새로운 페이지에 기록하고 그 페이지의 주소가 X-L2P 에서 관리되며, 이 때 읽기 트랜잭션은 L2P 테이블로부터 기존의 데이터를 읽어오는 방법으로 트랜잭션의 원자성을 제공한다.

그러므로 쓰기 트랜잭션이 커밋되어 X-L2P 의 변경사항이 L2P 테이블에 적용이 된다면, 쓰기 트랜잭션과 동시에 동작하던 읽기 트랜잭션의 경우 L2P 테이블을 참조하여 데이터를 읽기 때문에 동일 트랜잭션 안에서도 변경 전, 변경 후의 페이지를 읽을 수 있다. 이는 SQLite 의 SERIALIZABLE 규칙에 어긋나고, 따라서 쓰기 트랜잭션이 실제로 쓰기 작업을 하는 동안 읽기 트랜잭션이 동시에 동작하는 것을 막아야 한다. 결과적으로 X-FTL 을 사용한다면 롤백 모드와 같은 수준의 동시성 제어를 제공할 수밖에 없다.

3-2. X-FTL 을 활용한 SQLite 다중 버전 동시성 제어

기존의 X-FTL 을 사용한다면 롤백 모드와 같은 수준의 동시성 제어만을 제공할 수 있으며, 쓰기 트랜잭션이 읽기 트랜잭션을 막지 않는 WAL 모드에 비해 낮은 동시성 제어 성능을 보인다.

이를 극복하기 위해 X-L2P 테이블의 항목에 (그림 2)와 같이 SCN (System Commit Number)을 추가하여 다중 버전의 페이지를 관리한다면 동시성 제어의 성능을 높일 수 있다. SQLite 로부터의 쓰기 요청에 SCN 정보를 추가하면 X-L2P 테이블에는 (그림 2)의 P1 과 같이 여러 버전의 페이지를 관리할 수 있으며, 읽기 요청에도 SCN 을 추가하여 해당 트랜잭션에 알맞은 버전의 페이지를 읽도록 할 수 있다.



(그림 2) SCN을 추가한 X-L2P 테이블 구조

이러한 방법을 통해 기존의 X-FTL 을 사용할 때 나타나는 쓰기 트랜잭션이 읽기 트랜잭션을 막는 문제를 해결할 수 있다. SCN 이 추가된 X-FTL 에서 쓰기 트랜잭션이 새로운 페이지에 변경사항을 적용하여 X-L2P 테이블에 그 내용을 반영하더라도 읽기 트랜잭션은 SCN 을 확인하여 알맞은 페이지를 읽을 수 있기 때문이다.

이처럼 X-FTL 에 SCN 을 적용하면 쓰기 트랜잭션이 읽기 트랜잭션을 막지 않아도 SERIALIZABLE 단계의 고립성을 제공할 수 있으며, 기존의 X-FTL 에 비해 좋은 동시성 제어 성능을 보일 수 있다. 더 나아가서 페이지 단위로 변경을 관리하는 X-FTL 의 장점을 활용하여 서로 다른 페이지를 수정하는 여러 쓰기 트랜잭션이 동시에 동작할 수 있다.

4. 결론 및 향후연구

본 논문에서는 SQLite DBMS 에서 트랜잭션의 원자성을 저장장치 단계에서 제공하는 X-FTL 을 활용할 때 발생하는 동시성 제어의 한계를 밝히고, 더 나아가 X-FTL 을 확장하여 SQLite 의 동시성 제어 성능을 향상시킬 수 있는 새로운 구조를 제안하였다.

기존의 X-FTL 을 그대로 사용하면 동시에 하나의 쓰기 트랜잭션이 동작할 수 있으며, 쓰기 트랜잭션이 저장장치에 변경사항을 기록하는 동안에는 읽기 트랜잭션이 동작할 수 없는 문제점이 있다. 이는 SQLite 의 저널링 모드 중 롤백 모드와 같은 수준의 동시성 제어 성능을 보이는 것이며, 또 다른 저널링 모드인 WAL 모드에서의 동시성 제어 성능보다 떨어진다.

이를 극복하기 위해 본 논문에서는 X-FTL 의 X-L2P 테이블을 확장하여 SCN 을 기록하는 새로운 구조를 제안하였다. SCN 을 통해 다중 버전의 페이지를 관리함으로써 쓰기 트랜잭션이 저장장치에 쓰기 연산을 진행하는 동안에도 읽기 트랜잭션이 동작할 수 있으며, 서로 다른 페이지에 대한 쓰기 트랜잭션의 경

우 동시에 여러 트랜잭션이 동작할 수 있을 것으로 기대된다.

향후 연구에서는 실제로 SCN 이 추가된 X-FTL 을 구현하여 동시성 제어 측면에서의 성능을 측정할 것이다. 또한 여러 쓰기 트랜잭션이 동시에 동작할 때 발생할 수 있는 데드락 문제를 해결할 방안에 대하여 연구할 계획이다.

참고문헌

[1] Woon-Hak Kang, Sang-Won Lee, Bongki Moon, Gi-Hwan Oh, Changwoo Min, "X-FTL: Transactional FTL for SQLite Databases", SIGMOD 2013.

[2] “SQLite documentation”, http://sqlite.org/docs.html.