다중 스레드 시스템에서 비휘발성 메모리의 Redo 복구 기법

김지현[0] 원유집
한양대학교
wlgus3018@hanyang.ac.kr, yjwon@hanyang.ac.kr

Redo Recovery Technique for Non-volatile Memory in Multiple-Thread system

Jihyun Kim[0] Youjip Won
Hanyang University

요 약

비휘발성 메모리는 전원이 차단된 뒤에도 메모리 내에 데이터를 유지하는 영속적인 특징을 가진 차세대 메모리이다. 그러나 비휘발성 메모리는 비정상적인 시스템 오류 시 데이터가 메모리에 존재하기 때문에 데이터 복구에 대한 기법이 필수적이다. 본 논문에서는 비휘발성 메모리를 관리하는 대표적인 메커니즘인 "HEAPO"를 이용하여 시스템 오류 시 발생하는 데이터 손실에 대한 복구 기법인 Redo 기법을 제안한다. 또한 단일 스레드와 다중 스레드 환경에서 Redo 복구 기법의 차이점을 설명하였다.

1. 서 론

일반적인 컴퓨터 시스템은 DRAM과 같은 휘발성 메모리에 근간한 시스템으로 구성되어 있다. 기존의 DRAM은 전원이 차단되면 메모리의 데이터는 모두 사라진다. 최근 Phase-change memory (PCM) 같은 비휘발성 특성을 지닌 차세대 메모리 소자가 개발 되었다. 비휘발성 메모리는 전원이 차단되어도 데이터를 유지하며 데이터의 바이트 접근이 가능하도록 한다. 데이터를 유지하는 특성 때문에 데이터의 일관성을 보장하는 연구가 활발히 진행 중이다. 현재 HEAPO(Heap-Based PersistentObject Store)[1], Mnemosyne[2]와 같은 비휘발성 메모리를 관리하는 매커니즘들이 존재하며 각 메커니즘은 TinySTM[3] 기반의 트랜잭션 시스템을 지원한다. 또 다중스레드 환경에서 트랜잭션 의 동시성을 제어 한다. 본 논문에서는 TinySTM 기반의 HEAPO 메커니즘 을 바탕으로, Linked List의 삽입 연산 수행 시 시스템 오류가 발생 하여 비정상적인 데이터가 메모리에 존재 할 때, 해당 자료구조를 복구하는 Redo 복구 기법을 제시 하고 단일 스레드와 다중 스레드에서의 Redo 복구 기법 차이와 복구 시간을 측정하였다.

2. 설 계

상용 데이터베이스인 SQLite[4]와 같은 시스템에서 데이터 손실이 발생 했을 때 사용하는 복구 방법에는 Undo 로깅 또는 Redo 로깅 등이 있다. 비휘발성 메모리 시스템에서 갑작스러운 시스템 오류 시에 남아있는 데이터의 일관성을 보장하기 위해서 위와 같은 기법들을 사용한다. 일관성 유지를 위해 존재하는 여러 기법들 중 데이터베이스 시스템의 대표적인 복구 기법인 Redo 로깅은 갑작스러운 시스템 장애 발생 시 현재의 상태로 복구 시켜주는 기법이다. 논리적으로 한순간에 발생하는 단위를 트랜잭션이라 하며 Redo 로깅은 트랜잭션동안 발생하는 데이터의 쓰기 연산들을 '로그'에 먼저 기록한다. 단일 스레드 환경에서 Redo 복구는 간단하다. 복구 데이터가 저장된 '로그'에서 순서대로 자료를 복구 하면 된다. 하지만 다중 스레드 환경에서 Redo 복구는 단일 스레드처럼 단순하지 않다. 그 이유는 다중스레드 환경에서 각 스레드의 '로그' 데이터는 서로 의존성이 존재할 수 있기 때문이다. 각 스레드 1, 스레드 2가 동일한 메모리 주소에 값을 쓰는 레코드가 있고 스레드 1이 먼저 수행 되었다 하자. 만약 복구 시 스레드2가 먼저 복구 된다면 스레드 1의 데이터는 먼저 수행됨에도 불구하고 스레드2의 덮어쓰기에 의해 데이터의 일관성이 깨지게 된다.

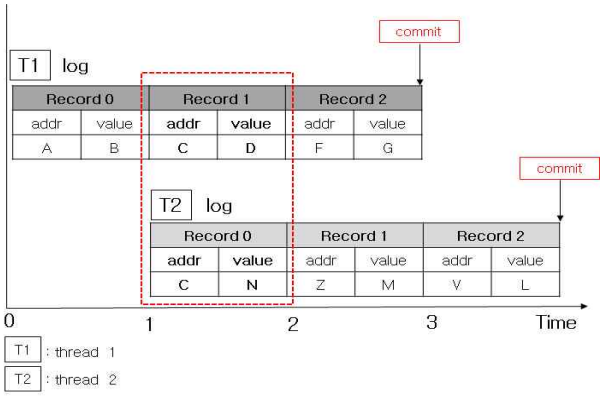

그림 1. 스레드간 데이터 의존성

일관성 있는 데이터 복구를 위한 고려사항은 두 가지다.

첫 번째 전체 트랜잭션의 레코드들이 '로그'영역에 기록되었는지 파악하는 것이다. 만약 전체 트랜잭션이 로그 영역에 기록되지 않았다면 해당 트랜잭션으로 복구 할 시 데이터의 일관성이 깨지므로 복구 할 수 없다. 전체 트랜잭션이 '로그' 영역에 존재함을 명시하기 위해 'Commit' 변수를 사용하였다. 'Commit' 변수는 트랜잭션의 마지막 레코드가 로그에 기록되는 시점에 설정된다. 복구 시 해당 변수가 설정되어 있지 않으면 해당 트랜잭션은 비정상적인 트랜잭션이므로 복구하지 않는다.

두 번째 스레드 간 '로그' 데이터의 의존성을 파악하는 것이다. 데이터 의존성을 파악하기 위해 레코드들의 기록순서를 명시하는 전역 변수인 'global-counter'를 적용하였다. 'global-counter'는 로그에 기록될 때 마다 원자적으로 값이 증가하게 된다. 복구 시에 가장 작은 global_counter 변수를 가지는 트랜잭션의 레코드부터 가장 큰 global_counter 변수를 가지는 레코드까지 데이터를 복구한다. 그림2.는 다중스레드 환경에서 Redo 복구 순서도 이다.

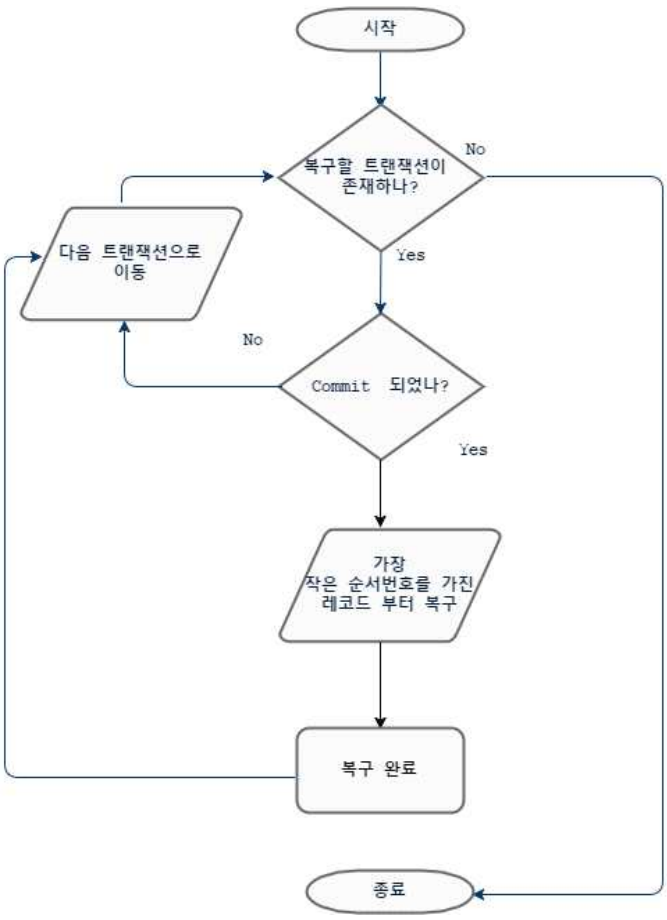

그림 2. 다중스레드 환경에서의 Redo 복구 순서도

3. 실 험

실험은 "HEAPO: TinySTM"에서 수행하였다. 워크로드는 Linked List 자료구조의 삽입연산이다. 각각 Undo 및 Redo 복구 기법의 복구시간을 측정하였다. 만개의 리스트 노드 삽입 중, 마지막 노드 삽입 중 인위적으로 프로그램을 종료하여 시스템 오류 상황을 가정하였다. 마지막 노드의 복구시간을 측정하였다. Undo시 복구 시간은 307마이크로초이고 Redo시 복구시간은 310마이크로초이다. 스레드의 개수는 2개이다. Undo/Redo의 복구속도 차이는 미비하다.

실험 환경은 Intel i7-3820@3.60Ghz, 12GB DRAM이다. 운영체제는 Ubuntu 10.04 이고 커널 버전은 Linux 2.6.38 이다.

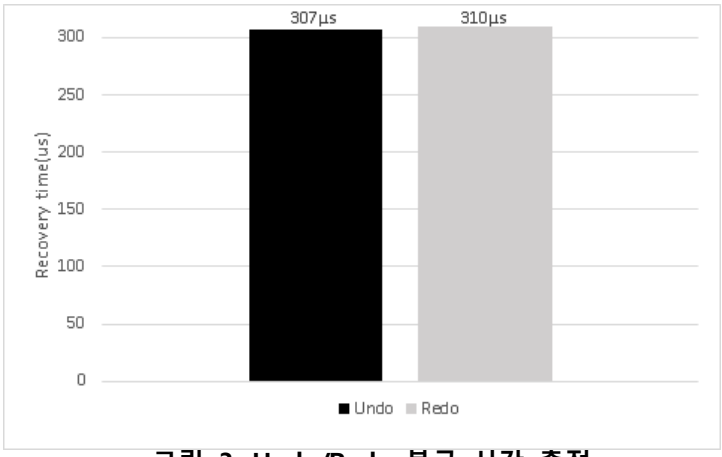

그림 3. Undo/Redo 복구 시간 측정

4. 결 론

본 논문에서는 비휘발성 메모리를 관리하는 "HEAPO : TinySTM" 의 키-값 자료구조인 Linked List를 사용하여, 비정상적인 시스템 오류 시에 발생하는 데이터 손실에 대한 Redo 복구 기법을 제안하였다. 또한 단일/다중스레드 환경에서 데이터 복구 시 차이점을 설명하고 복구 기법을 구현 하였다.

5. 사 사

본 연구는 지식경제부 및 한국산업기술평가관리원의 산업원천 기술개발사업(정보통신)의 일환으로 수행하였음. [No.10041608, 차세대 메모리 기반의 스마트 디바이스용 임베디드 시스템 소프트웨어]

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학ICT연구센터 육성지원사업의 연구결과로 수행되었음 (IITP-2016- H8501-16-1006)

참고 문헌

[1] Taeho Hwang, Jaemin Jung and Youjip Won,

"HEAPO: Heap-Based Persistent Object Store" in ACM Transactions on Storage(TOS), Volume 11 Issue 1,
[2] Haris Volos , Andres Jaan Tack , Michael M. Swift, Mnemosyne: lightweight persistent memory, Proceedingsof the sixteenth international conference on Architectural support for programming languages and operating systems February 2015.
[3] Pascal, F., Christof, F., and Torvald, R. 2008. Dynamic performance tuning of word-based software transactional memory. In Proceedings of the 13th ACM SIGPLAN Symposium on Principles and practice of parallel programming
[4] The SQLite http://www.sqlite.org/