제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003.5)

실시간 데이터베이스 시스템을 위한 확장된 동시성 제어 기법

김세윤*, 김응모*
*성균관대학교 정보통신공학부 컴퓨터공학과
e-mail : seyoonkim@ece.skku.ac.kr

An Extended concurrency control protocol for Real-Time database system

Se-Yoon Kim*, Ung-Mo Kim*
**Dept of Computer Engineering, Sung-Kyun-Kwan University

요 약

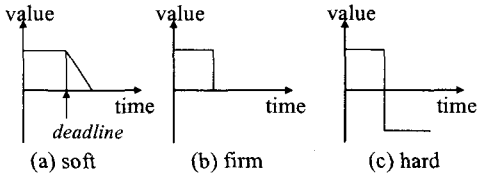
본 논문은 실시간 데이터베이스 환경에서의 동시성제어 프로토콜을 제안 그 성능을 향상시키고자 한다. 기존에 실시간 데이터베이스에서 주로 쓰이는 2PL-HP(2 Phase Locking with High Priority)의 방법은 높은 우선순위를 갖는 트랜잭션(HPT)의 선행 처리를 항상 보장하기 때문에 낮은 우선순위를 갖는 트랜잭션(LPT)의 철회 및 블로킹이 불가피하였다. 이러한 문제를 해결하고자 본 논문에서는 실시간 데이터베이스에 기존의 XAL의 기부연산을 확장하고 우선순위 개념을 도입해 LPT의 불필요한 철회 및 기다림을 방지할 수 있는 효율적인 동시성제어 프로토콜을 제안한다.

1. 서론

최근 실시간 시스템의 중요성이 증대됨에 따라 트랜잭션의 시간적 제약성과 데이터의 일치성을 보장하는 스케줄링 기법이 필요하게 되었다. 이로서 실시간적이 요소와 데이터베이스 기법을 혼용한 실시간 데이터베이스 시스템(Real-Time Database System : RTDBS)[5]이 등장하게 되었다.

실시간 데이터베이스 시스템은 기존의 데이터베이스 시스템(Conventional Database Systems : CDBS)[4]의 특성인 데이터 일관성 제약조건(data consistency constraint)에 실시간 시스템의 시간적 제약조건(timing constraints)을 첨가한 트랜잭션 처리 시스템(transaction process system)이라고 말할 수 있다. 이러한 실시간 데이터베이스 시스템이 도출한 결과는 실행의 논리적인 정확성(logical correctness) 뿐 아니라 생성된 시간에 종속 되어 있다.

실시간 트랜잭션의 시간 제약을 표현하기 위해서 가치함수(value function)를 사용한다. 시간에 따라서 결과가 결정되는 가치함수의 값은 트랜잭션을 완료한 후에 실시간 데이터베이스 시스템에게 기여한 정도를 표현한다.



(그림 1) 가치함수의 분류

실시간 데이터베이스 시스템의 트랜잭션은 (그림 1)과 같이 soft, firm, hard등으로 구분될 수 있다.

soft 실시간 트랜잭션은 마감시간(deadline)에 대한 제약조건을 완화시킴으로써 트랜잭션이 마감시간 내에 종료가 되지 않더라도 시스템의 실패는 의미하지 않는 즉, 마감시간을 지나서까지 수행이 가능한 트랜잭션이며, firm 실시간 트랜잭션은 마감시간을 지나서도 수행을 완료하지 못한 트랜잭션은 시스템에서 더 이상 수행될 수 없는 경우를 말하며, hard 실시간 트랜잭션은 마감시간 내에 수행을 완료하지 않는 경우 시스템의 실패뿐만 아니라 막대한 피해를 낼 수 있는 트랜잭션을 말한다.

실시간 데이터베이스 시스템에서 트랜잭션은 시간적 제약조건을 갖기 때문에 일반 데이터베이스 시스템과는 다른 방법으로 트랜잭션을 스케줄링해야 한다. 다시 말하면, 보다 많은 트랜잭션들이 주어진 시간적 제약조건을 만족시킬 수 있도록 스케줄링 되어야 하기 때문에 모든 트랜잭션이 공정하게 스케줄링 되지 못한다.

그러므로 실시간 데이터베이스 시스템에서는 우선순위를 사용해서 먼저 수행되어야 할 트랜잭션의 순서를 정하고 있다. 즉 실시간 데이터베이스 시스템에서는 일관성 역시 중요하나, 데이터 객체들의 값들을 마감시간 내에 계산하는 것도 중요하다. 따라서 이러한 경우 부여된 마감시간에 가까운 트랜잭션을 먼저 처리하는 스케줄링 기법이 필요로 하게 된다.

2. 관련연구

2.1 2PL-HP

2PL-HP는 기존에 데이터베이스에서 가장 일반적인 방법인 2단계 잠금 기법(Two-Phase Locking)[1]에서 발전된 방법이다. 기존의 2단계 잠금 기법의 경우 우선순위 역전(Priority Inversion) 및 교착 상태(Deadlock)가 발생하는 문제점을 갖는다. 2단계 잠금 기법은 우선순위를 전혀 고려하지 않기 때문에 실시간 데이터베이스에는 적합하지 못한 방법이다. 여기에서 우선순위 역전이란 우선순위가 높은 트랜잭션이 우선순위가 낮은 트랜잭션에 의해 블록 되는 현상을 말한다.

(그림 2)와 같은 2PL-HP 방법은 충돌 발생 시 트랜잭션들의 우선순위를 비교하여 LPT를 취소시키고, 항상 HPT가 수행되며 현재 가장 일반적인 방법이다. 그러나 이 방법에서 우선순위 할당 방법으로 최소여유시간 우선 방식(Least Slack First)을 사용하는 경우 트랜잭션의 반복적인 철회 문제가 발생할 수 있다. 이 문제를 해결하기 위해서 (그림 3)과 같이 충돌 검사 시점에서 LPT의 재시작 후의 우선 순위를 미리 계산하여 평가에 반영하도록 하는 방법이 제안됐다.

```
//p(TH)는 Holder의 우선순위
//p(TR)은 Requester의 우선순위
if p(TH) < p(TR) then {
     abort TH:
     run TR:
     }
else{
     TR blocks:
     run TH:
     }
```

(그림 2) 일반적인 2PL-HP 방법

```
//p(TaH)는 재시작시의 우선순위
if p(TH)<p(TR) and p(TaH)<p(TR)
  then {
       abort TH:
       run TR:
       }
  else{
       TR blocks:
       run TH:
       }
```

(그림 3) 반복적인 철회 해결을 고려한 2PL-HP 방법

2. 2 Extended Altruistic Locking Protocol (XAL)

확장된 이타적 잠금 기법(eXtended Altruistic Locking: XAL)[3]은 이타적 잠금 기법(Altruistic Locking)[2]을 보완한 것이며 이타적 잠금 기법은 2단계 잠금 기법(Two-Phase Locking)을 보완한 것이다.

즉 2단계 잠금 기법 -> 이타적 잠금 기법 -> XAL 위와 같이 보완되어진 것이다.

우선 이타적 잠금 기법은 기부(donate)라고 하는 새로운 동시성 제어연산을 제공한다. 해제와 마찬가지로 기부는 객체를 점유한 트랜잭션이 더 이상 그 객체를 필요로 하지 않을 때 데이터베이스 관리자에게 그 객체를 다른 트랜잭션들이 사용할 수 있도록 허락하도록 요청한다. 기부된 객체를 사용하려고 할 때 다른 트랜잭션이 사용한 후 기부한 객체들의 자취(wake)에 따라 그 트랜잭션이 기부하거나 사용했던 객체들에 대해서만 사용 요청을 할 수 있다.

이타적 잠금 기법에서의 기본적인 개념은 트랜잭션이 잠금한 객체가 더 이상 사용되지 않을 것이라고 판단이 되면 그 객체는 미리 잠금를 풀어 주어 다른 트랜잭션으로 하여금 그 객체를 사용할 수 있도록 하는 것이다. 다른 조기 해제 기법과는 달리, 이타적

잠금 기법은 트랜잭션의 직렬성을 보장하였고, 트랜잭션들이 데이터베이스를 순차적으로 처리되도록 강요하는 등의 자료의 처리 방법에 대한 제약 조건이 없다.

2단계 잠금 기법에서와 마찬가지로 이타적 잠금 기법 에서도 트랜잭션들이 동시에 동일한 객체에 대해서 로크를 할 수 없다는 규칙을 지키는 프로토콜이다. 다음은 이타적 잠금 기법에서 지켜야 하는 두가지 규칙이다.

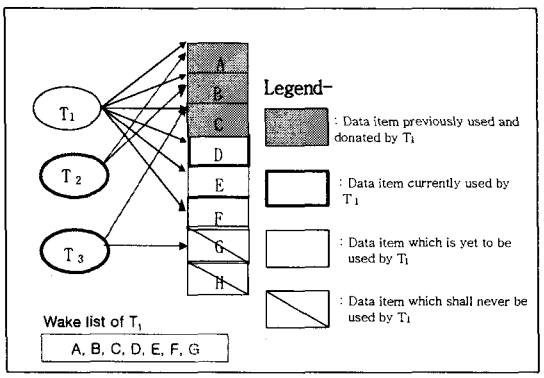
규칙1(로크의 배타성): 하나의 트랜잭션이 객체를 먼저 기부하지 않는 한 두개의 트랜잭션은 동시에 동일한 객체에 대하여 로크를 할 수 없다.

만일 트랜잭션 T_X가 다른 트랜잭션 T_Y에 의해 기부된 객체에 대하여 로크를 하였다면, Tx는 T_Y의 자취를 온전히 따라야 한다.

규칙2(완전한 자취 추적): 트랜잭션 T_X가 트랜잭션 T_Y의 자취를 따른다면 T_X는 T_Y가 로크를 처음으로 해제할 때까지 Y의 자취를 따라야 한다.

기부된 객체를 사용한 트랜잭션은 기부한 트랜잭션의 자취(wake)를 온전히 따라야 한다는 제약을 약화하여 그 자취에 속하지 않은 객체로의 요청을 가능케 하기 위해 고안된 잠금 기법이 XAL이다.

이 기법은 트랜잭션에 의해 쓰이지 않을 객체가 트랜잭션의 자취리스트(wake-list)에 포함될 수 있다는 것이다. 그러한 처리를 위해서, XAL에서는 트랜잭션이 어떠한 객체를 처리할 것인가에 대한 정보를 가지고 있어야 한다.



(그림 4) XAL의 기부에 의해 수행되는 예

(그림 4)를 보면 트랜잭션 T2는 이타적 잠금 기법의 예이다. T2는 T1의 자취리스트안의 객체만을 따랐으며 그 외의 객체에 접근은 하지 않았다.

하지만 XAL에서는 트랜잭션 T2뿐 아니라 T3처럼 자취리스트에 없는 객체도 참조하면서 그 객체 마져도 T1의 자취리스트에 포함함으로서 이타적 잠금 기법을 보다 확장했다고 볼 수 있다.

3. 제안하는 동시성 제어방법

기존의 데이터베이스 시스템에서 일반적으로 사용하고 있는 2단계 잠금 기법이나 관련연구에서 제시한 XAL의 경우 기존의 데이터베이스 시스템에서 적합한 동시성 제어 기법들이다 하지만 실시간 데이터베이스 시스템의 경우 서론에서 언급한 바와 같이 트랜잭션은 마감시간 내에 계산을 완료하는 것이 매우 중요하며 이를 지키기 위해서 우선순위 기법의 적용이 필요하다. 위의 2PH-HP의 방법은 실시간 데이터베이스 스케쥴링 방법으로 성능이 우수한 스케쥴링 기법이다. 그러나 이러한 방법은 높은 우선순위를 갖는 트랜잭션(HPT)의 선행 처리를 항상 보장하기 때문에 낮은 우선순위를 갖는 트랜잭션(LPT)의 철회 및 블로킹이 불가피하였다. 본 논문은 이러한 철회와 블로킹으로 인한 기다림을 최소화해 실시간 데이터베이스 시스템의 성능 향상을 목표로 출발하였다.

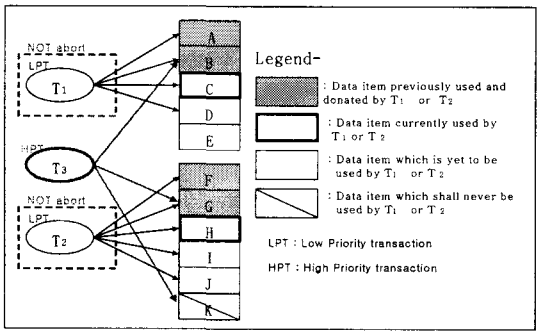
3.1 실시간 데이터베이스에서의 XAL (RTXAL)

Real-Time Extended Altruistic Locking(RTXAL)은 실시간 데이터베이스 시스템 환경에 맞게 XAL을 개선 확장하여 적용시킨 프로토콜이다. 기존의 XAL의 경우 실시간 데이터베이스 시스템 환경에 적용시키는데 크게 2가지의 문제점을 가지고 있다.

첫 번째로 두개의 트랜잭션으로부터 기부를 받을 수 없다. (그림 5)에서 트랜잭션 T1, T2, T3는 순차적으로 발생한다. T1은 A와 B를 성공리에 사용한 후 기부하였다. T1은 객체 C에 갱신 연산을 하고 있는 중이다. T2는 F와 G의 실행을 성공리에 마치고 기부하였고 H에 갱신 연산을 하고 있다. 기존의 XAL의 경우 (그림 5)의 T3 같이 T1에서 기부한 B와 T2에서 기부한 G를 사용할 수 없다. 그것이 XAL의 한계인데 RTXAL에서는 T1과 T2의 자취리스트에 T3의 객체만 포함시키는 것이 아니라 T3가 기부 받은 대상의 자취리스트도 포함시켜 (그림 5)과 같이 T3가 여러 트랜잭션이 기부한 객체를 사용할 수 있게 하였다.

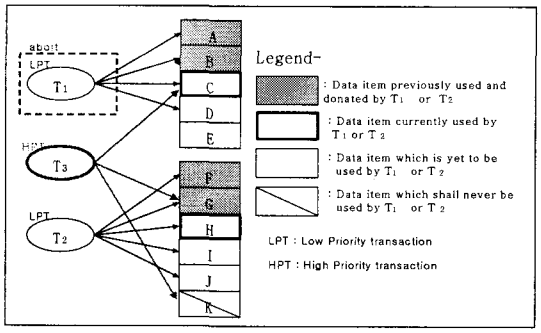
두 번째로 기존의 XAL의 경우 실시간 환경을 고려하여 만들어진 프로토콜이 아니기 때문에 마감시간을 고려하지 않고 있다. 이를 보안하기 위해서 RTXAL에서는 우선순위 기법을 적용하여 그 문제

를 해결하였다.



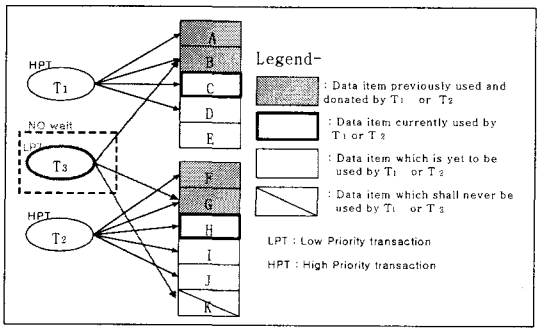
(그림 5) RTXAL의 예제 1

(그림 5)의 경우 T1, T2, T3의 순으로 트랜잭션이 실행되고 T1과 T2 보다 T3의 우선순위가 높은 경우 기존의 2PL-HP에서는 낮은 우선순위를 가지는 T1, T2는 모두 철회되고 높은 우선순위를 가지는 T3가 실행된 후 T1, T2가 재 시작 되어야만 한다. 하지만 RTXAL의 경우에는 철회시키지 않고도 높은 우선순위인 T3의 트랜잭션이 진행될 수 있다.



(그림 6) RTXAL의 예제 2

(그림 6)의 경우에는 높은 우선순위인 T3가 낮은 우선순위인 T1의 기부되지 않은 C 객체를 요구할 경우 T1은 철회 되지만 낮은 우선순위인 T2의 경우에는 기부된 객체 G를 T3에서 요구하기 때문에 철회를 하지 않고도 계속 진행할 수 있다.



(그림 7) RTXAL의 예제 3

(그림 7)의 경우 T1, T2, T3의 순으로 트랜잭션이 실행되고 T1과 T2가 T3보다 우선순위가 낮은 경우 기존의 2PL-HP에서는 낮은 우선순위를 가지는 T3는 높은 우선순위를 가지는 T1과 T2가 모두 종료할 때 까지 기다려야 하고 T1, T2가 모두 종료 후에 실행할 수 있다. 하지만 RTXAL의 경우에는 T3는 낮은 우선순위를 가지는 트랜잭션 임에도 높은 우선순위를 가지는 T1, T2의 기부된 객체만을 요구하기 때문에 기다리지 않고도 T3가 진행될 수 있다.

4. 결론

본 논문에서는 soft 실시간 데이터베이스에 RTXAL을 적용 효율적인 동시성 제어 방법을 제안하였다. 객체를 점유한 트랜잭션이 더 이상 그 객체를 필요로 하지 않을 때 그 객체를 기부함으로 우선순위가 낮은 트랜잭션들의 불필요한 재 시작 및 기다림을 방지함으로써 시스템의 처리율을 향상시킬 수 있었다. 향후, 제안된 RTXAL의 성능 평가 및 firm과 hard 실시간 데이터베이스 시스템에서의 적용을 연구 중에 있다.

참고문헌

[1] Meichun Hsu, Arvola Chan "Partitioned two-phase locking" ACM Transactions on Database Systems (TODS) December 1986

[2] Kenneth Salem, Hector Garcia-Molina, Jeannie Shands "Altruistic Locking" ACM Transactions on Database Systems, Vol. 19, No 1, March, Pages 117-165.

[3] K. Yu, "Extended Altruistic Locking Favoring Short-Lived Transactions in Database Management Systems: XAL/S," M.S. Thesis, Dept. of Techno-management, KAIST, Korea, 1996

[4] N. S. Garghouti and G. E. Keiser, "Concurrency Control in Advanced Database Applications," ACM Computing Surverys, Vol. 23, No. 3, ppo. 269-317, Sep. 1991.

[5] Tei-Wei Kuo, Ming-Chung Liang, LihChyun Shu "Abort-Oriented Concurrency Control for Real-Time Databases," IEEE Transactions on Computers, Vol. 50, No. 7, July, 2001.