실시간 능동 트랜잭션 처리를 위한 동시성 제어 기법

홍석희*

Concurrency Control for Processing Real-Time Active Transactions

Seok-Hee Hong*

이 논문은 2000학년도 경성대학교 학술지원연구비에 의하여 지원되었음

요 약

실시간 능동 데이터베이스 시스템은 외부의 환경의 변화에 반응하여 트랜잭션들을 주어진 시간 내에 처리할 수 있어야 한다. 본 논문에서는 Thomas 쓰기 규칙을 적용하여 보다 향상된 실시간 능동 동시성 제어 기법을 제안한다. 제안하는 동시성 제어 기법은 데이터 충돌을 해결하기 위해서 우선순위와 능동 규칙을 고려하여 불필요하게 철회되는 트랜잭션의 수를 감소시킬 수 있다. 본 논문에서는 모의실험을 통해서 제안하는 동시성 제어기법에 의해서 보다 많은 트랜잭션이 종료시한을 만족함을 보인다.

ABSTRACT

Transactions in real-time active databases have the notion of activeness where transactions are generated by external effects and another transaction. In this paper, we improve the real-time active concurrency control algorithm by applying Thomas' write rule and considering relationship between transactions fired by active rules. We also present the experimental results of our algorithm comparing other real-time active concurrency control algorithms. The experimental results show that our algorithm has superior performance with respect to the raio of transactions satisfying deadline.

키워드

실시간 능동 데이터베이스, 실시간 동시성 제어 기법

I. 서 론

실시간 능동 데이터베이스 시스템(RTADBS : Real-Time Active Database System)은 최근에 다양한 응용 분야에 활용되면서 많은 연구가 진행되고 있다.

RTADBS는 실시간 데이터베이스 시스템과 능동 데이터베이스 시스템의 특성들을 함께 지원한다. 실시간 시스템이 사용되는 환경은 크게 통제 시스템

* 경성대학교 정보과학부

접수일자 : 2002. 4. 11

(controlling system)과 통제대상 시스템(controlled system)으로 구성된다. 통제대상 시스템은 주로 외부 환경을 감지하고 결과를 시스템에 전송하는 작업을 하며 통제 시스템은 외부로부터 전송된 데이터를 활용하여 적절한 작업을 한다. RTADBS는 실시간 시스템에서 통제 시스템으로 사용되어 대량의 데이터를 ECA (Event-Cond ition-Action) 규칙에 근거하여 종료시한(deadline)을 갖는 작업을 실행한다[6,9]. 외부 환경의 변화에 의해서 ECA 규칙이 발생하고 통제 시스템은 적절한 작업을 실시간 트랜잭션의 형태로 실행하게 된다.

실시간 데이터베이스 시스템에 대한 연구는 주로 트랜잭션 처리를 위한 스케줄링 기법과 동시성 제어 기법에 대한 연구에 집중되었고 능동 데이터베이스 시스템에 대한 연구는 ECA 규칙의 정의 및 효율적 처리 기법에 대한 연구가 많이 이루어졌다. 실시간 능동 데이터베이스 환경에서 ECA 규칙을 기반으로 트랜잭션 모델을 정립한 연구가 있었다[6, 7]. 특히, [7]의 논문에서는 ECA 규칙에 의해서 실행되는 트랜잭션들 사이의 관계를 기반으로 우선순위(priority)를 할당하는 기법을 제시하고 성능을 분석하였다. 그 외에 RTADBS를 분산 환경에 적용하여 분산 실시간 능동 트랜잭션을 위한 처리 기법을 연구한 논문이 있다[5]. RTADBS에 대한 대부분의 연구는 트랜잭션 모델링, RCA 규칙의 정의와 실행, 프로토타입 시스템의 구현, 버퍼 관리 등에 많이 이루어졌다. 그러나, 실시간 능동 트랜잭션을 처리하기 위해서 중요한 문제점중 하나인 동시성 제어에 대한 연구는 많이 진행되지 않았다. 본 논문에서는 RCA 규칙에 의해서 생성된 triggering 트리 구조를 분석하여 좀 더 효율적인 동시성 제어 기법을 제안하고자 한다. RTA-MVPR(Multi version concurrenc y control with precedence rela tionship for real-ti me active transactions)은 triggering 트랜잭션과 triggered 트랜잭션들 사이의 관계를 기반으로 하는 실시간 다중 버전 동시성 제어 기법이다. ECA 규칙에 의해서 실행되는 트랜잭션을 triggered 트랜잭션이라 하고 triggered 트랜잭션을 실행시킨 트랜잭션을 triggering 트랜잭션이라고 한다. 이 두 트랜잭션들 사이에는 triggering 관계가 성립하고 종료 종속성과 철회 종속성 등과 같은 특성이 나타난다[3,6,7]. RTA-MVPR은 이와 같은 종속성외에 트랜잭션들 사이의 직렬화가능(serializable) 순서를 고려하여 데이터 충돌을 해결한다[1]. 본 논문은 Thomas의 쓰기 규칙(write rule)을 적용하여 트랜잭션들 사이의 동시성을 증가시키도록 기존의 RTADBS를 위한 동시성 제어 기법인 RTA-MVPR[2]을 확장한다.

본 논문의 구성은 2장에서 기존에 연구되었던 실시간 능동 트랜잭션 처리 기법에 대해서 알아보고, 3장에서는 RTA-MVPR의 문제점을 제시하고, 4장에서는 실시간 능동 트랜잭션을 위한 개선된 동시성 제어 기법을 제안한다. 5장에서는 제안한 동시성 제어 기법을 다른 기법과 비교함으로서 성능평가를 한다. 마지막으로 6장에서 결론과 앞으로의 연구방향에 대해서 알아본다.

II. 관련 연구

기존의 실시간 트랜잭션을 위한 동시성 제어 기법을 기반으로 실시간 능동 동시성 제어 기법을 제안한 연구가 있었다[3]. [3]의 연구에서는 기존의 실시간 낙관적 동시성제어기법인 OCCL을 기반으로 한 OCC-APFS(OCC Adaptative Priority Fan-out Sum)이라는 동시성 제어 기법을 고안하였다. OCC-APFS는 데이터 충돌을 해결하기 위해서 트랜잭션들 사이의 능동성 관계와 우선순위를 고려하였다. 트랜잭션의 철회 가능성을 CPI(Concurrency Priority Index)라는 척도로 표현하여 데이터 충돌을 해결하기 위한 중요한 요소로 활용한다. CPI는 능동성 관계에 포함된 트랜잭션들의 수와 우선순위 등을 포함하기 때문에 트랜잭션들이 불필요하게 철회되거나 시스템 자원을 낭비하지 않도록 능동성 관계를 고려하여 데이터 충돌을 해결하도록 한다. OCC-APFS는 데이터 충돌을 해결하기 위해서 서로 능동성 관계를 가지고 있지 않은 트랜잭션들 사이의 CPI를 사용하였다. 그러나, 능동성 관계를 가지고 있는 트랜잭션들 사이의 데이터 충돌을 해결하는 경우 낮은 우선순위의 트랜잭션을 불필요하게 철회시킬 수 있다.

실시간 트랜잭션을 위한 동시성 제어 기법으로 제안된 MVPR(Multiversion with Precedence Relationship)은 다중버전 2 PL(Phase Locking)을 실시간 데이터베이스 환경에 적용시킨 기법이다[1]. 트랜잭션들 사이의 직렬화가능 순서를 트랜잭션들 사이의 선행관계(precedence relationship)로 표현하여 우선순위를

기반으로 데이터 충돌을 해결한다. MVPR은 읽기, 쓰기, 검증 등의 세 가지 잠금 모드를 사용한다. 읽기 잠금은 데이터베이스로부터 데이터를 읽기 위해서 사용하며 쓰기 잠금은 지역 공간에 읽어드린 데이터를 변경시킬 때 사용한다. 트랜잭션이 종료하는 시점에 소유하고 있는 모든 쓰기 잠금은 검증 잠금으로 변환되어야 한다. 검증 잠금을 소유한 후에는 해당 데이터를 실제로 데이터베이스에 반영할 수 있게 된다. MVPR을 실시간 능동 데이터베이스 환경에 적용시킨 기법이 RTA-MVPR이다[2].

III. RTA-MVPR의 문제점

3.1 Thomas 쓰기 규칙의 적용

트랜잭션들이 동시에 같은 데이터를 사용하려고 하는 경우 충돌이 발생할 수 있다. 충돌이 발생한 트랜잭션들 사이에는 수행 순서가 결정될 수 있으며 데이터베이스 시스템은 트랜잭션들 사이의 수행 순서가 순환관계가 되지 않도록 스케줄 해야 한다. Thomas 쓰기 규칙은 쓰기 연산들 사이의 충돌을 유연하게 해결하기 위해서 많이 적용된다[1]. [예 1]에서 RTA-MVPR에서 Thomas 쓰기 규칙이 어떻게 적용될 수 있는지 알아보자. 그림 1은 예제로 사용할 triggering 트리의 구조를 보여준다. 트랜잭션 T_1은 T_2, T_3, T_4, T_5, T_6, T_7 등 총 6개의 트랜잭션을 ECA 규칙에 의해서 실행시켰다.

[예 1] 그림 1과 같은 triggering 트리의 구조에 대해서 두 트랜잭션 T_2와 T_9 사이에 트랜잭션 T_2가 높은 우선순위를 가진다고 하자. 그리고 다음과 같이 각 트랜잭션들이 실행된다고 하자. R, W, C는 각각 읽기, 쓰기, 검증 잠금을 의미한다.

$T_2 = R_2[a];R_2[b];W_2[b];C_2[b]$
$T_9 = W_9[a];W_9[b];C_9[b];C_9[a]$
$H = R_2[a];W_9[a];R_2[b];W_2[b];W_9[b];C_9[b];C_2[b]$

수행순서 H는 두 트랜잭션이 실제로 잠금을 요청한 순서를 나타낸다. 수행순서에 의하면 트랜잭션 T_2가 데이터 a에 대해서 읽기 잠금을 소유한 후 T_9이 같은 데이터에 대해서 쓰기 잠금을 요청했으므로 두 트랜잭션 T_2와 T_9 사이에는 T_2가 T_9을 수행순서에서 선행하는 $T_2 < T_9$의 직렬화가능 순서가 성립함을 알 수 있다. 트랜잭션 T_9이 검증 잠금을 데이터 b에 대해서 소유한 후에 T_2가 같은 데이터에 대해서 검증 잠금을 요청하는 상황을 보자. T_9의 검증 잠금이 T_2의 검증 잠금 보다 선행되어 소유되었기 때문에 $T_2 < T_9$의 직렬화가능 순서를 위반한다. 따라서, RTA-MVPR에서는 우선순위가 낮은 T_9을 철회시킨다. 그러나, 이 경우 Thomas 쓰기 규칙을 적용한다면 단순히 트랜잭션 T_2의 검증 잠금을 무시하여 계속 수행을 하면 된다.

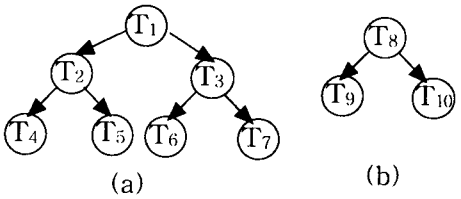


그림 1. triggering 트리의 예
Fig. 1 Example of triggering tree

3.2 Triggering 트리 구조의 활용

능동 데이터베이스 시스템의 트랜잭션들은 ECA 규칙의 실행에 의해서 triggering 트랜잭션과 triggered 트랜잭션의 관계가 성립할 수 있다. 이를 triggering 관계라 하고 triggering 관계에 의해서 종료 종속성과 철회 종속성이 성립하게 된다. 그림 1에서 만일 두 트랜잭션 T_1과 T_8이 데이터 충돌을 발생시켜서 트랜잭션 T_1이 철회되어야 한다면 T_1이 실행시킨 6개의 트랜잭션들도 연쇄적으로 철회되어야 한다. 따라서, 실시간 능동 데이터베이스 시스템에서 트랜잭션들 사이의 동시성 제어를 위해서는 트랜잭션들 사이의 우선순위와 triggering 관계를 고려해야 한다. [예 2]는 triggering 트리 구조를 이용하여 데이터 충돌을 해결하는 예제이다.

[예 2] 그림 1의 트리 구조에서 트랜잭션 T_1의 우선순위는 2이고 T_2에서 T_7 까지의 6개 트랜잭션은 우선순위가 1이라고 하자. 또한, 트랜잭션 T_8, T_9, T_{10} 등은 3의 우선순위를 가진다고 하자. T_1과 T_8이 데이터 충돌을 발생시켜서 두 트랜잭션들 중 하나를 철회시켜야 한다고 하자. RTA-MVPR에서는 T_1이 실행시킨 트랜잭션들

의 우선순위의 합을 GP(Group Priority)로 정의하여 GP(T_1)을 8로 결정하고, GP(T_8)은 9로 결정한다. 두 트랜잭션 T_1과 T_8의 GP 값에 따라서 GP 값이 작은 T_1을 철회시킨다. 따라서, T_1이 실행시킨 6개의 트랜잭션들도 철회시킨다. 그러나, 이 경우 T_1을 포함해서 7개의 트랜잭션이 했던 작업이 상실되므로 시스템 자원을 낭비한 결과가 된다. 단순히, 트랜잭션의 우선순위의 합만을 고려하지 않고 만일 triggering 트리에 포함된 트랜잭션의 수도 고려한다면 T_1 보다는 T_8을 철회시키는 것이 유리할 것이다.

IV. 실시간 능동 트랜잭션 처리 기법

4.1 동시성 제어 기법의 흐름

RTA -MVPR은 다중버전 잠금 기법을 기반으로 하기 때문에 트랜잭션 T가 데이터 D에 대해서 임의의 연산을 하기 전에 연관된 잠금 L을 요청해야 한다. 만일, 잠금 요청이 허용되면 이 트랜잭션은 데이터 D에 대해서 잠금 L을 소유한 상태가 되고 해당 연산을 할 수 있다. 데이터 D에 대해서 다른 트랜잭션과 데이터 충돌이 발생한다면 적절한 충돌 해결 기법을 적용해야 한다.

요청자 (T_R) / (T_O) 소유자	$T_O \ll T_R$			$T_R \ll T_O$		
	읽기	쓰기	검증	읽기	쓰기	검증
읽기 (Read)	Y	Y	T_R 대기	Y	T_O 철회	T_O 철회
쓰기 (Write)	T_R 대기	Y	T_R 대기	Y	Y	Y
검증 (Certify)	Y	Y	Y*	T_O 철회	T_O 철회	Y+

표 1. intra triggering 잠금 호환성 표
Table 1. Lock compatibility for resolving intra triggering conflicts

이후로 본 논문에서 제안한 실시간 능동 동시성 제어기법은 RTA-MVTWR(Real-Time Active Multiversion using Thomas' Write Rule)로 명시한다. RTA-MV WTR은 데이터 충돌을 두 가지 종류로 구분하여 해결한다. 먼저, intra triggering 충돌은 같은 triggering 트리 내에 있는 트랜잭션들 사이의 데이터 충돌을 나타내며 inter triggering 충돌은 서로 다른 triggering 트리에 포함된 트랜잭션들 사이의 데이터 충돌을 나타낸다.

4.2 intra triggering 충돌 해결 기법

표 1은 잠금을 요청하는 트랜잭션 T_R과 잠금을 소유한 트랜잭션 T_O 사이의 잠금 호환성 표를 나타내며, T_1이 T_2를 실행시키는 triggering 관계는 $T_1 \ll T_2$로 표시한다. intra 잠금 호환성 표는 잠금을 요청한 트랜잭션과 잠금을 소유하고 있는 트랜잭션들 사이의 trigg ering 관계에 의해서 크게 두 부분으로 구성된다. 표 1에서 $T_O \ll T_R$의 triggering 관계일 때 잠금을 요청하는 트랜잭션 T_R은 triggering 트랜잭션에 해당하는 T_O가 이미 잠금을 소유 중이므로 대부분의 경우 그대로 잠금이 허용되거나 수행을 대기하게 됨을 알 수 있다. 이와는 반대로 $T_R \ll T_O$의 경우 잠금을 요청하는 트랜잭션 T_R가 잠금을 소유한 트랜잭션 T_O를 실행시켰기 때문에 직렬화가능 순서에서 $T_R < T_O$의 관계가 발생한다. 결과적으로 직렬화가능 순서의 역순으로 T_R가 잠금을 요청하기 때문에 직렬화가능 순서를 유지시키기 위해서 T_R는 철회될 확률이 높게된다.

TO가 T_R을 실행하는 경우 두 트랜잭션 사이에는 TO < T_R의 선행관계가 성립한다. 이 상황에서 TO이 검증 잠금을 소유하고 있고 T_R가 검증 잠금을 요청하는 경우 Thomas 쓰기 규칙에 의해서 수행순서가 앞서는 TR의 검증 연산을 중지시키고 TO의 검증 잠금을 허용한다(Y*의 경우). 반대로, TR ≪ TO의 trigg ering 관계가 성립하고 TR이 검증 잠금을 소유하고 있고 TO가 검증 잠금을 요청한다고 하자. 이 경우도 Thomas 쓰기 규칙에 의해서 선행관계에서 뒤에 오는 TO가 먼저 검증 연산을 했기 때문에 선행관계에서 앞서는 TR의 검증 연산은 할 필요가 없다. 따라서, TR의 검증 잠금 요청은 단순히 무시하면 된다(Y+의 경우).

4.3 inter triggering 충돌 해결 기법

제안하는 RTA MVTWR에서는 triggering 트리의 구조를 고려하여 WP(Weighted Priority)라는 값을 사용하여 데이터 충돌을 해결한다. 다음은 WP (T)의 정의를 나타낸다.

$$CET(T) = \frac{|T_{Triggered}| - |T_{Uncomm}|}{|T_{Triggered}|} \times 100(\%)$$

N은 트랜잭션 T를 포함하여 triggering 트리 내 트랜잭션들의 수를, Priority(i)는 트랜잭션 i의 우선순위를 의미한다. WP(T)는 triggering 트리에 포함된 트랜잭션의 합과 트랜잭션들의 수를 더한 값으로 우선순위와 triggering 트리의 노드 수를 고려한다. 또한, α(0 $\leq \alpha \leq$1)는 WP의 결과 값에 우선순위의 합과 트랜잭션의 수에 대한 가중치를 나타낸다. 트랜잭션의 수행 상태를 고려하기 위해서 RTA-MVWTR에서는 trigg ering 트리 구조에 대해서 각 트랜잭션의 종료 시점에 근접한 정도를 나타내는 CET(Commitment Extent of Trigg ering tree)라는 평가 척도를 사용한다. 트랜잭션 T에 대한 CET는 다음과 같이 정의된다. 이 식에서 | TTriggered | 는 트랜잭션 T의 하위 triggering 트리의 모든 트랜잭션들의 수를 의미한다. | TUncomm | 은 트랜잭션 T의 하위 triggering 트리의 모든 트랜잭션들 중에서 아직 종료단계에 있지 않은 트랜잭션들의 수를 의미한다. 결국, CET(T)는 트랜잭션 T가 실행시킨 트랜잭션들 중에서 종료단계에 있는 트랜잭션의 비율을 의미한다. RTA-MVWTR에서 CET(T)는 트랜잭션 T가 종료단계에 있는 얼마나 많은 트랜잭션을 실행시켰는가를 평가한다. RTA-MVWTR은 CET의 값에 대한 임계값인 Ratedc 값을 사용하여 Ratedc 보다 CET(T)가 크다면 트랜잭션 T가 종료단계에 근접한 것으로 여기며 이와 같은 상황을 트랜잭션 T가 DC(Dominant Commit) 상태에 있다고 한다. Ratedc 는 시스템에서 적절하게 변경할 수 있는 값으로 RTA-MVWTR 스케줄러의 전체 성능에 영향을 줄 수 있다. Ratedc 값이 높을수록 보다 많은 수의 트랜잭션들이 철회될 것이다.

요청자 / 소유자		$T_{Tlwp} < T_{Thwp}$			$T_{Thwp} < T_{Tlwp}$		
		읽기	쓰기	검증	읽기	쓰기	검증
읽기 (Read)	NOT DC	Y	소유자 철회	소유자 철회	Y	소유자 철회	소유자 철회
	DC	Y	요청자 대기	요청자 대기	Y	소유자 철회	소유자 철회
쓰기 (Write)	NOT DC	소유자 철회	Y	소유자 철회	Y	Y	Y
	DC	요청자 대기	Y	요청자 대기	Y	Y	Y
검증 (Certify)	NOT DC	소유자 철회	Y	Y*	Y	Y	Y
	DC	요청자 대기	Y	Y*	Y	Y	Y

표 2. THwp가 TLwp가 소유한 잠금을 요청했을 때 inter triggering 잠금 호환성표
Table 2. Inter triggering lock compatibility matrix when THwp requests a lock held by TLwp

요청자 / 소유자		$T_{Tlwp} < T_{Thwp}$			$T_{Thwp} < T_{Tlwp}$		
		읽기	쓰기	검증	읽기	쓰기	검증
읽기 (Read)	NOT DC	Y	요청자 철회	요청자 철회	Y	요청자 철회	요청자 철회
	DC	Y	요청자 철회	요청자 철회	Y	요청자 대기	요청자 대기
쓰기 (Write)	NOT DC	Y	Y	Y	요청자 대기	Y	요청자 대기
	DC	Y	Y	Y	요청자 대기	Y	요청자 대기
검증 (Certify)	NOT DC	Y	요청자 철회	Y	요청자 대기	Y	Y*
	DC	Y	요청자 철회	Y	요청자 대기	Y	Y*

표 3. TLwp가 THwp가 소유한 잠금을 요청했을 때 inter triggering 잠금 호환성표
Table 3. Inter triggering lock compatibility matrix when TLwp requests a lock held by THwp

높은 그룹 우선순위를 가지는 트랜잭션을 THwp라 하고 낮은 그룹 우선순위를 가지는 트랜잭션을 TLwp라 하자. 표 2는 TLwp가 소유한 잠금에 대해서 THwp가 잠금을 요청한 경우 inter triggering 충돌을 해결하기 위한 잠금 호환성 표이다. 또한, 표 3은 TLwp가 소유한 잠금에 대해서 THwp가 잠금을 요청한 경우 inter triggering 충돌을 해결하기 위한 잠금 호환성 표이다. inter triggering 잠금 호환성 표는 Thomas 쓰기 규칙에 대한 고려를 하지 않았다는 점을 제외하고는 RTA-MVPR의 잠금 호환성 표와 거의 동일하다. 표 2.에서 소유자와 요청자가 서로 검증 잠금에 대해서 데이터 충돌을 발생시킨 경우 Thomas 쓰기 규칙에 의해서 잠금을 요청한 트랜잭션 TLwp는 직렬화가능 순서에서 뒤에 오기 때문에 검증 연산을 할 필요가 없다. 따라서 트랜잭션 TLwp는 단순히 검증 잠금을 무시하면 된다(Y+의 경우). 또한, 표 3에서 같은 상황에서 잠금 소유자인 THwp의 검증 연산은 더 이상 불필요하기 때문에 TLwp의 검증 연산을 곧바로 허용한다(Y*의 경우). 결과적으로 Thomas 쓰기 규칙을 적용한다면 불필요하게 철회되거나 대기하는

상황을 방지할 수 있다.

4.4 RTA-MVPR의 정확성

RTA-MVTWR은 기존에 제안된 RTA-MVPR을 기반으로 하는 알고리즘이기 때문에 선행관계에 의한 트랜잭션들 사이의 직렬화가능 스케줄의 증명은 피하기로 한다. RTA-MVPR에 대한 보다 자세한 증명은 [1, 2]에서 참조할 수 있다. RTA-MVTWR에서 triggering 관계는 선행관계와 밀접한 연관성을 가진다. 예를 들어, 두 트랜잭션 T1과 T2가 T1 《 T2의 triggering 관계를 가진다면 T1과 T2 사이에는 T1 < T2의 선행관계가 항상 성립한다. triggering 관계가 트리 구조로 표현되기 때문에 트랜잭션들 사이의 triggering 관계는 순환적인 형태를 가질 수 없다. 즉, triggering 관계를 기반으로 하는 T1 < T2와 T2 < T1이 동시에 만족되는 직렬화가능 스케줄을 위반하는 경우는 발생하지 않게 된다. 따라서, triggering 관계를 가지고 있는 트랜잭션들 사이에는 직렬화가능 스케줄을 항상 만족시킬 수 있다.

V. 성능 평가

본 장에서는 제안하는 동시성 제어기법과 다른 실시간 동시성 제어 알고리즘을 비교함으로서 성능 평가 결과를 기술하고자 한다. 성능 평가 프로그램은 쓰레드(thread)를 기반으로 하는 모의실험(simulation) 도구인 C++SIM[18]을 사용하여 수행한다. 성능 평가에 사용되는 실시간 능동 데이터베이스 시스템은 데이터베이스가 디스크 상에 저장되어있다고 가정하며 하나의 프로세서를 기반으로 운영된다. 시스템에 도착하는 트랜잭션들 간의 시간은 지수(exponential) 분포를 따르며, 각 트랜잭션은 트랜잭션의 크기, 종료시한, 우선순위, 도착시간 등의 속성을 가지고 수행된다. 임의의 트랜잭션은 일련의 읽기 또는 쓰기 연산으로 구성되며 성능 평가를 단순화시키기 위해서 버퍼링은 고려하지 않는다.

표 4. 매개 변수와 의미
Table 4. Simulation parameters

매개 변수	의미
DBsize	데이터베이스의 크기 (페이지의 수) : 4,000개 페이지
NumTrans	수행될 총 트랜잭션의 수
WriteProb	쓰기 연산의 비율 (%) : 20%
DiskI/O	디스크 I/O 시간 (ms) : 25ms
CPU	하나의 데이터 페이지에 대한 CPU 수행시간 (ms) : 8ms
TranSize	트랜잭션의 크기
ArrRate	트랜잭션 도착 비율 (trans/sec)
SlackFactor	종료시한의 여유 정도
TriggerProb	능동규칙의 실행 비율 (%) : 70%

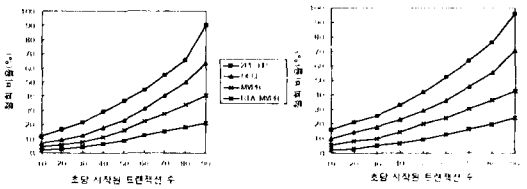


그림 2. 트랜잭션 수에 따른 트랜잭션 철회 비율
Fig. 2 the ratio of transaction aborts

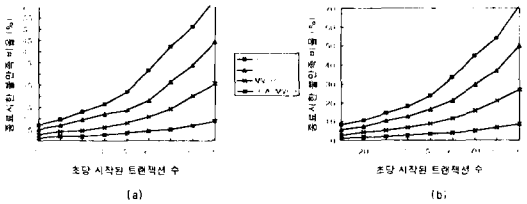


그림 3. 트랜잭션 수에 따른 종료시한 불만족 비율
Fig. 3 the ratio of deadline missing

표 4는 모의실험에 필요한 기본적인 매개변수와 의미를 나타낸다. TriggerProb는 사건에 의해서 조건이 만족하여 능동규칙의 대응하는 처리가 실행될 확률을 의미한다. TriggerProb 값이 클수록 동시에 실행되는 실시간 능동 트랜잭션의 수가 증가하게 된다. 본 모의실험에서 제안하는 실시간 동시성 제어 알고리즘과 성능을 비교하기 위한 알고리즘으로 RTA-MVPR[1], OCC APFS[3] 등을 채택하였다. OCC-APFS는 낙관적 동시성 제어기법에 잠금을 적용하여 데이터 일관성

을 유지한다. OCC-APFS는 트랜잭션들 사이의 직렬화가능 순서를 유지하기 위해서 타임 스탬프 개념을 적용하였다. 트랜잭션들 사이의 triggering 관계를 고려하기 위해서 해당 트랜잭션의 triggering 트리 상의 자식 트랜잭션의 수와 우선순위가 데이터 충돌의 해결에 중요한 요인이 된다.

그림 2는 초당 시작되는 트랜잭션의 수에 따른 트랜잭션의 철회 비율을 보여준다. 그림 2에서 나타난바와 같이 제안하는 알고리즘인 RTA-MVWTR이 다른 알고리즘에 비해서 철회 비율이 현저하게 낮은 걸 알 수 있다. 특히, 트랜잭션의 수가 60을 넘기면서 RTA-MVPR과 OCC-APFS는 철회 비율이 상당히 증가하는데 비해서 RTA-MVWTR은 철회 비율은 완만하게 증가함을 볼 수 있다. 이런 결과가 나온 원인은 RTA-MVWTR이 시스템의 부하가 커질수록 Thomas 쓰기 규칙에 의해서 다른 알고리즘에서는 철회될 수 있었던 트랜잭션들을 계속 수행하도록 하기 때문이다.

그림 3은 초당 시작되는 트랜잭션의 수에 따른 종료시한을 만족하지 못한 트랜잭션의 비율을 보여준다. 이 결과에서 RTA-MVPR과 OCC-APFS는 시스템 내의 트랜잭션의 수가 증가할수록 종료시한을 넘기는 트랜잭션의 비율이 상당히 증가함을 알 수 있다. 그에 반에 RTA-MVWTR은 트랜잭션의 수가 90인 경우 21.2% 정도의 트랜잭션들이 종료시한을 만족하지 못함을 볼 수 있다. 이와 같은 결과는 그림 2의 트랜잭션의 철회 비율과 밀접한 관계를 가진다. 모의 실험에서 철회된 트랜잭션은 재수행되기 때문에 종료시한을 만족할 여유시간이 감소하게 된다. 따라서, 철회 비율이 높을수록 재수행되는 트랜잭션들은 점점 종료시한을 만족하기 힘들기 때문에 RTA-MVWTR이 다른 알고리즘에 비해서 종료시한을 만족하는 트랜잭션의 비율이 높은 것이다.

VI. 결론 및 향후 연구 방향

본 논문에서는 최근 각광 받고 있는 연구 분야인 실시간 능동 데이터베이스 환경을 위한 동시성 제어 기법을 제안하였다. 제안하는 동시성 제어 기법은 triggering 관계와 선행 관계를 함께 고려함으로서 보다 효율적으로 트랜잭션들 사이의 충돌을 해결하고자 시도하였다. 쓰기 연산들 사이의 관계를 직렬화가능 순서에 맞도록 재배열하는 Thomas 쓰기 규칙을 적용함으로써 불필요하게 철회되는 트랜잭션의 수를 감소시켰다. 또한, 데이터 충돌을 해결할 때 단순히 트랜잭션들이 포함된 triggering 트리 내 우선순위의 합뿐 아니라 트랜잭션의 수도 고려하여 좀 더 유용하게 트랜잭션을 철회하고자 하였다. 모의실험을 통해서 OCC-APFS, RTA-MVPR 등과 같은 알고리즘과 성능 평가를 하였다. 성능평가 결과에서 제안한 알고리즘인 RTA-MVWTR이 Thomas 쓰기 규칙과 triggering 트리의 정보를 활용하여 OCC-APFS와 RTA-MVPR 보다 트랜잭션 철회 비율이 낮고 유용한 재수행 비율이 높음을 보였다.

향후 연구 방향은 RTA-MVWTR이 적용되는 실시간 능동 DBMS의 프로토타입을 구현하여 실제 데이터베이스 환경에 적합한지를 검증하는 것이다.

참 고 문 헌

[1] 김명호, 홍석희, "선행관계를 고려한 다중버전 기반 실시간 동시성 제어 기법", 한국정보과학회지, 제24권, 제11호, pp. 1123-1133, 1997.

[2] 홍석희, "실시간 능동 데이터베이스에서 triggering 관계를 고려한 동시성 제어 기법", 한국정보처리학회 논문지 제8-D권, 제1호, pp. 10-23, 2001.

[3] A. Bajaj A. Datta and R. Veloo, "A study of concurrency control in real-time active database systems", Technical Report RTRG -TR-95-06, Dept. of MIS, Univ. of Arizona, Tucson, AZ 85721, August 1995.

[4] A. Bernstein, A. Philip, V. Hadzilacos and N. Goodman, "Concurrency control and Recovery in database systems", Addison-Wesley, 1987.

[5] O. Ulusoy, "Transaction processing in distributed active real-time database systems", the Journal of Systems and Software, 42, pp. 247-262, 1998.

[6] A. P. Sistla and O. Wolfson, "Temporal triggers in active databases. IEEE Trans. on Knowledge and Data Engineering", Vol. 7, No. 3, pp.

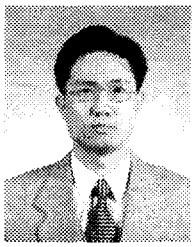
471-486, June 1995.

[7] B. Purimetla, R. M. Sivasankaran, and J.Stankovic, "Priority Assignment in Real-Time Active Databases", VLDB Journal, January, 1996.

[8] Univ. of Newcastle upon Tyne, "C++SIM User's Guide", 1996.

[9] U. Dayal, et. al., "The HiPAC Project" Combining active database and timing constraints", ACM SIGMOD Record, Vol. 17, No. 1, pp. 51-70, 1988.

저 자 소 개



홍석희(Seok-Hee Hong)

1989년 홍익대학교 공과대학 전자계산학과 졸업(학사)

1991년 한국과학기술원 전산학과 졸업(석사)

1997년 한국과학기술원 전산학과 졸업(박사)

1997년 3월~8월 한국전자통신연구원 데이터베이스 연구실 Post Doc.

1997년 9월~현재 경성대학교 정보과학부 조교수

※관심분야 : 실시간 데이터베이스, 능동 데이터베이스, 동시성 제어, 트랜잭션 처리, 저장 시스템