通信情報保護學會論文誌
第6卷 第1號, 1996. 3

2버전 2단계 로킹을 기반으로 하는 다단계 보안 스케줄러[1)]

유 진 호*, 박 석*

Multilevel Secure Scheduler based on Two Version Two Phase Locking

Jin-ho Yoo, Seog Park

요 약

데이타베이스 시스템의 다단계 보안을 위해 비밀채널의 제거는 필수적인 것이다. 본 논문은 비밀채널[8]을 제거하고 성능향상을 위해 보안 2버전 2단계 로킹을 제안한다. 보안 2버전 2단계 로킹은 기본 2PL에 부가적으로 2개의 버전과 보안 요소를 고려한 것이다. 즉, 상위등급 트랜잭션 판독과 하위등급 트랜잭션 기록 사이의 충돌을 피하고 상위등급 판독연산과 동일등급 사이의 성능을 향상시키기 위한 방법이다. 기존의 보안 2단계 로킹기법과는 달리 2개 버전의 데이타와 회사기법을 사용하여, 데이타의 신선도가 떨어지지 않으면서 성능의 향상을 얻을 수 있게 한다.

Abstract

The elimination of covert channel is needed to ensure the Multi-level security of Database System. In this paper, secure two version two phase locking(Secure 2V2PL) is proposed to eliminate the covert channel[8] and to provide more efficient performance. Secure 2V2PL consists of two versions and secure elements on basic 2PL. Secure 2V2PL is the method of avoiding the confliction between high-level transaction read and low-level transaction write. It also improves the performance of read operations of high-level transaction and the performance among same-level transactions. Using two version data and a donation scheme in a different way with S2PL scheme, we can get some benefit for reading the lastest version of data.

---

* 서강대학교 전자계산학과 데이타베이스 연구실

1) 본 논문은 정보통신연구관리단의 "다단계 보안 데이타베이스 시스템 개발 연구"(1995) 과제의 일부로 수행 되었음

1. 서 론

정보가 많고 다양해짐에 따라서 사용자들은 데이타베이스의 기능을 의지하게 된다. 원활한 정보량의 처리 못지 않게 기밀성이 있는 정보의 취급을 위한 보안 요소를 필요로 하게된다. 이러한 보안 요소 중 하나인 비밀채널[8]의 제거는 기밀성 보호를 위해 꼭 필요한 것이다.

데이타베이스 시스템에서 트랜잭션 연산의 스케줄링 시에 발생되는 비밀채널(covert channel)은 원하지 않는 데이타의 누출을 야기시키는 것으로써 시스템 보안에 치명적인 피해를 준다. 이러한 비밀채널을 생성하는 연산 스케줄은 연산충돌에 의한 것인데 서로 다른 등급 즉, 상위등급과 하위등급 트랜잭션의 충돌연산에 의한 것이다. 서로 다른 등급의 연산간 충돌로 인한 비밀채널을 제거하는 방법으로 상위 트랜잭션만을 위한 판독전용 데이타를 둔다. 상위등급 판독전용 데이타를 통해 상위 트랜잭션과 하위 트랜잭션간의 충돌을 피하며 성능향상까지 얻을 수 있다. 이러한 방법으로 기존의 보안 2PL[7]에 비해 데이타의 신선도가 떨어지지 않으면서 더 좋은 성능향상을 얻을 수 있다. 그리고 보안 2PL보다 더 신선한 데이타 판독을 위해서 트랜잭션이 사용하고 다시 사용하지 않을 데이타를 대여해 주는 방법을 고려한다. 보안 2PL의 데이타 신선도라는 장점을 그대로 가지면서 상위 트랜잭션을 위한 판독전용 데이타를 두어 성능향상을 얻어 내고자 한다. 이러한 성능향상은 상위 트랜잭션과 하위 트랜잭션 사이의 연산에 관한 것이다. 동일 트랜잭션 사이의 연산을 위한 성능향상을 위해서는 이미 사용한 후 종료까지 다시 접근하지 않을 데이타를 다른 트랜잭션에게 대여하는 방법을 사용한다. 아울러 보안 2PL과 다중버전 기법과의 차이점을 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 다단계 보안 관련 연구를 위해서 보안을 위한 기본개념과 기존에 있던 스케줄러에 대해서 살펴 보고, 3장에서는 본 논문이 제시하는 보안 2버전 2단계 로킹 스케줄러에 대한 개념과 스케줄러 모델 그리고 연산 스케줄을 위한 규칙을 살펴본다. 그리고 4장에서는 동일등급에서의 성능향상 방안과 희사(donation)로 인한 상위등급 트랜잭션의 신선도 향상을 살피고, 5장에서는 기존 스케줄러인 보안 2단계 로킹 기법과의 비교를 통해서 개선된 점을 관찰하고, 6장에서는 성능평가를 통해 어느 정도의 향상이 있었는지를 실험을 통해서 알아 본다. 마지막 7장에서는 결론을 내린다.

2. 다단계 보안 관련 연구

2.1 기본 개념

보안에 관한 연구가 진행되면서 다단계 보안 데이타베이스 관리 시스템(Multilevel Secure Database Management System : MLS/DBMS)에 대한 연구도 진행이 되었다. MLS/DBMS는 시스템이 하나 이상의 보안인가를 가진 사용자에 의해 공유되고 시스템 내의 데이타들이 하나 이상의 보안 등급을 가진 시스템이다. MLS/DBMS는 두가지 특성을 갖는다. 우선은 MLS/DBMS가 제어하는 모든 데이타 항목은 자신과 관련된 고유한 등급을 갖는다. 다음으로 사용자가 데이타에 접근하는 것은 사용자의 보안 인가와 데이타의 보안 등급에 따라 제어된다. 이러한 시스템 상에서 상위등급 정보의 기밀성은 보호가 되어야 한다. 즉, 하위등급으로 유출되어서는 안된다. 보안 모델 상에서 제약사항을 두더라도 트랜잭션 연산의 스케줄링 시에는 임의의 채널을 통해 기밀성이 있는 정보가 하위등급으로 유출될 수가 있다. 이렇게 스케줄링 시에 발생

할 수 있는 비밀정보의 흐름을 비밀채널이라고 한다. 비밀채널은 시스템이 내세운 보안정책을 위반하여 정보가 상위 보안등급에서 하위 보안등급으로 흘러가는 것을 허용한다. 이러한 비밀채널의 종류에는 시간 비밀채널과 저장장소 비밀채널로 나누는 견해[5]가 있고 지연 보안, 값 보안, 회복 보안으로 나누는 견해[6]가 있다. 시간 비밀채널은 예 1과 같은 불법적인 정보의 흐름을 만드는 것이다. 저장장소 비밀채널이라는 것은 상위등급에 의해서 이미 생성된 화일을 하위등급의 주체가 같은 화일이름으로된 화일을 생성시 "접근 거부"라는 메시지를 받는다면 하위등급의 주체는 자신보다 상위등급에 그 화일이 있음을 알게 되므로써 정보가 아래로 흐르는 효과가 나타나게 되어 생기는 채널이다. 지연 보안이라는 것은 하위등급 트랜잭션의 지연이 상위등급의 트랜잭션에 의해서 생기는 현상으로 이것도 정보의 하향 흐름이 일어나 채널이 생성된다. 값 보안이라는 것은 하위등급 트랜잭션이 판독하는 데이타 값이 상위등급 트랜잭션에 의해서 변할 때 정보의 하향 흐름이 존재하여 생기는 채널이다. 그리고 회복 보안이라는 것은 어떤 스케줄에서 상위 트랜잭션의 존재여부에 의해서 교착상태 발생 여부가 결정되므로써 생기는 채널의 경우이다. 이것을 막는 MLS/DBMS를 운영하기 위해서는 시스템이 가정하는 보안정책과 이에 맞는 보안모델이 있어야만 한다.

본 논문에서 가정하는 보안모델은 아래와 같은 제한 사항을 갖는다.

1) 트랜잭션 T는 다음의 조건을 만족할 때에만 데이타 항목 X에 대한 판독연산이 가능하다.

조건 : 트랜잭션 T의 보안등급
$L(T) \geq L(X)$

2) 트랜잭션 T는 다음의 조건을 만족할 때에만 데이타 항목 X에 대한 기록연산이 가능하다.

조건 : 트랜잭션 T의 보안등급
$L(T) = L(X)$

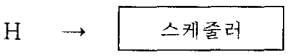
보안등급을 이루는 구성요소는 계층적 등급(hierarchical level)과 범주 집합(category set)이다. 계층적등급은 수직적으로 순서화되어 있고 범주집합은 수평적으로 순서화되어 있다. 보안등급은 L = (A, B)와 같이 정의된다. 여기서 A는 계층적 등급으로 2급 비밀, 3급 비밀, 대외비, 평문 등으로 구성되고, B는 범주집합으로 인사, 작전, 군수, 정보 등으로 구성된다.

【예 1】 시간 비밀 채널

$T_1(S)$		R(x)		C_{HL}
$T_2(U)$	W(x)		Cert(x)	

위의 예에 의하면 $T_1(S)$이 R(x)를 수행하고 있을 때, $T_2(U)$의 Cert(x)연산이 제기되면, 연산 $T_2(U)$의 Cert(x)연산은 지연되고, 이에 대해서 Cert(x)연산을 수행한 $T_2(U)$는 $T_1(S)$의 신호를 받게 된다. 이러한 신호는 비트 정보를 전달할 수가 있고 대량의 연산을 통해서 대량의 비트정보가 전달되어 메시지 전달효과를 가져올 수가 있다. 이렇듯 시간지연에 의해서 정보가 전송되는 효과를 가진 자료흐름을 시간 비밀 채널이라고 한다.

보안 정확성 기준을 위해 purge연산을 정의하자. 입력 스케줄을 H, 출력 스케줄을 H′, 보안 등급을 L이라고 한다. 스케줄러에 입력이 되는 스케줄과 출력 스케줄을 도식화하면 다음과 같다.



[그림 1] 스케줄러에 의한 출력 스케줄

스케줄 H에서 보안 등급이 L보다 높은 연산들을 제거하고 남은 스케줄을 purge(H, L)이라고 한다. 본 논문에서는 보안 스케줄러의 정확성 기준을 세 가지 요소를 통해 평가할 것이다. 다음의 3가지 성질을 만족하는 스케줄러를 보안 스케줄러라고 한다[6].

【성질 1】 값 보안(value security)

출력 스케줄 H′가 값 보안 성질을 만족하는 경우는 다음과 같다. 스케줄 purge(H′, L)는 스케줄 purge(H, L)을 입력 스케줄로 한 경우의 출력 스케줄 h와 뷰 동일(view equivalent)[2] [1,4]하다. 이것은 하나의 주체가 판독한 데이타의 값은 더 높은 등급을 가진 주체에 의해 영향을 받지 않음을 의미한다. □

값 보안을 도식화하면 다음과 같다

[그림 2] 값 보안

【성질 2】 지연 보안(delay security)

연산 O_1이 스케줄 purge(H, L)을 입력 스케줄로 해서 나온 출력 스케줄 h에서 O_2에 의해 지연되었다면 스케줄 purge(H′, L)에서도 지연된다. □

【성질 3】 회복 보안(recovery security)

교착 상태가 발생하면 스케줄 내에 높은 등급의 행동이 있어도 이와는 독립적으로 하위등급의 주체에게는 모두 동일하게 보여야 하며, 교착상태에서 회복하기 위해 취해진 행동이 높은 등급의 트랜잭션에 의해 영향받지 않아야 한다. □

보안 환경에서 사용되는 모든 스케줄링 방법이 보안 스케줄러의 정확성 기준을 만족하는지에 대한 증명이 필요하며 위의 3가지 조건을 만족하면서 병행수행 정도도 높일 수 있는 보안 스케줄러를 제안하는 것이 본 논문의 목표이다.

2.2 기존의 보안 스케줄러

비밀채널[8] 방지를 위해서 Keefe[6]나 Jajodia[11]가 다중버전 보안 알고리즘에 대한 연구를 하였다. Keefe의 알고리즘에는 구버전 판독의 문제가 있었고, Jajodia의 연구에서는 상위등급 트랜잭션의 불합리한 블러킹의 문제가 있었다. 본 연구실에서 발표한 논문[12]인 "보안환경에서 트랜잭션 분류에 기초한 병행수행 제어"에서는 이러한 문제를 하위 보안등급을 갖는 데이타를 판독한 상위등급 트랜잭션이 유도갱신이거나 판독전용일 때는 블럭될 필요가 없으므로 블럭시키지 않고, 순수갱신이거나 기록전용일 경우에도 이미 기록을 수행하였다면 블럭되지 않도록 하였다. 이렇게 하여 구버전 판독의 단점을 없애면서도 전체적으로 트랜잭션의 병행수행의 정도를 높인 연구를 수행 발표하였다.

로크기반 스케줄러에 대한 연구로는 보안 2PL[7]이 있다. 보안 2PL은 스케줄러가 스케줄

2) 2개의 스케줄이 같은 트랜잭션과 같은 연산으로 구성되며 기록-판독 관계(read-from relationship)가 같고 각 데이타 항목에 대한 마지막 기록(final write)이 동일한 경우를 말한다.

을 할 때 만들어 질 수 있는 비밀채널을 막는 기법을 연구한 것으로 비밀채널을 지연보안의 관점에서 접근한 것이다. 보안 2PL 스케줄러는 비밀채널 제거를 위해서 몇 개의 부가적인 리스트와 판독 및 기록 연산을 트랜잭션 지역공간에 하는 것으로 지연보안의 문제를 해결한다. 하지만, 보안 2PL이 장점으로 보았던 데이타의 신선도가 이러한 지역공간에 연루된 연산때문에 나빠진다. 이러한 지역공간에 연루된 연산을 하면 데이타의 일관성에 문제가 있을 수 있다. 하지만, 보안 2PL이 동적인 스케줄러로써 구성되었다는 것은 장점이라 할 수가 있다.

3. 보안 2버전 2단계 로킹 스케줄러

기존의 보안 2PL[7]은 다중버전 기법에 비해서 적은 기억장소를 사용하면서 데이타의 신선도가 앞선다는 것이 장점이었다. 보안 2PL에서는 상위 트랜잭션과 하위 트랜잭션의 충돌을 해결하기 위해서 지역기록을 사용하였다. 보안 2PL은 로크기반의 스케줄러이기 때문에 기본적으로 데이타의 신선도가 우수하다. 그러나 지역기록의 사용으로 인하여 실제기록이 지연되어 사실상은 기본 2PL만큼의 신선한 데이타를 판독하지는 못한다. 보안 2PL은 가상적인 기록과 판독이 있고 상위 트랜잭션의 지연으로 인해 성능이 좋지 못하다. 게다가 기본 2PL에 있는 지연 등의 특성은 그대로 가지고 있다. 그래서 본 논문에서는 보안 2PL이 갖는 단점인 지역기록과 지연으로 인한 성능저하와 비밀채널 문제를 해결하여 더 좋은 성능을 가지며, 데이타의 신선도 또한 보안 2PL에 떨어지지 않는 스케줄러를 제안한다.

지연의 문제에 있어서 판독과 기록 간의 지연을 기존의 2버전 2단계 로킹기법의 판독과 확인 간의 지연으로 변형하면 지연시간을 줄일 수 있다[10]. 2버전 2단계 로킹기법의 사용을 통해서 판독과 기록 간의 충돌과 지연시간으로 인한 부하의 해결이 쉬워진다. 그리고 보안 2버전 2단계 로킹 스케줄러는 상위 트랜잭션의 판독요구와 하위 트랜잭션 확인 사이의 충돌을 피하기 위해서 상위등급 판독전용 데이타를 둔다. 그리고 동일한 등급의 두 버전 중 한 개의 버전은 복사본 데이타이고 다른 데이타는 사용 중인 데이타이다. 이러한 지연의 문제와 충돌로 인한 비밀채널을 고려하여 보안 2버전 2단계 로킹을 구성한다.

보안 2버전 2단계 로킹 스케줄러로 구성할 수 있는 기법에는 동적 기법, 선언적 기법, 정적 기법[9] 등이 있는데 선언적 기법(Declaration scheme)은 트랜잭션이 어떤 명령으로 구성되어 있는가는 알 수 있지만, 동시에 수행되는 병행수행 트랜잭션 간에서 어떤 연산이 어떤 순서로 제출될 지는 알 수 없다고 가정하고 처리하는 기법이다. 트랜잭션 수행시에 컴파일 등의 과정에서 그에 해당하는 트랜잭션의 연산명령 집합을 예측할 수 있기 때문에 보안 2버전 2단계 로킹 스케줄러에서는 선언적 기법을 사용할 것이다.

3.1 스케줄러 모델

2버전 2단계 로킹에서는 각 데이타 항목에 대해서 많아야 2개의 버전이 존재한다. 데이타 항목이 갱신될 때, 복사본 데이타는 갱신되지 않고 대신에 새로운 버전이 만들어 진다. 보안 2버전 2단계 로킹에서는 2버전 2단계 로킹의 2개의 데이타에 상위 트랜잭션 판독전용 데이타 하나를 더 둔다. 상위 트랜잭션에게는 상위 트랜잭션 판독전용 데이타에 대한 판독만을 허용하여 상위 트랜잭션의 판독과 하위 트랜잭션의 확인으로 인해 발생하는 비밀채널을 막는다. 그리고 두개의 버전을 사용하여 동일

등급 두 트랜잭션 간의 판독과 기록의 충돌로 인한 성능저하를 해결한다. 왜냐하면 트랜잭션 간의 판독과 기록간의 충돌을 판독과 확인 간의 충돌로 변화시키면 지연 효과가 덜하기 때문이다[10].

기본적으로 충돌의 제거는 호환성표에 근거하며, 상위 트랜잭션에게 신선한 데이타를 판독하게 하고 트랜잭션의 병행 수행 정도를 높이기 위해 희사기법을 도입한다. 본 논문에서는 보안 수행을 위한 상위등급 판독전용 데이타의 관리와 희사 기법에 촛점을 둘 것이다.

스케줄러의 모델은 보안 2버전 2단계 로킹 스케줄러의 요구사항을 위해서 그림 1과 같은 모델을 사용한다. 트랜잭션 스케줄러는 생성된 트랜잭션의 연산정보를 받고 연산 스케줄러에게 넘겨 준다. 연산 스케줄러에 의해 스케줄된 연산은 보안 2버전 2단계 보안 프로토콜 제어기에 보내진다.

보안 프로토콜 제어기는 입력된 연산에 대해 연산이 수행될 것인가를 보안 2버전 2단계 로킹기법에 의해 결정한다. 보안 프로토콜 제어기는 로크요구가 합당하면 데이타 관리기에 보내서 연산을 수행하고, 그렇지 않으면 연산은 블럭되어 큐로 들어가 다음 기회를 기다려야 한다.

3.2 보안 2버전 2단계 로킹

보안 2버전 2단계 로킹에서 상위등급 트랜잭션의 판독요구는 결코 지연되지 않는 반면, 스케줄러는 항상 새로운 값이 사용 가능하지는 않기 때문에 판독로크을 필요로 한다. 그리고 여전히 그 복사본 데이타 값을 판독하는 트랜잭션이 있는지 없는지를 알아야만 한다.

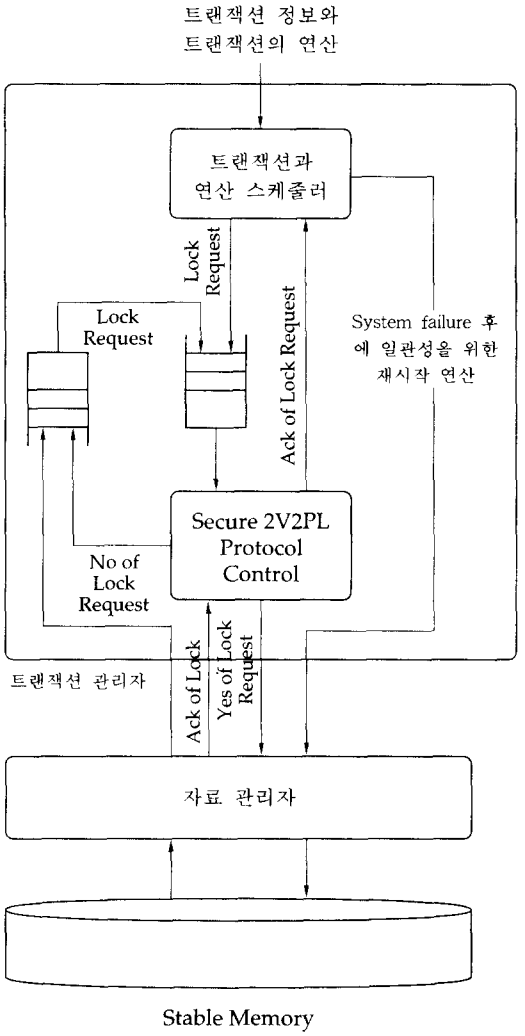


그림 1 스케줄러 모델

표 1 2버전 2단계 로킹의 호환성표

	Read	Write	Certify
Read	yes	yes	no$^+$
Write	yes	no	no
Certify	no$^+$	no	no

상위 트랜잭션은 하위의 데이타 항목에 대해 판독동작만을 하기때문에 생각해야 할 것

은 상위 트랜잭션의 판독과 하위 트랜잭션의 확인 간의 충돌이다. 그래서, 표 1의 2버전 2단계 로킹의 호환성표에서 보는 바와 같이 no^{+}에서 지연보안에 의한 비밀채널을 생성할 위험이 있다. 이것은 트랜잭션이 한 데이타 항목에 관하여 2개의 데이타와 상위등급 판독전용 데이타가 있다고 하면, 비밀채널이 없는 호환성표를 구성할 수 있다. 상위등급 트랜잭션은 하나의 데이타 항목에 대해서 2개의 데이타와 상위등급 판독전용 데이타를 가져서 비동기적인 판독을 한다. 이렇게 2개의 데이타와 상위등급 판독전용 데이타를 둔다고 해도 스케줄에서 발생하는 비밀채널을 제거하려면 몇가지 추가적인 규칙이 필요하다. 다음 절에서 몇가지 예를 통해 규칙을 정의하도록 한다.

3.3 보안 2버전 2단계 로킹 프로토콜

로크의 의미는 로크대상이 되는 데이타 항목에 대해서 로크를 가진 트랜잭션만이 접근할 수 있게 하는데 있다. 보안 2버전 2단계 로킹은 이런 점에 착안하여 데이타 항목에 대해 로크가 설정되어 있는 기간 동안 즉, 로크의 생존기간에 한 데이타 항목에 접근하는 트랜잭션에게 같은 값을 보여주면 되는 것이다. 보안 2버전 2단계 로킹은 이러한 사항과 직렬화 순서를 고려하여 만들어진 성능 중심의 프로토콜이다. 다음은 상위등급 판독전용 데이타를 통해서 보안을 지키면서 자신의 성능을 향상시키는 예들이다.

보안 2버전 2단계 로킹이 사용되므로 데이타는 상위등급 판독전용 데이타(x_{bc}), 복사본 데이타(x_c), 사용중인 데이타(x_w)로 분류된다. 아래의 몇가지 경우를 통해서 보안 2버전 2단계 로킹기법을 완성하도록 한다.

【예 2】 상위등급 판독과 하위등급 확인으로 인한 비밀채널의 제거

$T_1(S)$		$R(x_{bc})$			C_1
$T_2(U)$	$W(x_w)$		$Cert(x)$	C_2	

기존의 $Cert_2(x)$의 동작은 $x_c \leftarrow x_w$이고, 제안하는 $Cert_2(x)$의 동작도 $x_c \leftarrow x_w$이다. C_1이 완료한 후 $x_{bc} \leftarrow x_c$를 해서 비동기적으로 x_{bc}에 대한 확인을 지연시킨다. 여기서 x_{bc}는 상위등급 판독전용 데이타로써 상위 트랜잭션이 하위 트랜잭션과의 충돌에 의해 지연됨이 없이 하나의 버전을 판독하는 것을 허락하기 위해 마련된 데이타 항목이다. $T_1(S)$는 x_{bc}를 읽게 되고, $T_1(S)$가 종료하면, $x_{bc} \leftarrow x_c$가 되고 이 때 부터 상위 트랜잭션은 x_c의 내용을 가진 x_{bc}를 읽는다.

【예 3】 다수의 상위 트랜잭션 판독과 하위 트랜잭션 확인으로 인한 비밀채널의 제거

$T_1(S)$				$R(x_{bc})C_1$	
$T_2(S)$			$R(x_{bc})C_2$		
$T_3(S)$		$R(x_{bc})C_3$			
$T_4(U)$	$W(x_w)$				$Cert(x)C_4$

제안하는 Cert(x)의 연산은 $x_c \leftarrow x_w$이다. C_1이 완료한 후 $x_{bc} \leftarrow x_c$를 해서 비동기적으로 확인을 지연시킨다. 위와 같은 경우의 예는 기존의 스케줄과 같은 결과를 가져오는 평범한 보안 2버전 2단계 로킹의 적용 예이다.

【예 4】 프로토콜을 완성하기 위한 사이클이 있는 하나의 예

$T_1(TS)$	$R(y_{bc})$				$R(x_{bc})C_1$
$T_2(S)$			$W(y_w)R(x_{bc})Cert(y)C_2$		
$T_3(U)$		$W(x_w)$		$Cert(x)C_3$	

위의 스케줄의 Cert(x)에서 x_{bc}에 대한 갱신이 일어난다면 트랜잭션 T_1의 $R(x_{bc})$로 부터 사이클($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$)이 생성이 된다 (단, $T_1(TS)$의 x_{bc}는 갱신된 x_c의 값을 가지고 있슴). 이는 bc버전 데이타의 갱신을 위한 예로 상위등급 트랜잭션의 판독집합에 포함이 되지 않을 경우에만 확인 연산 시에 bc버전의 데이타가 갱신된다는 것을 보여준다. 위의 스케줄로 부터 프로토콜을 위한 규칙을 정의할 수가 있다.

보안 2버전 2단계 로킹의 규칙

규칙 1. x_{bc}는 상위등급 트랜잭션의 판독을 위한 것인데, 갱신되는 시점이 중요하며, x_{bc}는 상위등급 트랜잭션으로 하여 한단계 구버전을 갖는다. x_c는 동일등급 간의 데이타를 처리하기위한 확인된 버전의 데이타이다.

규칙 2.

$T_1(S)$		$R(x)$		C_{HL}
$T_2(U)$	$W(x)$		$Cert(x)$	

위의 예에서는 C_{HL}의 시점에서 $x_{bc} \leftarrow x_c$ 이 수행이 된다. 즉, 현재 수행 중인 상위등급 트랜잭션 판독집합에 포함되지 않으면, Cert(x)에서 $x_b \leftarrow x_c$를 수행하고, 포함이 되면 그 판독연산이 있는 상위 트랜잭션의 완료연산(C_{HL})에서 $x_{bc} \leftarrow x_c$를 수행한다.

규칙 3. 규칙 1의 모든 판독은 판독대상이 되는 데이타가 모두 x_{bc} 버전을 읽거나, 모두 갱신된 x_{bc} 버전을 읽어야만 한다.

이렇게 하여 규칙을 적용한 후의 스케줄은 다음과 같이 편성된다.

$T_1(TS)$	$R(y_{bc})$				$R(x_{bc})C_1$
$T_2(S)$			$W(y_w)R(x_{bc})Cert(y)C_2$		
$T_3(U)$		$W(x_w)$		$Cert(x)C_3$	

여기서 $T_1(TS)$의 x_{bc}는 규칙 2에 의해서 갱신되지 않은 x_c의 값을 가지고 있으므로 사이클이 생성되지 않는다. 즉, 상위등급 트랜잭션의 판독집합에 포함이 되지 않을 경우에만 확인 연산시에 bc버전의 데이타가 갱신이 될 수 있고 그렇지 않은 경우에는 판독 연산을 가진 상위등급 트랜잭션이 종료된 후에 bc버전의 데이타를 갱신해야만 한다.

4. 동일 등급의 성능향상을 위한 방안

2PL을 확장시킨 Altruistic Locking은 트랜잭션이 한 번 접근한 후에 더 이상 사용하지 않는 데이타는 다른 트랜잭션이 사용할 수 있도록 하는 기법을 사용한다. 이러한 기법은 Altruistic Locking[3]을 그 기반으로 한다. 이 기법은 자원의 효율적인 사용에 의하여 트랜잭션들의 직렬화 순서를 어기지 않으면서 성능을 향상시키는 방법이다.

한 번 접근한 후에 더 이상 쓰지 않는 데이타는 다른 트랜잭션에게 쓸 수 있도록 하는 기법을 희사기법이라 한다. 희사된 데이타 항목을 사용하려는 트랜잭션은 일정한 조건이 만족되면 그 희사된 데이타 항목을 사용할 수 있다[3]. 이러한 조건을 로킹 스케줄러에 반영하여 로킹에 대한 동작을 수정하도록 한다. 또한 이 기법은 자원을 사용하지 않으면서 오래 점유하는 트랜잭션이 존재할 경우 특히 유용하며, Altruistic Locking 기법은 2PL을 기반으로 이루어져 있으므로 2버전 2단계 로킹으로의 구현에도 확장이 용이하다. 이 기법은 직렬화순서를 만족하고, 수행시간이 긴 트랜잭션들의 성능에 큰 향상을 준다. 왜냐하면, 사용하지 않고 장시간 로크를 당하는 데이타 항목의 로크를 희사하여 다른 트랜잭션에게 데이타를 사용할 수 있게 하기 때문이다. 즉, 직렬화순서를 지키면서 자원사용의 효율을 기하여 성능을 향상시킨다.

4.1 Altruistic Locking Protocol

Altruistic Locking의 대상이 되는 스케줄은 그림 2와 같다. 그림 2와 같이 트랜잭션 T_1이 수행 중에 있고 이미 트랜잭션 T_1은 데이타 항목 A, B, C를 사용했고 D를 사용하고 있다. 트랜잭션 T_1의 수행 도중에 트랜잭션 T_2, T_3, T_4가 데이타의 사용을 요구한다고 하자. 기존의 방법에 의하면 트랜잭션 T_4만이 허락될 수 있다. 그러나, 여기서 T_3는 어차피 로크를 허용해도 직렬화 순서를 어기게 되므로 사용할 수가 없지만, 트랜잭션 T_2가 접근하는 데이타 항목 A, B는 트랜잭션 T_1에 의해서 더 이상 사용되지 않는 데이타 항목이므로 허락될 수 있다는 가능성이 있다. 이러한 가능성을 이용하여 프로토콜을 구성한 것이 Altruistic Locking Protocol이다. 이러한 방법은 자원을 효율적으로 사용하는 것에 의해서 병행성을 높여 성능향상을 도모한 것이다.

Altruistic Locking 규칙을 수행하는 프로토콜을 설계하는 것은 그리 어렵지 않다. 기존의 2PL에 몇가지 희사에 대한 자료구조와 동작을 부가하면 된다.

A, B : T_2에 의해 사용이 요구되는 데이타
A, B, C : T_1에 의해 이미 사용된 데이타
C, E : T_3에 의해 사용이 요구되는 데이타
D : T_1에 의해 현재 사용되는 데이타
F, G : T_4에 의해 사용이 요구되는 데이타
E, F, G : T_1이 나중에 사용할 데이타

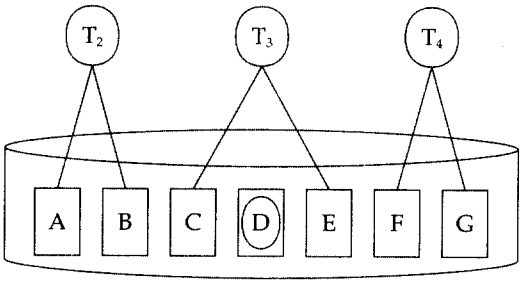


그림 2 Altruistic Locking이 병행성을 확장시키는 예

트랜잭션 T_2가 트랜잭션 T_1에 의해 희사된 데이타 항목을 로크했다고 가정하면, 트랜잭션 T_2가 트랜잭션 T_1에 대하여 충돌관계(indebt)

에 있다는 것은 두 트랜잭션의 로크가 충돌하거나, 세번째 트랜잭션에 의한 로크가 두 트랜잭션과 충돌한다는 것과 동치이다. 이러한 관점을 이용하여 마련된 Altruistic Locking Protocol[3]은 다음과 같다. 첫째는 트랜잭션 중 하나의 트랜잭션이 먼저 객체를 희사하지 않으면, 같은 객체 상에 동시에 로크를 가질 수 없다. 둘째는 트랜잭션 T_a가 또 다른 트랜잭션 Tb에 대하여 충돌관계(indebt)에 있다면, 그것은 T_b가 그것의 로크해제를 수행할 때까지 완전하게 T_b의 포함상태에 있어야만 한다.

본 연구는 희사기법을 적용하여 보안 2버전 2단계 로킹 프로토콜의 규칙을 다음과 같이 수정하고, 관련 알고리즘을 연구하였다.

보안 2버전 2단계 로킹의 규칙

규칙 1. x_{bc}는 상위등급 트랜잭션의 판독을 위한 것인데, 갱신되는 시점이 중요하며, x_{bc}는 상위등급 트랜잭션으로 하여 한단계 구버전을 갖는다. x_c는 동일등급 간의 데이타를 처리하기 위한 확인된 버전의 데이타이다.

규칙 2.

T_1(S)		R(x)		C_{HL}
T_2(U)	W(x)		Cert(x)	

위의 예에서는 C_{HL}의 시점에서 $x_{bc} \leftarrow x_c$이 수행이 된다. 즉, 현재 수행 중인 상위등급 트랜잭션 판독집합에 포함되지 않거나 희사되어 있으면, Cert(x) 시점에서 $x_{bc} \leftarrow x_c$를 수행된다. 포함이 되면 그 판독연산이 있는 상위 트랜잭션의 완료연산(C_{HL})에서 $x_{bc} \leftarrow x_c$가 수행된다.

규칙 3.

트랜잭션 중 하나의 트랜잭션이 먼저 객체를 희사하지 않으면, 같은 객체 상에 동시에 로크를 가질 수 없다.

규칙 4.

트랜잭션 T_a가 또 다른 트랜잭션 T_b에 대하여 충돌관계(indebt)에 있다면, 그것은 T_b가 그것의 로크해제를 수행할 때 까지 완전하게 T_b의 포함상태에 있어야만 한다.

규칙 5.

희사받는 트랜잭션 T_a가 희사한 트랜잭션 T_b와 충돌관계이면 희사받은 트랜잭션의 연산은 지연이 된다.

규칙 6.

보안등급이 $L_1 < L_2 < L_3$인 T_1, T_2, T_3에 대하여 T_1이 종료후 T_2가 T_1이 기록한 데이타에 대해 판독요구시 T_3가 그 데이타를 판독 후 희사하지 않았거나 판독집합에 포함되었는데 판독을 하지 않았으면 T_3는 취소된다.

T_1(S)				$R(x_{bc})$	C_1	
T_2(U)	W(x)Cert(x)					
T_3(U)			W(x)Cert(x)			
T_4(TS)		W(z)				$R(x_{bc})$

규칙 2에 의해서 트랜잭션 T_3의 Cert(x)에서는 트랜잭션 T_4의 $R(x_{bc})$에 의해서 $x_{bc} \leftarrow x_c$를 수행하지 않는다. 그래서 T_1은 T_2가 갱신한 x_{bc}의 값을 판독한다. 하지만, 트랜잭션 T_4가 없는 경우에 트랜잭션 T_1의 $R(x_{bc})$는 T_3가 갱신한 값을 판독하게 된다. 그러므로 이런 스케줄이 있을 때 트랜잭션 T_4는 취소되어야 값 보안을 만족시킬 수가 있다. 그리고 트랜잭션 T_4의 취소 후에 $x_{bc} \leftarrow x_c$가 수행된 후 트랜잭션 T_1의 $R(x_{bc})$가 수행이 된다.

스케줄러는 받아들여진 연산이 판독, 기록, 확인, 완료인지를 판단하여 각각에 해당하는 등급별로 주어진 데이타에 대해서 동작을 한다. 이와같이 연산이 제출되면 연산과 데이타 항목의 등급을 비교한 후 해당 데이타에 걸려 있는 로크를 판별한 후, 그 데이타에 대한 로크를 허용할 것인지의 여부를 결정한다.

4.2 회사동작이 주는 영향

다음과 같은 스케줄이 있다고 가정하자.

T_1(TS)	$R(y_{bc})$			$R(x_{bc})C_1$
T_2(S)		$W(y_w)R(x_{bc})Cert(y)C_2$		
T_3(TS)			R(ybc)C_3	

위의 스케줄과 아래와 같이 데이타 항목 y를 회사했을 경우 데이타의 신선도를 비교할 수 있다.

T_1(TS)	$R(y_{bc})d(y)$			$R(x_{bc})C_1$
T_2(S)		$W(y_w)R(x_{bc})Cert(y)C_2$		
T_3(TS)			$R(y_{bc})C_3$	

데이타 항목 y는 회사동작 d(y)로 인해서 T_2(S)의 Cert(y)연산이 $y_c \leftarrow y_w$, $y_{bc} \leftarrow y_w$로 수행이 되고, 결과적으로 신선한 데이타를 좀 더 일찍 읽을 수 있다. 상위등급의 트랜잭션이 데이타 항목을 회사하는 것은 그 데이타에 대한 확인동작($x_{bc} \leftarrow x_c$)을 좀 더 일찍 일어나게 한다. 자신 이상의 높은등급 트랜잭션에게 좀 더 신선한 데이타를 판독하게 하고, 다시 사용치 않는 데이타를 대여해 주어서 상호간의 병행성의 정도를 증가시킨다. 회사된 데이타 항목은 보안 2PL보다 더 우수한 데이타의 신선도를 예 5에서 처럼 가질 수 있다. 일단, 데이타가 회사되고 나면 그때 부터는 그 데이타를 사용 가능하기 때문에 이로 인한 데이타 신선도 뿐만아니라 병행성도 좋아진다.

【예 5】 보안 2PL과 보안 2버전 2단계 로킹의 신선도에 있어서의 비교

보안 2PL :

$T_1(S)$					R(y)		C_1
$T_2(S)$	R(x)		R(y)			C_2	
$T_3(U)$		W(x)W(y)		C_3			

보안 2PL 스케줄 :

$rl_2(x)r_2(x)vwl_3(x)vw_3(x)dvwl_3(y)vw_3(y)rl_2(y)r_2(y)C_3rl_1(y)r_1(y)\ \overline{r_1(y)}C_2C_1$ (T_3의 x,y real write)

보안 2버전 2단계 로킹 :

$T_1(S)$					$R(y_{bc})$		C_1
$T_2(S)$	$R(x_{bc})$		$R(y_{bc})D(x)D(y)$			C_2	
$T_3(U)$		$W(x_w)W(y_w)$		$Cert(x)Cert(y)C_3$			

보안 2버전 2단계 로킹 스케줄 :

$rl_2(x_{bc})r_2(x_{bc})wl_3(x_w)w_3(x_w)wl_3(y_w)w_3(y_w)rl_2(y_{bc})d(x)d(y)Cert_3(x)Cert_3(y)\ \overline{C_3}\ \overline{r_1(y_{bc})}C_2C_1$

위와 같은 스케줄에서는 회사된 데이타(x_{bc})를 완료연산에서 갱신하면 그 다음 연산인 $R(y_{bc})$는 갱신된 데이타(y_{bc})를 판독하여 더 신선한 데이타를 판독하게 된다. $Cert_3(x)$에서는 $x_c \leftarrow x_w$가 수행되고, $Cert_3(y)$에서는 $y_c \leftarrow y_w$가 수행된다. C_3에서는 $x_{bc} \leftarrow x_c$, $y_{bc} \leftarrow y_c$이 수행된다. 위의 스케줄을 비교해 보면 사실상 트랜잭션의 수행순서는 $T_2 \rightarrow T_3 \rightarrow T_1$의 순서가 되어야 한다. 즉 각각의 순서에 의해서 데이타값은 영향을 받아야 한다. 그리고 트랜잭션은 T_1은 반드시 T_3에 의해서 쓰여진 값을 읽어야 하는데도 불구하고 보안 2단계 로킹이 만들어 내는 스케줄에서는 그렇게 하지 않음을 볼 수가 있다. 스케줄이 만들어 내는 연산의 순서를 보면, 우선 트랜잭션 T2가 데이타 항목 x에 대해서 판독로크를 걸고 데이타 항목 x를 판독한다. 그리고 난 후 트랜잭션 T_3가 데이타 항목 x에 대한 기록로크를 요구할 때 이러한 요구는 원래 기본 2단계 로킹에서는 블럭시키지만 보안 2단계 로킹에서는 가상로크로 이를 해결하여 보안을 만족시켜야 하므로 데이타 항목에 대한 연산을 지역장소에 하게 된다. 그 다음으로 오는 연산도 비슷한 방법으로 의존적 로크를 하고 연산을 지역장소에 하게 된다. 그리고 트랜잭션 T_2에 의해서 판독 연산이 입력이 되면 실제 그 데이타 항목에 대해서 연산을 하게 되고 트랜잭션 T_3는 완료를 하고 트랜잭션 자신이 가상연산을 한 것이 실제로크를 얻기 위해서 현재 걸려 있는 실제로크가 해제되기를 기다리게 된다. 그러나 트랜잭션 T_2가 데이타 항목 y에 대해 판독로크를 가지고 있으므로 트랜잭션 T_3의 실제연산을 위한 실제로크는 트랜잭션 T_2의 완료 뒤로 미루어 지게 된다. 트랜잭션 T_3가 트랜잭션

T_2의 완료를 기다리고 있을 때 트랜잭션 T_1의 데이타 항목 y에 대한 판독요구를 가지므로 인해서 트랜잭션 T_3의 실제연산을 위한 실제 로크는 지연이 되게 된다. 결국 트랜잭션 T_3의 실제연산을 위한 실제로크는 트랜잭션 T_1의 종료 후로 연기가 되고 트랜잭션 T_1은 트랜잭션 T_3가 쓴 데이타를 읽지 못하게 되어 트랜잭션 T_1이 판독한 데이타는 구버전의 것이 되고 $T_1 \rightarrow T_2 \rightarrow T_3$의 수행순서를 갖는다. 이렇게 하여 트랜잭션 T_1은 구버전 판독의 문제를 갖는다.

이에 대해서 보안 2버전 2단계 로킹 알고리즘은 데이타 항목 y에 대한 트랜잭션 T_2의 판독연산 후에 곧 바로 데이타 항목 x, y를 희사하게 되어서 트랜잭션 T_3는 완료 전에 데이타 항목 x, y에 대해서 확인 연산을 하므로 데이타 x_{bc}, y_{bc}가 갱신된다. 결국 트랜잭션 T_1은 갱신된 bc버전의 데이타를 판독하게 되어 신선도가 좋아진다. □

4.3 보안 기준에 의한 정확성 증명

본 논문에서 제시한 스케줄링 방법이 보안 요구 사항을 만족하는지를 평가할 수 있는 보안 정확성 기준을 2.1절에서 제시하였다. 그러므로 제시한 방법이 보안 정확성 기준인 값 보안, 지연 보안, 회복 보안을 만족하는 스케줄링 방법이라는 것을 증명하면 다음과 같다.

❖ **정리 1** 본 논문에서 제안한 보안 2버전 2단계 로킹 스케줄링 방법은 값 보안 성질을 만족한다.

(증 명)

보안등급이 L_1인 트랜잭션 T_1이 판독연산 $r_1(x)$를 한다고 가정하자. L_1보다 하위 보안등급을 가진 두 개의 트랜잭션 T_{j1}, T_{j2}가 각각 $w_{j1}(x)$, $w_{j2}(x)$를 수행한다고 하고 보안 등급이 $L_2 > L_1$인 트랜잭션 Tk를 가정하자. 이 때 스케줄 내에 T_k가 있는 경우에 $r_1(x)$는 $w_{j1}(x)$가 기록한 x의 버전을 읽고 T_k가 없는 경우에 $w_{j2}(x)$가 기록한 x의 버전을 읽는다면 값 보안 요구 사항을 만족하지 못하는 것이다. 그러므로 본 논문에서 제안한 스케줄링 방법을 사용하면 $r_1(x)$가 T_k에 의해서 영향을 받지 않는다는 것을 보여야 한다. T_k에 의해서 $r_1(x)$가 판독하는 x의 값이 달라질 수 없어야 한다는 것이다. 값 보안을 위배하는 경우는 T_2가 시작되고 바로 T_k가 시작된 후 x에 대한 판독연산을 $Cert_2(x)$연산 후에 수행하는 경우인데 이러한 값 보안 위반의 경우는 규칙 6에 의해서 상위 트랜잭션의 취소에 의해서 해결이 된다. 그러므로 보안 2버전 2단계 로킹 스케줄러는 값 보안 성질을 만족한다. □

❖ **정리 2** 본 논문에서 제안한 보안 2버전 2단계 로킹 스케줄링 방법은 지연 보안 성질을 만족한다.

(증 명)

본 논문에서 제안한 방법에서 하위 보안등급을 가진 트랜잭션의 연산은 상위 보안등급을 가진 트랜잭션에 의해서 지연되지 않는다. 상위등급 트랜잭션이 접근하는 데이타는 bc버전의 데이타이고 사실상 상위 트랜잭션의 판독 요구와 하위 트랜잭션의 확인 요구 사이에는 접근하는 데이타 자체가 분리되어 있으므로 충돌이 있을 수 없으므로 하위 트랜잭션은 결코 어떤 지연도 없다. 그리고 데이타의 갱신은 일방적으로 $x_{bc} \leftarrow x_c$와 같이 하위 등급에서 상위 등급으로 방향이 정해지므로 하위 등급의 트랜잭션은 결코 상위 등급의 트랜잭션에 의해서 지연되지 않는다. 즉, 보안 2버전 2단계 로킹 스케줄러는 지연 보안의 성질을 만족한다. □

❖ 정리 3 본 논문에서 제안한 보안 2버전 2단계 로킹 스케줄링 방법은 회복 보안 성질을 만족한다.

(증 명)

상위 트랜잭션과 하위 트랜잭션이 가지는 데이타 자체가 분리가 되어 있으므로 하위 트랜잭션에 의해서 생기는 교착상태 자체에는 상위 트랜잭션이 관여해 있을 수가 없다. 즉 교착 상태의 발생시에 상위 트랜잭션을 다 제거하더라도 교착 상태는 없어지지 않는다. 즉, 하위 트랜잭션의 교착 상태는 상위 트랜잭션과 관계가 없다. 즉, 보안 2버전 2단계 로킹 스케줄러는 회복 보안의 성질을 만족한다. □

5. 기존의 보안 스케줄러와의 비교분석

제안하는 보안 2버전 2단계 로킹 스케줄러와 동일한 로크기반 스케줄러인 보안 2PL 스케줄러와 비교해 보면, 상위등급 트랜잭션의 판독이 지연되는 것을 해결하는 각각의 기법을 보면, 보안 2PL[7]은 가상로크(virtual lock), 의존적인 가상로크(dependency virtual lock)에 의한 지역공간 기록에 의해서 지연을 해결한다. 이것을 모조연산(fake operation)에 의한 해결이라 한다. 사실상, 실제동작을 지연시키고 또 다른 동작을 구상한 것으로 해결하고 있다. 이에 대하여 보안 2버전 2단계 로킹에서는 데이타의 복사를 통해 상위 트랜잭션을 위한 판독전용 데이타에 의해서 하위등급이나 상위등급의 트랜잭션이 결코 지연됨이 없이 트랜잭션의 보안평가 기준을 만족시킨다. 이러한 데이타 복사를 통한 해결은 더 많은 기억장소가 요구되지만 다중버전에 비해 작은 기억장소로 성능을 향상시키는 결과가 효과가 있다.

데이타의 신선도라는 측면을 서로 비교해 보면, 보안 2PL에서는 로킹기법의 기본적인 특성으로 인해 데이타가 신선하다. 그러나 보안 2버전 2단계 로킹보다 더 신선하지는 않다. 왜냐하면, 가상로크와 의존적인 가상로크의 지역공간 기록 때문이다. 지역공간 기록을 하고 실제기록을 하기 전에 접근되는 동작은 구버전의 데이타를 판독하기 때문에 보안 2버전 2단계 로킹기법보다 더 신선한 데이타 값을 갖게 되지는 않는다. 이에 대하여, 보안 2버전 2단계 로킹기법에서는 보안 2PL 만큼은 신선한 데이타를 갖는다. 이것은 보안 2버전 2단계 로킹기법이 제한된 버전의 데이타를 사용하기 때문이다.

다음은 보안 2PL과 보안 2버전 2단계 로킹의 신선도에 있어서의 비교이다.

아래의 예 6은 평이한 경우의 예로써 보안 2PL의 논문에 있는 것으로 비교에 적당한 예이다.

【예 6】 스케줄 비교

보안 2PL :

$T_1(S)$	R(x)		$R(v)C_1$
$T_2(U)$		$W(x)W(y)C_2$	

보안 2PL 스케줄[7] :

$rl_1(x)r_1(x)vwl_2(x)dvwl_2(y)C_2rl_1(y)\ \overline{r_1(y)}\ C_1\ ru_1(x)ru_1(y)wl_2(x)w_2(x)wu_2(x)wl_2(y)\ \overline{w_2(y)}wu_2(y)$

위의 스케줄에서 연산 $\overline{r_1(y)}$의 y는 $T_2(U)$가 갱신하기 전의 y의 값을 판독하게 된다.

보안 2버전 2단계 로킹 :

$T_1(S)$	R(x)		$R(v)C_1$
$T_2(U)$		$W(x)W(y)C_2$	

보안 2버전 2단계 로킹 스케줄 :

$rl_1(x_{bc})r_1(x_{bc})wl_2(x_w)w_2(x_w)wl_2(y_w)$ $\overline{w_2(y_w)}$ $C_2wu_2(x_w)wul(y_w)Cert_2(x)Cert_2(y)rl_1(y_{bc})$ $\overline{r_1(y_{bc})}$ $C_1ru_1(x_{bc})ru_1(y_{bc})C_1(x_{bc} \leftarrow x_c)(y_{bc} \leftarrow y_c)$

연산 $Cert_2(x)$는 $x_c \leftarrow x_w$를 수행하고, 연산 $Cert_2(y)$는 $y_c \leftarrow y_w$를 수행한다. 이와 같은 예는 적어도 보안 2버전 2단계 로킹은 보안 2PL의 신선도를 갖는다는 것을 보인다.

다음의 예에서는 앞에서 보였던 예 5에서와 같이 보안 2버전 2단계 로킹 기법이 보안 2PL보다 더 좋은 신선도를 갖는 예이다. 그리고 상위 트랜잭션의 성능이 좋아지는 예이다.

【예 7】

$T_1(TS)$					R(z)				C_1
$T_2(S)$						R(y)		C_2	
$T_3(S)$	R(x)			R(y)			C_3		
$T_4(U)$			W(x)W(y)		C_4				
$T_5(U)$		W(z)							C_5

위의 예에서 트랜잭션 T_2는 보안 2PL에서보다 더 신선한 값을 판독하게 되고 트랜잭션 T_1은 보안 2PL에서 보다 더 빠른 수행을 한다. 즉 트랜잭션 T_2는 데이타의 신선도가 향상이 되고 트랜잭션 T_1은 결코 블럭되지 않는다. 그러므로 더 나은 데이타의 신선도와 상위 등급 트랜잭션의 성능이 향상된다. 그리고 예 7에서 만약 트랜잭션 T_5가 아주 긴 수행시간을 갖는다면 트랜잭션 T_1은 아주 긴 시간 블럭이 될 수도 있다. 또한 이는 무결성(integrity)에 영향을 주는 것이라고 할 수 있다. □

성능향상에 대해서는 보안 2PL은 지역공간 기록을 할 뿐이고, 기본 2PL의 근원적인 충돌은 남아 있어서 보안 2PL의 검증을 한 다음, 기본 2PL에서 취소되는 스케줄의 트랜잭션은 보안 2PL에서도 취소된다. 즉 지연보안을 위해서 모조연산[7]을 두었을 뿐이고 성능향상의 부분은 고려되지 않았다. 그리고 보안 2PL은 로킹기법이 가지는 문제점에 대한 근본적인 해결은 어렵다. 그러나 보안 2버전 2단계 로킹에서는 상위 트랜잭션이 결코 지연되지 않을 뿐 아니라, 동일등급의 트랜잭션도 복사본 데이타를 판독해서 판독과 기록 사이의 지연을 판독과 확인 사이의 지연으로 변화하여 성능향상이 된다.

다중버전기법과 비교해 보면 보안 2버전 2단계 로킹기법보다 다중버전기법은 신선도가 떨어지고, 상당히 많은 기억공간이 필요하다. 보안 2버전 2단계 로킹은 제한된 버전의 데이타를 가지고 제한된 범위 내의 신선한 데이타를 접근하는 기법이다.

보안 2버전 2단계 로킹기법의 장점으로는 첫째, 보안 2PL처럼 모조연산이 없고, 그에 대한 나중 검증이 필요없다. 이는 전체 수행시간 감소라는 전체적인 성능향상을 가져온다. 둘째, 결코 상위등급 트랜잭션을 지연시키지 않는다. 세째, 동일등급 트랜잭션의 병행수행의 정도를 증가시킨다. 넷째, 구버전 데이타를 가지고 있으므로 이것을 회복하기 위한 자료로 사용하여 재실행, 취소를 용이하게 할 수 있다[2]. 다섯째, 제한된 버전 사용으로 기억장소를 적게 쓴다[2]. 여섯째, 보안을 만족시키면서 상위등급 트랜잭션의 성능이 향상된다. 다음으로 다중버전의 장시간 수행되는 트랜잭션이 판독전용의 트랜잭션일 경우 성능이 좋은 데 반해 이 기법은 판독전용이 아니고 판독과 기록이 혼합되어 있는 경우라도 규칙만 만족한다면 성능향상을 얻을 수 있다. 이러한 로크기반의 스케줄러는 효율적이며 실제 시스템에서 많이 사용되고 있다.

6. 성능 평가

본 논문에서 제안한 보안 2버전 2단계 로킹 스케줄러의 성능을 평가하기 위해서 SLAM II[13]를 사용하여 스케줄러의 알고리즘을 만들었다. 또한 기존에 제시되었던 보안 2단계 로킹 스케줄러의 알고리즘도 작성하여 비교를 위해 사용하도록 한다. 시뮬레이션 코드는 앞에서 제시하였던 여러 규칙을 프로그래밍화한 것으로 두 스케줄러의 처리량을 통해서 그 성능을 알아 보도록 한다.

시뮬레이션 모델은 공유메모리 방식의 단일 디스크 상에 있는 데이타베이스 시스템으로 가정한다. 데이타베이스의 단위는 페이지이고, 시스템은 디스크기반의 데이타베이스와 주기억장치 캐쉬로 구성이 된다. 만약 페이지가 트랜잭션이 요구한 데이타베이스의 페이지가 주기억장치 캐쉬에 존재하지 않는다면, 그 페이지는 디스크로 부터 읽어 와야 한다. 논리적인 모델은 중앙처리장치와 디스크라는 2가지 물리적인 장치를 가진다.

우선 트랜잭션 생성자가 트랜잭션을 판독전용, 기록전용, 판독과 기록의 혼용 트랜잭션을 발생시킨다. 트랜잭션의 길이를 불규칙하게 하므로 데이타베이스 내의 트랜잭션의 충돌을 만들어 그러한 충돌시에 나타나는 트랜잭션의 성능을 비교해 보고자 한다. 이렇게 하여 만들어진 트랜잭션은 큐에 들어가 수행을 위해서 대기하게 된다. 대기하는 트랜잭션들은 트랜잭션 스케줄러에 의해서 스케줄이 된다. 이렇게 해서 스케줄이 된 트랜잭션은 그 각각의 연산이 연산 스케줄러에 의해서 수행이 되고 충돌은 블러킹을 통해서 해결을 한다. 그러나 스케줄러가 보안을 고려하여야 하므로 블러킹이 되는 트랜잭션은 결코 자신보다 상위등급 트랜잭션에 의해서 블러킹이 되지는 않는다. 이때 이러한 보안 수행은 보안 2버전 2단계 로킹 알고리즘에 의해서 수행이 진행된다. 이렇게 되어 선택된 연산은 중앙처리장치를 점유하고 완료연산을 수행하기 전까지 계속된다.

매개변수, 성능 평가를 위한 자료 그리고 성능 평가 결과는 표 2와 같다. 여기에서 쓰여진 매개변수들은 기존에 성능 평가 자료로 쓰이던 것들이다. 이 매개변수는 제안된 보안 2버전 2단계 로킹 스케줄러의 성능 평가를 위한 시스템 환경을 제공한다. 보안 등급은 5등급으로 구성하여 등급간 영향을 고려하였다. 기억장치 버퍼에 원하는 데이타가 있거나 없을 확률을

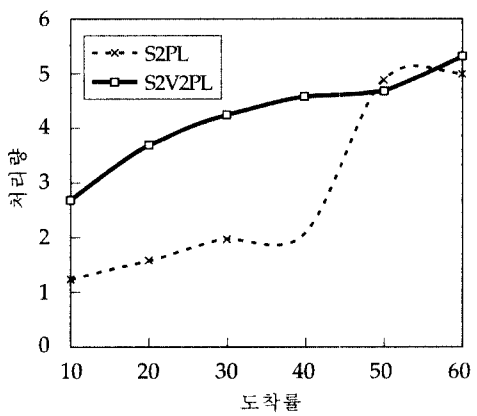
반반으로 구성하였다. 그리고 트랜잭션의 길이를 변화있게 해서 실상황에 더 가까운 성능평가가 되도록 하였다. 트랜잭션의 길이는 5 - 30개의 연산으로 구성되지만 트랜잭션이 매우 길 때에는 S2PL보다 보안 2버전 2단계 로킹 스케줄러가 성능면에서 더 유리하다.

데이타베이스 크기	500
CPU 연산 시간	15 msec
Disk 접근 시간	25 msec
Memory buffer에 페이지가 있을 확률	0.5
보안등급의 수	5

(a) System Resource Parameters

재시작 지연시간		20 msec
트랜잭션 크기	최소 크기	5
	최대 크기	30

(b) Workload Parameters



(c) 성능평가 결과

표 2 성능평가 변수와 결과

이러한 결과는 트랜잭션의 길이가 길고 충돌이 심한 결과도 이유가 되겠지만, 보안 2단계 로킹에서는 상위등급 트랜잭션이 자신보다 등급이 낮은 트랜잭션을 지연시키지 않기 위해서 가상로크와 지역기록으로 인한 지연과 상위등급 트랜잭션의 긴 지연에 있다. 트랜잭션의 길이가 불규칙하고 긴 것으로 인해 쉽게 트랜잭션이 종료되지 않으므로 계속해서 지연되는 트랜잭션은 시스템의 처리량을 떨어 뜨린다. 그러나 트랜잭션의 길이가 짧다면 지연이 되더라도 연속되어 지연이 되는 연산이 많지 않으므로 그 지연효과가 덜할 수 있을 것이다.

7. 결론과 연구방향

본 논문의 보안 2버전 2단계 로킹기법은 희사기법을 사용하는 것에 의해서 보안 2PL 기법에 비해 그 데이타의 신선도가 떨어지지 않고 동시에 성능향상을 얻을 수 있슴을 보였다. 비밀채널 제거를 위한 충돌제거는 호환성표에 근거해서 상위등급 판독전용 데이타를 두었다. 이 상위등급 판독전용 데이타는 상위 트랜잭션과 하위 트랜잭션과의 충돌을 피하고 성능향상까지 얻을 수 있었다. 보안 2PL기법에 비해 데이타의 신선도가 떨어지지 않으면서 성능향상을 얻은 것이다. 데이타를 희사하는 기법의 사용으로 보안 2PL보다 더 신선한 데이타를 읽게 하는 것이 가능했다. 이러한 장점은 간단한 스케줄의 일례로서 입증이 가능했다. 보안 2PL 기법의 데이타의 신선도라는 장점을 그대로 유지하면서 상위등급 판독전용 데이타를 유지하여 성능향상을 꾀한 것이다. 희사기법은 이미 사용되었고 종료할 때까지 사용하지 않을 데이타를 다른 트랜잭션에게 허락하는 방법을 사용했다. 희사기법은 동일등급의 트랜잭션의 성능향상에도 도움을 주는 것이다.

위에서는 이러한 속도나 신선도 등의 성능향상을 논리적으로 설명하고 예를 들어 설명하였으며, 이를 보다 정확히 실상황에서 관찰하기 위해 이에 대한 시뮬레이션을 수행하였다. 보안 2PL은 상위 트랜잭션 지연과 지역공간이라는 문제로 과부하가 문제되어 시스템 처리량을 떨어 뜨린다. 이에 비하여 제한된 버전 사용을 하는 보안 2버전 2단계 로킹 스케줄러는 블럭됨이 없이 보안 2PL보다 나은 성능을 가짐을 보인다. 데이타의 신선도도 예 5를 통해서 관찰한 것처럼 보안 2버전 2단계 로킹이 더 우수하다.

로킹에 관한 것으로 동일등급 간의 연산충돌로 인한 시간지연 효과를 해결하기 위해 데이타를 운영하는 기법을 더 깊이 생각하고, 성능의 관점에서 판독과 기록 연산의 개념을 연구할 것이다.

참 고 문 헌

[1] N. S. Barghouti and G. E. Kaiser, "Concurrency Control in Advance Database Applications", ACM Computing Surveys, 23:269-317, September 1991.

[2] R. L. C. Mohan and H. Pirahesh, "Efficent and Flexible Methods for Transient Versioning of Records to Avoid Locking by Read-Only Transactions", Technical Report, IBM Almaden Research Center, San Jose, CA 95120, U.S.A., 1992.

[3] H. Gracia-Molia, K. Salem and J. Shands, "Altruistic Locking", ACM Transactions on Database Systems, 19:117-165, March 1994.

[4] N. Goodman, P. A. Bernstein and V. Hadzilacos, Concurrency Control and Recovery in Database Systems, Addison-Wesley Publishing Company, Massachusetts, 1987.

[5] B. W. Lampson "A Note on the Confinement Problem", Communications of the ACM, vol. 16, No. 10, pp 613-615, Oct.. 1973.

[6] T. F. Keefe and W. T. Tsai, "Multiversion Concurrency Control for Multilevel Secure Database Systems", Proc. of the 10th IEEE Symposium on Research in Security and Privacy, May 1990, pp. 369-383.

[7] R. David and S. H. Son, "A Secure Two Phase Locking Protocol", Proc. of the 12th IEEE symposium on Reliable Distrubuted systems, October 1993.

[8] J. K. Millen, "Covert Channel Capacity", Proc. of the IEEE symposium on Research in Security and Privacy, April, 1987, pp. 60-66.

[9] C. Papadimitriou. The Theory of Database Concurrency Control, Rockville, MD, Computer Science Press, 1986.

[10] B. Claybrook, OLTP Online Transaction Processing Systems, John Wiley & Sons, Inc., 1992.

[11] S. Jajodia and V. Atluri, "Alternative Correctness Criteria for Concurrent Execution of Transactions in Multilevel Secure Database Systems", Proc. of the 12th IEEE Symposium on Research in Security and Privacy, May 1992, pp. 216-224.

[12] 오미나, 박 석, "보안 환경에서 트랜잭션 분류에 기초한 병행수행 제어", 통신정보보호학회 논문지 vol.4. No.1, JUN. 1994. pp. 59-71.

[13] A. Alan and B. Pritsker, Introduction to Simulation and SLAM II, 3rd edition, Systems Publishing corporation, 1986.

부 록 A

Altruistic Locking을 첨가하여 보안 2버전 2단계 로킹기법에 맞게 구성한 로크 알고리즘은 다음과 같다.

스케줄링에 필요한 리스트와 연산에 대한 스케줄러의 동작 :
rl(x) : 데이타 항목 x에 대해 판독로크를 갖는 트랜잭션의 집합.
wl(x) : 데이타 항목 x에 대해 기록로크를 갖는 트랜잭션의 집합.
cl(x) : 데이타 항목 x에 대해 확인로크를 갖는 트랜잭션의 집합.
d(x) : 데이타 항목 x를 희사한 트랜잭션의 집합.
T(x) : x를 가진 트랜잭션.
L(x) : x의 등급.

연산 p(x)가 입력되었을 때,

case 1. p가 read연산인 경우

```
if L(p) < L(x) then
  /* restricted *-property */
  p is refused
endif
if L(p) = L(x) then
if x is read-locked
 then
    if T(p) = T(x) then p(xw) endif
    if T(p) ≠ T(x) then p(xc) endif
endif
if x is write-locked
then
   if T(p) = T(x) then p(xw) endif
   if T(p) ≠ T(x) then p(xc) endif
endif
```

```
if x is certify-locked
then
 if cl(x) ⊂ d(x) then p(x)
 else p is delayed
 endif
 if x is unlocked
 then p(x) endif
endif
if L(p) > L(x)
then
   if x is read-locked then p(x_bc) endif
   if x is write-locked then p(x_bc) endif
   if x is certify-locked then p(x_bc) endif
   if x is unlocked then p(x_bc) endif
endif
```

case 2. 연산 p가 write일 경우

```
if L(p) < L(x) then
/* restricted *-property */
 p is refused
endif
if L(p) = L(x)
then
 if x is read-locked
 then
    if T(p) = T(x) then p(x_w) endif
    if T(p) ≠ T(x) then p(x_w) endif
endif
if x is write-locked
 then
  if T(p) = T(x) then p(x_w) endif
   if T(p) ≠ T(x) then p is delayed endif
endif
if x is certify-locked
 then
   if cl(x) ⊂ d(x) then p(x)
   else p is delayed
   endif
   if x is unlocked
   then p(x) endif
endif
if L(p) > L(x) then
  /* restricted *-property */
  p is refused
endif
```

case 3. p가 certify인 경우

```
if L(p) < L(x) then
/* 있을 수 없는 상황 */
 p is not occurred
endif
if L(p) = L(x)
then
if x is read-locked
then
   if T(p) = T(x) then p(x) endif
   if T(p) ≠ T(x) then
     if rl(x) ⊂ d(x) then
       p(x);
       x_bc ← x_w
     else p is delayed
     endif
   endif
endif
if x is write-locked
then
    if T(p) = T(x) then not occurred endif
    if T(p) ≠ T(x) then
       if cl(x) ⊂ d(x) then
            p(x);
            x_bc ← x_w
            else p is delayed
        endif
     endif
   endif
   if x is unlocked
   then
     p(x);
     if ((rl(x_bc) = ∅) or (rl(x_bc) ⊂ d(x_bc)))
     then
         x_bc ← x_w
    endif
  endif
endif
if L(p) > L(x)
```

```
   if T(p) = T(x) then p(x) endif
   if T(p) ≠ T(x) then p is delayed endif
endif
if x is certify-locked
then
   if T(p) = T(x) then p(x) endif
   if T(p) ≠ T(x) then p is delayed endif
endif
if x is certify-locked
then
   p is refused
endif
```

case 4. 연산 p가 commit일 경우

p를 수행;

if $rl(x_{bc}) = \varnothing$ or $(rl(x_{bc}) \subset d(x_{bc}))$

then

$x_{bc} \leftarrow x_c$;

endif

□ 著者紹介

유 진 호

1994년 광운대학교 전자계산학과(학사)
1994년 ~ 1996년 서강대학교 전자계산학과(석사)
1996년 ~ 현재 (주)LG 정보통신

박 석

1978년 서울대학교 계산통계학과 학사
1980년 한국과학기술원 전산학과 석사
1983년 한국과학기술원 전산학과 박사
1983년 ~ 현재 서강대 전자계산학과 교수
1989년 ~ 1991년 University of Virginia 방문교수
1992년 ~ 현재 한국정보과학회 데이타베이스 연구회 부위원장
1993년 ~ 1994년 한국정보과학회 논문지 편집위원
1995년 ~ 현재 통신정보보호학회 논문지 편집위원
1996년 ~ 현재 한국정보과학회지 편집위원

※ 주관심 분야 : 실시간 데이타베이스, 보안 데이타베이스, 주기억장치 데이타베이스, 멀티미디어 데이타베이스 시스템, 트랜잭션 관리, 병행수행 제어