보안 2버전 2단계 로킹 스케줄러[1]

유 진 호, 박 석
서강대학교 전자계산학과 데이타베이스 연구실

요 약

본 논문은 데이타베이스의 보안을 위해 비밀채널[Mil87]을 제거하는 기법의 연구로써 기존의 2버전 2단계 로킹에 보안을 고려한 새로운 기법을 제안한다. 즉, 상위레벨 트랜잭션 판독과 하위레벨 트랜잭션 기록 사이의 충돌을 피하고 동일레벨 간의 성능을 향상시키기 위한 방법으로 기존의 보안 2단계 로킹기법과는 달리 2개 버전의 데이타와 희사기법(donation)을 사용하여, 데이타의 신선도가 떨어지지 않으면서 성능의 향상을 얻을 수 있게 한다.

1 서 론

데이타베이스 시스템에서 트랜잭션의 스케줄링 시에 발생되는 비밀채널[Mil87]은 원하지 않는 데이타의 누출을 야기시키는 것으로써 시스템에 치명적인 피해를 준다. 따라서 상위레벨 트랜잭션 판독과 하위레벨 트랜잭션 기록 간의 충돌을 피하는 것이 필요하다. 본 연구에서는 충돌을 해결하기 위해서 기본적인 호환성표를 사용한다. 서로 다른 레벨의 연산 간의 충돌로 인한 비밀채널을 제거하는 방법으로 상위 트랜잭션만을 위한 판독전용 데이타를 둔다. 상위레벨 판독전용 데이타를 통해 상위 트랜잭션과 하위 트랜잭션간의 충돌을 피하며 성능향상까지 얻을 수 있다. 기존의 보안 2PL[Dav93]에 비해 데이타의 신선도가 떨어지지 않으면서 성능향상을 기대할 수 있다. 그리고 보안 2PL[Dav93]보다 더 신선한 데이타 판독을 위해서 트랜잭션이 사용하고 다시 사용하지 않을 데이타를 대여해 주는 방법을 고려한다. 보안 2PL의 데이타 신선도라는 장점을 유지하면서 상위 트랜잭션을 위한 판독전용 데이타를 유지하여 성능향상을 얻어 내고자 한다. 이러한 성능향상은 상위 트랜잭션의 판독과 하위 트랜잭션의 기록, 상위 트랜잭션의 판독과 하위 트랜잭션의 확인 사이의 연산에 관한 것이고, 동일 트랜잭션 사이의 연산을 위한 성능향상도 이미 사용한 후 종료까지 다시 접근하지 않을 데이타를 다른 트랜잭션에게 대여하는 방법을 사용한다. 아울러 뒷 부분에 보안 2PL[Dav93]과 다중버전기법과의 장점과 차이점을 분석한다.

2 보안 관련 연구

2.1 기본 개념

DBMS 운영 중에 발생할 수 있는 비밀정보의 흐름을 비밀채널이라고 한다. 비밀채널은 시스템이 내세운 보안정책을 위반하여 정보가 상위 보안등급에서 하위 보안등급으로 흘러가는 것을 허용한다. 비밀채널에는 저장장소채널과 시간채널 2가지가 있다. 이는 예 1과 같은 불법적인 정보의 흐름을 만드는 것이다. 이것을 막는 DBMS를 운영하기 위해서는 시스템이 가정하는 보안정책과 이에 맞는 보안모델이 있어야만 한다. 보안 시스템이 보안정책을 어떻게 시행하는지를 나타내는 것에 관한 추상적인 모델을 보안모델이라 하며 군사보안정책을 시행하는 가장 대표적인 모델은 벨-라파듈라 모델(Bell-Lapadula Model)이다. 이 논문에서는 이 모델을 기본으로 가정한다.

보안등급을 이루는 구성요소는 계층적등급(hierarchical level)과 범주집합(category set)이다. 계층적등급은 전체적으로 순서화되어 있고 범주집합은 부분적으로 순서화되어 있다. 보안등급은 L = (A, B)와 같이 정의된다. 여기서 A는 계층적등급으로 2급비밀, 3급비밀, 대외비, 평문 등으로 구성되고, B는 범주집합으로 인사, 작전, 군수, 정보 등으로 구성된다. 보안모델은 아래와 같은 제한 사항을 갖는다.

[1] 본 논문은 정보통신연구관리단의 "다단계 보안 데이타베이스 시스템 개발연구"과제의 일부로 수행 되었음

1) 트랜잭션 T는 다음의 조건을 만족할 때에만 데이타 항목 X에 대한 판독 연산이 가능하다.
조건 : 트랜잭션 T의 보안등급 L(T) ≥ L(X)
2) 트랜잭션 T는 다음의 조건을 만족할 때에만 데이타 항목 X에 대한 기록 연산이 가능하다.
조건 : 트랜잭션 T의 보안등급 L(T) = L(X)
여기서 1)은 벨-라파듈라 모델이고, 2)는 제한된 *-특성이다.

【예 1】 시간 비밀 채널

$T_1(S)$:R(X) C_{HL}
$T_2(U)$: 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 지연이 되고 이에 대해서 Cert(X)연산을 수행한 $T_2(U)$는 $T_1(S)$의 신호를 받게 된다. 이러한 신호는 비트정보를 전달할 수가 있고 대량의 연산을 통해서 대량의 비트정보가 전달되어 메시지 전달효과를 가져올 수가 있다. 이렇듯 정보가 전송되는 효과를 가진 자료흐름을 비밀채널이라고 한다.

2.2 기존 보안 스케줄러

비밀채널[Mil87] 방지를 위해서 Keefe[Keef90]나 Jajodia[Jaj92]가 다중버전 보안 알고리즘에 대한 연구를 하였다. Keefe의 알고리즘에는 구버전 판독의 문제가 있었고, Jajodia의 연구에서는 상위레벨 트랜잭션의 불합리한 블러킹의 문제가 있었다. 본 연구실에서 발표한 논문[박석94]인 "보안환경에서 트랜잭션 분류에 기초한 병행수행 제어"에서는 이러한 문제를 하위 보안등급을 갖는 데이타를 판독한 상위레벨 트랜잭션이 유도갱신이거나 판독전용일 때는 블럭될 필요가 없으므로 블럭시키지 않고, 순수갱신이거나 기록전용일 경우에도 이미 기록을 수행하였다면 블럭되지 않는다. 이렇게 하여 구버전 판독의 단점을 없애면서도 전체적으로 트랜잭션의 병행수행의 정도를 높인 연구를 수행 발표하였다.

로크기반 스케줄러에 대한 연구인 보안 2PL[Dav93]은 스케줄러가 스케줄링을 할 때 만들어 질 수 있는 비밀채널을 막는 기법을 연구한 것이다. 비밀채널을 지연보안의 관점에서 접근한다. 상위레벨과 하위레벨 간의 충돌을 해결하기 위해서 몇 개의 부가적인 리스트와 판독과 기록 연산을 트랜잭션 지역공간에 하는 것에 의해서 지연보안의 문제를 해결한다. 하지만, 보안 2PL의 장점인 데이타의 신선도가 이러한 지역공간에 연루된 연산때문에 나빠진다. 이러한 지역공간에 연루된 연산을 하면 데이타의 일관성에는 큰 문제가 될 수 있다. 그러나 보안 2PL이 동적인 스케줄러로써 구성되었다는 것은 장점이라 할 수가 있다.

3 보안 2버전 2단계 로킹 스케줄러

기존의 보안 2PL[Dav93]은 다중버전에 비해서 적은 기억장소를 사용하면서 데이타의 신선도가 앞선다는 것이 장점이었다. 보안 2PL에서는 상위 트랜잭션과 하위 트랜잭션의 충돌을 해결하기 위해서 지역기록(local write)을 사용하였다. 2PL은 로크기반의 스케줄러이기 때문에 기본적으로 데이타의 신선도가 우수하다. 그러나 지역기록의 사용으로 인하여 실제기록(real write)이 지연되어 사실상은 기본 2PL만큼의 신선한 데이타를 판독하지는 못한다. 보안 2PL은 가상적인 기록과 판독이 있어서 성능적인 면이 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 고안한다.

지연의 문제에 있어서 판독과 기록 간의 지연을 기존의 2버전 2단계 로킹기법의 판독과 확인 간의 지연으로 변형하면 지연시간을 줄일 수 있다[Cray92]. 2버전 2단계 로킹기법의 사용을 통해서 판독과 기록 간의 충돌과 지연시간으로 인한 부하의 해결이 쉬워진다. 그리고 상위 트랜잭션의 판독요구와의 충돌을 피하기 위해서 상위레벨 판독전용 데이타를 둔다. 그리고 동일한 등급의 두 버전 중 한 개의 버전은 복사본 데이타이다. 이러한 지연의 문제와 충돌로 인한 비밀채널을 고려하여 보안 2버전 2단계 로킹을 생각한다.

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

3.1 스케줄러 모델

2버전 2단계 로킹에서는 각 데이타 항목에 대해서 많아야 2개의 버젼이 존재한다. 데이타 항목이 갱신될 때, 복사본 데이타(shadow)는 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 이 데이타에 대한 판독만을 허용하여 상위 트랜잭션의 판독과 하위 트랜잭션의 확인으로 인해 발생하는 비밀채널을 막는다. 두 트랜잭션 간의 판독과 기록의 충돌로 인한 성능저하를 해결하기 위해 2버전 2단계 로킹[Good87]을 생각한다. 왜냐하면 트랜잭션 간의 판독과 기록간의 충돌을 판독과 확인 간의 충돌로 변화시키면 지연 효과가 덜하기 때문이다[Cray92].

기본적으로 충돌의 제거는 호환성표에 근거하며, 상위 트랜잭션에게 신선한 데이타를 판독하게 하고 트랜잭션의 병행수행 정도를 높이기 위해 희사기법(donation)을 도입한다. 본 논문에서는 보안 수행을 위한 상위레벨 판독전용 데이타의 관리와 희사 기법에 촛점을 둘 것이다.

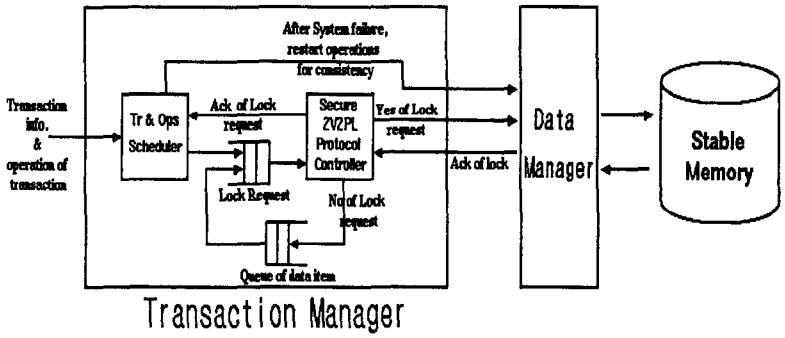

그림 1 스케줄러 모델

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

3.2 보안 2버전 2단계 로킹

제안하는 보안 2버전 2단계 로킹에서 데이타는 동일레벨에 대해 복사본 데이타와 현재 수행 중인 데이타를 갖고 상위레벨에 대해서는 상위레벨 판독전용 데이타를 갖는다고 가정한다. 이 상위레벨 판독전용 데이타는 상위레벨의 판독로크와 하위레벨의 확인로크에서의 충돌을 해결하기 위한 것으로 이 데이타에 의해서 판독이 있는 상위레벨의 트랜잭션은 하위레벨의 기록이나 확인에 의해서 지연되거나 결코 취소되지 않는다. 주어진 시간에 대하여 데이타 항목은 다음 3가지 상태 중 하나이다.

- 상태 1. 데이타 항목은 두개의 데이타 값을 갖는다
- 상태 2. 데이타 항목은 두개의 데이타 값을 갖는다. 트랜잭션은 이 값을 읽을 수 있고, 다른 하나의 트랜잭션은 동시에 데이타 항목에 대한 새로운 버전을 만들고 있다.
- 상태 3. 데이타 항목은 세개의 값을 갖는다 - 복사본 데이타 값, 새로운 버전의 값과 상위레벨 판독전용 데이타.

상위레벨 트랜잭션의 판독요구는 결코 지연되지 않는 반면, 스케줄러는 새로운 값이 항상 사용가능하지는 않기 때문에 판독로크을 필요로 한다. 그리고 여전히 그 복사본 데이타 값을 읽는 트랜잭션이 있는지 없는지를 알아야만 한다.

	Read	Write	Certify
Read	yes	yes	no+
Write	yes	no	no
Certify	no+	no	no

표 1 2버전 2단계 로킹의 호환성표

2버전 2단계 로킹의 호환성표는 표 1 같이 테이블을 구성하며 no+에서 지연보안에 의한 비밀채널을 생성할 위험이 있다. 상위 트랜잭션은 하위의 데이타 항목에 대해 판독동작만을 하기때문에 생각해야 할 것은 상위 트랜잭션의 판독과 하위 트랜잭션의 확인과의 충돌이다. 이것은 트랜잭션이 한 데이타 항목에 관하여 2개의 데이타와 상위레벨 판독전용 데이타가 있다고 하면, 비밀채널이 없는 호환성표를 구성할 수 있다. 상위레벨 트랜잭션은 하나의 데이타 항목에 대해서 2개의 데이타와 상위레벨 판독전용 데이타를 가져서 비동기적인 판독을 하므로 상위 트랜잭션은 결코 지연되지 않는다. 상위 트랜잭션 판독전용 데이타에 대한 가정으로 인해서 상위레벨 판독로크와 하위레벨 확인 로크사이의 충돌을 해결하여 지연보안의 문제는 해결이 된다.

3.3 보안 2버전 2단계 로킹 프로토콜

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

보안 2버전 2단계 로킹이 사용되므로 데이타는 상위레벨 판독전용 데이타(X_{bc}), 복사본 데이타(X_c), 사용중인 데이타(X_w)로 분류된다. 아래의 몇가지 경우를 통해서 보안 2버전 2단계 로킹기법을 완성하도록 한다.

CASE 1. 상위레벨 판독과 하위레벨 확인으로 인한 비밀채널의 제거

$T_1(S)$: $R(X_{bc})$ C_1

$T_2(U)$: $W(X_w)$ $Cert_2(X)$ C_2

기존의 $Cert_2(X)$의 동작은 $X_c \leftarrow X_w$이고, 제안하는 $Cert_2(X)$의 동작은 $X_c \leftarrow X_w$이다. C_1이 완료한 후 $X_{bc} \leftarrow X_c$를 해서 비동기적으로 확인을 지연시킨다. 여기서 X_{bc}는 상위레벨 판독전용 데이타로서 상위 트랜잭션이 하위 트랜잭션과의 충돌에 의해 지연됨이 없이 하나의 구버전을 읽는 것을 허락하기 위해 마련된 데이타 항목이다. X_{bc}는 하위레벨의 트랜잭션에 의해서 로크가 걸려있는 데이타를 접근하는 상위 트랜잭션의 충돌을 없애기 위해서 마련된 것이다. $T_1(S)$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.

CASE 2. 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.

CASE 3. 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 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). 위의 스케줄로 부터 프로토콜을 위한 규칙을 정의할 수가 있다.

보안 2버전 2단계 로킹의 규칙

규칙 1. X_{bc}는 상위레벨 트랜잭션의 판독을 위한 것이다. X_c는 확인된 데이타이다. X_{bc}는 갱신되는 시점이 중요하며, X_{bc}는 상위레벨 트랜잭션으로 하여 한단계 구버전을 갖는다. X_c는 동일레벨 간의 데이타를 처리하기 위한 것이다.

규칙 2.

$T_1(S)$: $R(X)$ C_{HL}

$T_2(U)$: $Cert(X)$

위의 예에서는 C_{HL}의 시점에서 $X_{bc} \leftarrow X_c$이 수행이 된다. 즉, 현재 수행 중인 상위레벨 트랜잭션 판독집합에 포함되지 않으면, 그 시점에서 $X_{bc} \leftarrow X_c$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 갱신되지 않은 x_c의 값을 가지고 있으므로 사이클이 생성되지 않는다.

4 동일 레벨의 성능향상을 위한 방안

2PL을 확장시킨 Altruistic Locking은 트랜잭션이 한 번 접근한 후에 더 이상 쓰지않는 데이타는 다른 트랜잭션에게 쓸 수 있도록 하는 기법을 사용한다. 이러한 기법은 Altruistic Locking[Gar94]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(donated) 데이타 항목을 사용하려는 트랜잭션은 일정한 조건이 만족이 되면 그 희사된 데이타 항목을 사용할 수 있다[Gar94]. 이러한 조건을 로킹 스케줄러에 반영하여 로킹에 대한 동작을 수정하도록 한다. 또한 이 기법은 자원을 사용하지 않으면서 오래 점유하는 트

랜잭션이 존재할 경우 특히 유용하며, 기존에 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 회사하여 다른 트랜잭션에게 데이타를 사용할 수 있게 하기 때문이다. 즉, 직렬화순서를 지키면서 자원 사용의 효율을 기하여 성능을 향상시킨다.

4.1 Altruistic Locking Protocol

Altruistic Locking의 대상이 되는 스케줄은 그림 2와 같다. 그림 2와 같이 트랜잭션 T_1이 수행중에 있고 그 수행 도중에 트랜잭션 T_2, T_3, T_4가 데이타의 사용을 요구한다고 하자. 기존의 방법에 의하면 트랜잭션 T_4만이 허락될 수 있다. 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 병렬성을 높여 성능향상을 꾀할 수 있다.

A, B : T2에 의해 사용이 요구되는 데이타
C, E : T3에 의해 사용이 요구되는 데이타
F, G : T4에 의해 사용이 요구되는 데이타
A, B, C : T1에 의해 이미 사용된 데이타
D : T1에 의해 현재 사용되는 데이타
E, F, G : T1이 나중에 사용할 데이타

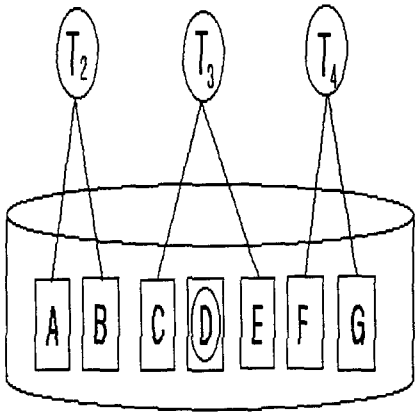

그림 2 Altruistic Locking이 병행성을 확장시키는 예

Altruistic Locking 규칙을 수행하는 프로토콜을 설계하는 것은 그리 어렵지 않다. 기존의 2PL에 몇가지 회사(donation)에 대한 자료구조와 동작을 부가하면 된다. 각각의 데이타베이스 객체 a에 대하여, 프로토콜은 다음의 두가지 집합을 유지한다: l(a), 데이타베이스 객체 a를 로크한 트랜잭션의 집합과 d(a), 데이타베이스 객체 a를 회사한(donated) 트랜잭션의 집합.

트랜잭션 T_2가 트랜잭션 T_1에 의해 회사된 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[Gar94]은 다음과 같다.
첫째는 트랜잭션 중 하나의 트랜잭션이 먼저 객체를 회사하지않으면, 같은 객체 상에 동시에 로크를 가질 수 없다. 둘째는 트랜잭션 T_a가 또 다른 트랜잭션 T_b에 대하여 충돌관계(indebt)에 있다면, 그것은 T_b가 그것의 로크해제를 수행할 때 까지 완전하게 T_b의 포함상태(wake상태)에 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 상위레벨 트랜잭션의 판독을 위한 것이다. X_c는 확인된 데이타이다. X_{bc}는 갱신되는 시점이 중요하며, X_{bc}는 상위레벨 트랜잭션으로 하여 한단계 구버전을 읽도록 한다. X_c는 동일레벨 간의 데이타를 처리하기 위한 것이다.

규칙 2.

$T_1(S)$:R(X) C_{HL}

$T_2(U)$: Cert(X)

위의 예에서는 C_{HL}의 시점에서 $X_{bc} \leftarrow X_c$이 수행이 된다.

즉, 현재 수행 중인 상위레벨 트랜잭션 판독집합에 포함되지 않거나 희사되어 있으면, 그 시점에서 $X_{bc} \leftarrow X_c$를 수행한다.

규칙 3.

트랜잭션 중 하나의 트랜잭션이 먼저 객체를 희사하지않으면, 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 포함상태(wake상태)에 있어야만 한다.

스케줄러는 받아들여진 연산이 판독, 기록, 확인, 완료인지를 판단하여 각각에 해당하는 레벨별로 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 희사동작이 주는 영향

다음과 같은 스케줄이 있다고 가정하자.

$T_1(TS)$:$R(y_{bc})$ $R(x_{bc})$ C_1

$T_2(S)$: $W(y_w)R(x_{bc})Cert(y)C_2$

$T_3(U)$: $W(x_w)$ $Cert(x)$ C_3

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

$T_1(TS)$:$R(y_{bc})d(y)$ $R(x_{bc})$ C_1

$T_2(S)$: $W(y_w)R(x_{bc})Cert(y)C_2$

$T_3(U)$: $W(x_w)$ $Cert(x)$ C_3

데이타 항목 y는 희사동작 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. 상위레벨의 트랜잭션이 데이타 항목을 희사하는 것은 그 데이타에 대한 확인동작($x_{bc} \leftarrow x_c$)을 좀 더 일찍 일어나게 한다. 자신 이상의 높은레벨 트랜잭션에게 좀더 신선한 데이타를 판독하게 하고, 다시 사용치 않는 데이타를 대여해 주어서 상호간의 병행성의 정도를 증가시킨다. 희사된 데이타 항목은 보안 2PL보다 더 우수한 데이타의 신선도를 예 2에서 처럼 가질 수 있다. 데이타가 일단 희사가 되고 나면 그때 부터는 그 데이타를 사용가능하기 때문에 이로 인한 데이타 신선도 뿐만아니라 병행성도 좋아진다.

【예 2】 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 데이타를 Commit연산에서 X_{bc}를 갱신해 주면 그 다음 연산인 $r(y_{bc})$는 갱신된 데이타(y_{bc})를 판독하여 더 신선한 데이타를 판독하게 된다. $Cert_3(x)$에서는 $x_c \leftarrow x_w$이 수행되고, $Cert_3(y)$에서는 $y_c \leftarrow y_w$이 수행된다. C_3에서는 $x_{bc} \leftarrow x_c$, $y_{bc} \leftarrow y_c$이 수행된다.

5 기존 보안 스케줄러와의 비교분석

제안하는 보안 2버전 2단계 스케줄러와 동일한 로크기반 스케줄러인 보안 2PL 스케줄러와 비교해 보면, 상위레벨 트랜잭션의 판독이 지연되는 것을 해결하는 각각의 기법을 보면, 보안 2PL[Dav93]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 상위 트랜잭션을 위한 판독전용 데이타에 의해서 하위레벨이나 상위레벨의 트랜잭션이 결코 지연됨이 없이 트랜잭션의 보안평가 기준을 만족시킨다. 즉, 데이타의 복사를 통한 해결을 한다.

데이타의 신선도라는 측면을 서로 비교해 보면, 보안 2PL에서는 로킹기법의 기본적인 특성으로 인해 데이타가 신선하다. 그러나 보안 2버전 2단계 로킹보다 더 신선하지는 않다. 왜냐하면, 가상로크(virtual lock)와 의존적인 가상로크(dependency virtual lock)의 지역공간 기록 때문이다. 지역공간 기록을 하고 실제 기록을 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 3은 평이한 경우의 예로써 보안 2PL의 논문에 있는 것으로 비교에 적당한 예이다.

【예 3】 스케줄 비교

보안 2PL :

$T_1(S)$: $r(x)$ $r(y)C_1$

$T_2(U)$: $w(x)$ $w(y)$ C_2

보안 2PL 스케줄[Dav93] :

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

$\overline{w_2(y)}\ wu_2(y)$

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

보안 2버전 2단계 로킹 :

$T_1(S)$: $r(x)$ $r(y)C_1$

$T_2(U)$: $w(x)$ $w(y)$ C_2

보안 2버전 2단계 로킹 스케줄 :

$rl_1(x_{bc})r_1(x_{bc})wl_2(x_w)w_2(x_w)wl_2(y_w)\ \overline{w_2(y_w)}\ C_2wu_2(x_w)wu_1(y_w)Cert_2(x)Cert_2(y)rl_1(y_{bc})$

$\overline{r_1(y_{bc})}\ C_1ru_1(x_{bc})ru_1(y_{bc})C_1(x_{bc} \leftarrow x_c)(y_{bc} \leftarrow y_c)$

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.

성능향상에 대해서는 보안 2PL은 지역공간 기록을 할 뿐이고, 기본 2PL의 근원적인 충돌은 남아 있어서 보안 2PL의 검증(validation)을 한 다음, 기본 2PL에서 취소되는 스케줄의 트랜잭션은 보안 2PL에서도 취소된다. 즉 지연보안을 위해서 모조연산(fake operation)[Dav93]을 두었을 뿐이고 성능향상의 부분은 고려되지 않았다. 그리고 보안 2PL은 로킹기법이 갖는 제한의 근본적인 해결은 어렵다. 그러나 보안 2버전 2단계 로킹에서는 상위 트랜잭션이 결코 지연되지 않을 뿐 아니라, 동일레벨의 트랜잭션도 복사본 데이타를 판독해서 판독과 기록 사이의 지연을 판독과 확인 사이의 지연으로 변화하여 성능향상이 된다.

보안 2PL에서 데이타의 등급별 분류는 고려하지 않았다(트랜잭션의 등급별 분류만 구별함). 그러나 보안 2버전 2단계 로킹에서는 이러한 데이타의 등급별 분류를 고려하여 편성해야 한다고 생각한다.

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 모조연산(fake operation)이 없고, 그에 대한 나중 검증(validation)이 필요없다. 이는 전체 수행시간 감소라는 전체적인 성능향상을 가져온다. 둘째, 결코 상위레벨 트랜잭션을 지연시키지 않는다. 세째, 동일레벨 트랜잭션의 병행수행의 정도를 증가시킨다. 넷째, 또 다른 데이타를 더 가지고 있으므로 이것을 회복하기 위한 자료로 사용하여 재실행(redo), 취소(undo)를 용이하게 할 수 있다[Moh92]. 다섯째, 제한된 버전 사용으로 기억장소를 적게 쓴다[Moh92]. 여섯째, 보안을 만족시키면서 상위레벨 트랜잭션의 성능이 향상된다. 다음으로 다중버전의 장시간 수행되는 트랜잭션이 판독전용의 트랜잭션일 경우 성능이 좋은 데 반해 이 기법은 판독전용이 아니고 판독과 기록이 혼합되어 있는 경우라도 제한 사항만 만족한다면 성능향상을 얻을 수 있다. 이러한 로크기반의 스케줄러는 효율적이며 실제 시스템에서 많이 사용되고 있다.

6 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 있음을 보였다. 비밀채널 제거를 위한 충돌제거는 호환성표에 근거해서 데이타의 복사본인 상위레벨 판독전용 데이타를 두었다. 이 상위레벨 판독전용 데이타는 상위 트랜잭션과 하위 트랜잭션과의 충돌을 피하고 성능향상까지 얻을 수 있었다. 보안 2PL기법에 비해 그 데이타의 신선도가 떨어지지 않으면서 성능향상을 얻은 것이다. 데이타를 희사하는 기법의 사용으로 보안 2PL[Dav93]보다 더 신선한 데이타를 읽게 하는 것이 가능했다. 이러한 장점은 간단한 스케줄의 일례로서 입증이 가능했다. 보안 2PL 기법의 데이타의 신선도라는 장점을 그대로 유지하면서 상위레벨 판독전용 데이타를 유지하여 성능향상을 꾀한 것이다. 희사기법은 이미 사용되었고 종료할 때까지 사용하지 않을 데이타를 다른 트랜잭션에게 허락하는 방법을 사용했다. 희사기법은 동일레벨의 트랜잭션의 성능향상에도 도움을 주는 것이다.

로킹에 관한 것으로 동일레벨 간의 연산충돌로 인한 시간지연효과를 해결하기 위해 데이타를 운영하는 기법을 더 깊이 생각하고, 성능의 관점에서 판독과 기록 연산의 개념을 연구할 것이다. 위에서는 이러한 속도나 신선도 등의 성능향상을 논리적으로 설명하고 예를 들어 설명하였으나, 이를 보다 정확히 실상황에서 측정하기 위해 이에 대한 시뮬레이션을 수행 중이다.

참고문헌

[Bar91] N. S. Barghouti and G. E. Kaiser, "Concurrency Control in Advance Database Applications", ACM Computing Surveys, 23:269-317, September 1991.

[Moh92] 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.

[Gar94] H. Gracia-Molia, K. Salem and J. Shands, "Altruistic Locking", ACM Transactions on Database Systems, 19:117-165, March 1994.

[Good87] N. Goodman, P. A. Bernstein and V. Hadzilacos, Concurrency Control and Recovery in Database Systems, Addison-Wesley Publishing Company, Massachusetts, 1987.

[Keef90] 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.

[Dav93] R. David and S. H. Son, "A Secure Two Phase Locking Protocol", Proc. of the 12th IEEE symposium on Reliable Distrubuted systems, October 1993.

[Mil87] J. K. Millen, "Covert Channel Capacity", Proc. of the IEEE symposium on Research in Security and Privacy, April, 1987, pp. 60-66.

[Cray92] B. Claybrook, OLTP Online Transaction Processing Systems, John Wiley & Sons, Inc., 1992.

[Jaj92] 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.

[박석94] 오미나, 박 석, "보안 환경에서 트랜잭션 분류에 기초한 병행수행 제어", 통신정보보호학회 논문지 vol. 4. No. 1, JUN. 1994. pp. 59-71.