클라이언트-서버 환경에서 공간 데이터 변경 트랜잭션을 위한 회복 기법의 설계 및 구현

박 재관[O*] 김 동현[*] 최 진오[**] 홍 봉희[*]

[*]부산대학교 컴퓨터공학과, [**]부산외국어대학교 컴퓨터공학과

Design and Implementation of Recovery Scheme for Update Transactions of Spatial Data in Client-Server Environment

Jae Kwan Park[O*] Dong Hyun Kim[*] Jin oh Choi[**] Bong Hee Hong[*]

[*]Dept. of Computer Engineering, Pusan National University

[**]Dept. of Computer Engineering, Pusan University of Foreign Studies

요 약

공간 데이타를 클라이언트에서 수정하는 클라이언트-서버 환경에서는 동시성 지원을 위해 클라이언트 트랜잭션들이 협동 작업을 수행하여야 한다. 이 때, 철회를 요구하는 트랜잭션의 회복을 제어하기 위해서는 협동 작업에 참여한 종속 트랜잭션들을 함께 고려하여야 하므로 기존의 회복 기법으로 지원되지 않는 문제가 있다.

이 논문에서는 클라이언트-서버 GIS에서 공간 관련성을 고려한 트랜잭션 회복 기법을 위해 부분 철회 기법과 회복 제어 알고리즘 및 프로토콜을 설계하고 구현한 결과를 보인다.

1. 서론

클라이언트-서버 환경에서 다수 클라이언트가 공간 데이터 변경 작업을 동시에 수행할 때, 공간 관련성을 고려한 협동 트랜잭션 기법을 이용하여 작업의 동시성을 향상시킬 수 있다[4]. 그러나 각 클라이언트는 협동 작업으로 수정한 결과를 여러 가지 이유로 철회할 경우가 발생한다. 이 때, 공동 작업에 참여한 다른 트랜잭션들은 철회하기 전 데이터를 바탕으로 변경 작업을 수행하였으므로 함께 철회 대상으로 고려되어야 한다. 이것은 기존의 회복 기법으로는 처리할 수 없기 때문에 새로운 회복 기법이 요구된다.

관계형 데이터베이스에서의 트랜잭션 회복 기법[1,2]은 로그와 체크포인트를 이용하여 'UNDO'와 'REDO'대상 트랜잭션을 찾아 보상 트랜잭션을 실행시킴으로써 해결한다. 그러나 공간 데이터는 동시 수정하는 트랜잭션들간에 공간 관련성에 의한 종속성이 존재하므로 한 트랜잭션의 철회는 다른 트랜잭션들의 철회를 요구하게 된다. 이것은, 협동 트랜잭션들이 완화된 잠금 기법을 이용하여 동일한 영역을 동시에 수정하기 때문에 발생한다.

이 문제를 다룬 협동 수정 트랜잭션의 회복 기법[3]은 지도의 오류를 방지하기 위해 수정 영역이 겹치는 모든 트랜잭션을 철회한다. 그러나, 이 기법은 불필요한 철회가 연쇄적으로 발생하는 단점이 있다. 이 문제를 해결하기 위하여 이 논문에서는 공간 관련성에 의한 트랜잭션간 종속성 유형을 정의하고, 각 종속성 유형에 따른 회복 기법을 제시하였다. 그리고 mid-rollback에 의한 부분 철회 기법과 회복 제어 알고리즘 및 프로토콜을 구현하여 공간 데이터 변경 트랜잭션을 위한 회복 기법을 구현하였다.

이 논문의 구성은 다음과 같다. 먼저 2장에서는 관련 연구를 기술하고 3장에서는 공간 데이터 변경 트랜잭션의 회복 메카니즘을 설명한다. 4장에서는 회복 프로토콜과 예제를 설명하며 5장에서는 구현 내용을 기술한다. 마지막으로 6장에서는 결론 및 향후 연구를 기술한다.

2. 관련 연구

관계형 데이터베이스에서는 긴 트랜잭션을 서브-트랜잭션으로 나누고 부분 철회로 회복 비용을 줄이는 방안이 제시되었다[1,2]. 기존의 단일 트랜잭션에서의 회복 기법을 서브-트랜잭션에 적용하기 위해서 변형된 로그-기반 철회(Log-Based Rollback) 알고리즘을 제시하였다. 그리고 서브-트랜잭션의 특징을 정의하고 제시된 트랜잭션의 회복 기법에 대한 무결성을 확인하였다. 그러나 이러한 기법은 공간 데이터를 동시 수정하는 환경에 적용하면 공간 관련성에 의한 트랜잭션간 종속성을 제어할 수 없으므로 데이터의 무결성을 보장할 수 없다.

[3]은 지도 수정에서 협동 작업을 하는 모든 클라이언트들의 작업을 부분 철회함으로써 수정 오류의 여지를 제거하는 방법을 제시하였다. 그러나 이 방법은 철회하지 않아도 수정 오류가 전혀 발생하지 않는 서브-트랜잭션들까지 철회를 강요하는 문제가 있다. 또한 불필요한 철회가 점차 수정 작업 중인 여러 사이트로 전파되는 문제가 있다. 따라서 반드시 철회되어야 할 트랜잭션만을 찾아내는 기법이 필요하다.

3. 공간 데이터 변경 트랜잭션의 회복메카니즘

이 장에서는 공간 관련성에 의해 발생하는 트랜잭션간의 종속성, mid-rollback, 그리고 cascading mid-rollback의 개념에 대해 설명한다. 또한 이 개념을 이용하여 이 논문에서 제시하는 트랜잭션 회복 제어 방법을 설명한다.

3.1 공간 관련성 기반 트랜잭션 종속성

공간 데이터 변경을 위해 협동 작업을 수행할 때, 무결성을 위해서는 공간 관련성을 고려해야 한다[3]. 서로 다른 두 트랜잭션이 공간 관련성을 가진 서로 다른 두 공간 데이터를 변경하는 경우에 공간 관련성을 유지하기 위해 작업은 둘 다 완료하거나 둘 다 취소하는 단일

작업으로 실행되어야 한다. 이것을 공간 관련성 기반 트랜잭션 종속성(이하 트랜잭션 종속성)이라고 한다.

3.2 mid-rollback과 cascading mid-rollback

mid-rollback은 rollback의 고비용 문제를 해결하기 위해 공간 데이터 변경 트랜잭션에서 사용될 수 있는 회복의 최소 단위이며 mid-commit의 취소를 의미한다[5]. 예를 들어 클라이언트 C_1에서 수행되는 트랜잭션 T_1이 있을 때, T_1의 서브-트랜잭션 T_1^a가 객체 obj_a를 변경하였으나, 작업의 오류로 인해 이를 철회할 수 있는데 이것이 mid-rollback이다.

mid-rollback은 공간 데이터를 수정하는 서브-트랜잭션 간의 트랜잭션 종속성 때문에 cascading mid-rollback을 발생시킨다. 서로 다른 두 클라이언트 C_1, C_2에서 수행되는 서브-트랜잭션을 T_1^a, T_2^b라고 하고 두 트랜잭션 사이에 트랜잭션 종속성이 존재하는 경우에 T_1^a 또는 T_2^b의 mid-rollback은 T_2^b 또는 T_1^a의 mid-rollback을 발생시키는데 이것이 cascading mid-rollback이다.

3.3 트랜잭션의 회복 제어

지도 수정 작업은 사용자의 상호 작용이 필요하므로 많은 시간이 걸린다. 따라서 작업의 철회는 고 비용의 재 수정을 요구한다. 지도 수정 작업의 이러한 특징 때문에 회복 기법은 철회의 발생을 줄일 수 있어야 한다. 이 논문에서 제시하는 회복 기법은 트랜잭션간의 관계를 유형별로 나누어 연쇄적으로 발생하는 철회를 줄인다.

한 트랜잭션에서 mid-rollback이 발생하면 수행 중인 타 협동 트랜잭션들은 트랜잭션 종속성 여부와 서브-트랜잭션의 완료 여부에 따라 redo, undo delta 그리고 cascading mid-rollback 중 하나의 작업을 수행한다. redo는 현재 수정 작업 중인 서브-트랜잭션의 재 수정을 의미하며 undo delta는 현재 수정 작업은 계속 유지하면서 delta만 변경한다.

[표 1] mid-rollback 실행 전 유형별 행동 제어

T_1^a \ T_2^b	SR_{before}							
mid-rollback	종속				독립			
	SR_{short}		SR_{after1}		SR_{after1}		SR_{after2}	
	종속	독립	종속	독립	종속	독립	종속	독립
	Redo	Redo	*mid-rollback*	Undo Delta	Redo	Undo Delta	*mid-rollback*	Undo Delta

[표 2] mid-rollback 실행 후 유형별 행동 제어

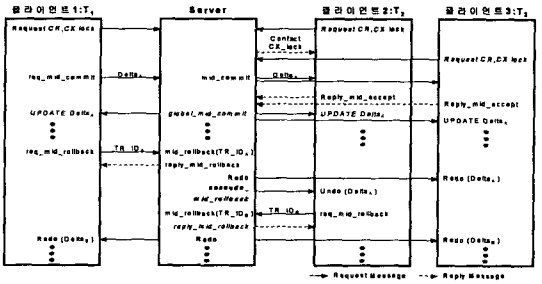
T_1^a \ T_2^b	$SR_{rollback1}$		$SR_{rollback2}$	
mid-rollback	종속	독립	종속	독립
	Redo	Undo Delta	*mid-rollback*	Undo Delta

[표 1, 2]는 서브-트랜잭션 T_1^a의 mid-rollback에 대한 타 서브-트랜잭션 T_2^b가 실행해야 하는 연산을 제어하는 기법을 보여준다. '종속'은 T_1^a와 T_2^b가 트랜잭션 종속성이 있음을 의미하고 '독립'은 T_1^a와 T_2^b가 트랜잭션 종속성이 없음을 의미한다. SR_{before}는 변경 시작 전 두 트랜잭션의 관계를 나타내고 SR_{after1}는 T_1^a의 변경이 완료된 상태를 나타낸다. 그리고 SR_{after2}는 T_1^a와 T_2^b 의 변경이 완료된 상태를 나타내며 $SR_{rollback1}$는 T_1^a가 철회될 때 T_2^b가 변경 중인 상태를 나타내고 $SR_{rollback2}$는 T_1^a가 철회될 때 T_2^b의 변경이 완료된 상태를 나타낸다. 이 때, [표 1]은 mid-rollback을 실행하기 전에 처리해야 하는 타 트랜잭션의 연산을 보여주고 있으며, [표 2]는 mid-rollback 실행 후에 처리해야 하는 타 트랜잭션의 연산을 보여준다.

예를 들어 T_2^b가 변경 이전에 T_1^a와 트랜잭션 종속성이 있고 (SR_{before}=종속) 변경을 완료한 후에도 트랜잭션 종속성이 존재하는 상태(SR_{after2}=종속)에서 T_1^a이 mid-rollback을 결정할 경우에 [표 1]에 따라 T_2^b는 서버로부터 'mid-rollback'메시지를 받는다. 그리고 T_1^a와 협동 작업을 하지 않은 트랜잭션은 즉시 전파에 의해 발생할 수 있는 서버의 부하를 줄이기 위해 탐지 기반 기법으로 철회 결과를 전파한다.

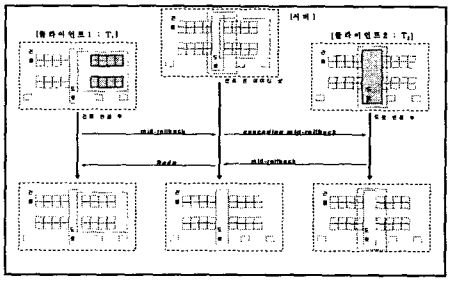
4. 회복 프로토콜 예

이 장에서는 이 논문에서 설계한 회복 프로토콜을 이용한 회복 제어 예를 보인다. [그림 1]은 각 클라이언트들이 T_1, T_2 그리고 T_3의 트랜잭션을 실행하여 서버의 데이터를 동시 변경할 때, 오류 발생에 의한 T_1의 철회 시 회복 절차를 보여준다. T_1에서는 서브-트랜잭션 T_1^a가 수정 작업을 하고 있고 T_2에서는 T_2^b가 수정 작업을 하고 있으며 T_3에서는 T_3^c가 수정 작업을 하고 있다. 그리고 각 서브-트랜잭션은 2단계 완료 프로토콜[3]에 의해 변경을 완료한다. 이 때 클라이언트1은 사용자의 오류 때문에 T_1^a의 mid-rollback을 요청하게 되고 서버는 [표 1, 2]에 따라 타 클라이언트의 트랜잭션을 제어한다. 따라서 서버는 T_1^a와 트랜잭션 종속성이 존재하는 T_2^b의 cascading mid-rollback을 요청하는 메시지를 클라이언트2에 보내고, T_1과 협동 작업 중이었던 클라이언트3에는 T_3^c의 redo 메시지를 보낸다. 따라서 T_3는 T_3^c를 재 시작하고 T_2^b는 *cascading mid-rollback*을 결정하고 서버에 mid-rollback을 요청한다. 그리고 서버는 T_1과 T_3에 redo 메시지를 보내어 T_1^a와 T_3^c를 재 시작하면 mid-rollback에 의한 작업은 종료되게 된다.



[그림 1] 회복 프로토콜

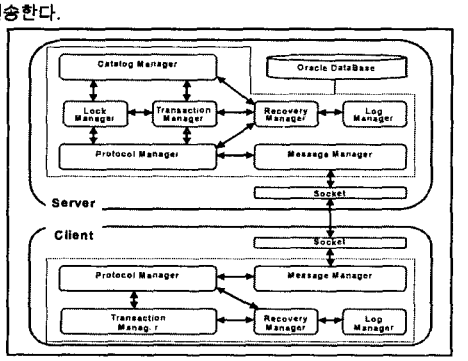
[그림 2]는 두 트랜잭션 T_1과 T_2가 도로와 건물을 동시에 변경한 후 회복하는 예를 보여 준다. 사용자 오류로 인해 T_1의 서버-트랜잭션 T_1^a가 mid-rollback하게 되면 [그림 1]의 프로토콜에 따라 T_1^a와 트랜잭션 종속성이 존재하는 T_2^b를 실행한 클라이언트2에 cascading mid-rollback을 요청하여 T_2^b를 mid-rollback한다. 즉, T_1^a와 T_2^b는 변경 이전 상태가 되어 무결성 원칙이 지켜진다.



[그림 2] 공간 데이터 변경 트랜잭션의 회복 예제

5. 구현

[그림 3]은 설계 및 구현한 시스템 구조를 보이고 있다. 서버는 DEC 400/500에 설치된 오라클 데이터베이스를 대상으로 구현하였고 클라이언트는 Windows 98 환경의 Microsoft Visual C++ 6.0 도구를 이용하여 구현하였다. 그리고 서버와 클라이언트는 소켓 통신으로 데이터를 전송한다.

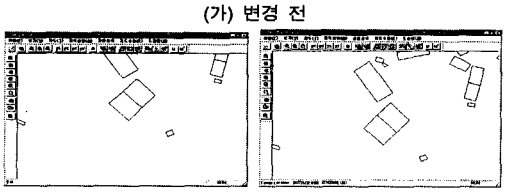


[그림 3] 시스템의 구조

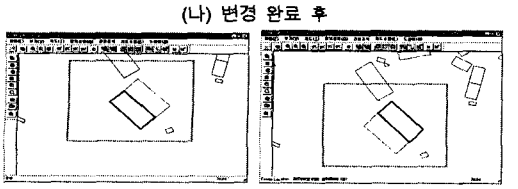
서버는 크게 Oracle Database에 접근해서 메타 정보를 관리하는 Catalog Manager, 잠금 정보를 설정하거나 가져오는 Lock Manager, 트랜잭션 정보를 관리하는 Transaction Manager, 회복에 관련된 제어를 하는 Recovery Manager, 그리고 클라이언트 요청 메시지에 대해 전체적인 프로토콜을 관리하는 Protocol Manager로 구성된다. 클라이언트는 Protocol Manager와 Transaction Manager, 그리고 Recovery Manager로 구성된다.

[그림 4]는 구현된 시스템을 적용하여 지도의 동시 변경에서 발생하는 트랜잭션 회복의 예를 보여 준다. [그림 4]의 데이터 셋은 도시 지역의 '주택외 건물' 레이어로써 현재 서로 다른 두 클라이언트가 공간 관련성이 존재하는 두 객체를 수정하고 있다. (가)는 초기 상태를 나타낸다. (나)는 두 클라이언트가 동시 수정을 완료한 상태를 보이고 있다. 여기서, 굵은 사각형은 사용자에 의해 설정된 영역 잠금[4]이다. (다)는 클라이언트1이 mid-rollback을 요청한 후 철회를 하고 클라이언트2는 서버로부터 mid-rollback 메시지를 받은 상태를 보인다. (라)는 클라이언트 트랜잭션 T_1 의 서브-트랜잭션 T_1^a가 mid-rollback되고 이에 따라 T_2의 서브-트랜잭션 T_2^b가 cascading mid-rollback되어 공간 관련성이 유지되었음을 보여 준다.

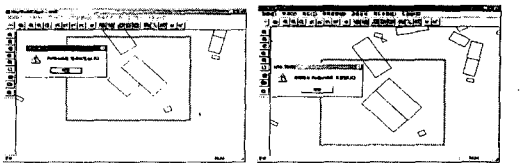
(가) 변경 전



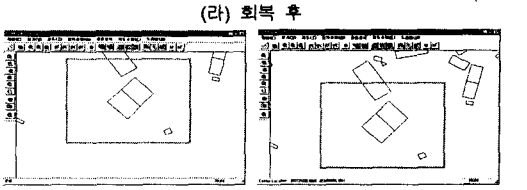
(나) 변경 완료 후



(다) 클라이언트1의 회복 완료 후



(라) 회복 후



<클라이언트1:T_1>　　<클라이언트 2 : T_2>

[그림 4] 구현된 시스템에 의한 트랜잭션의 회복 예제

6. 결론 및 향후 연구

이 논문에서는 클라이언트-서버 GIS 환경에서 다수의 클라이언트에 의해 동시 수정되는 공간 데이터 변경 트랜잭션을 위한 회복 기법을 설계 및 구현하였다. 즉 mid-rollback을 이용한 부분 철회 기법과 회복 제어 알고리즘 및 프로토콜을 설계하고 구현하여 결과를 보였다.

향후 연구 과제는 이 논문에서 제시한 회복 기법을 미들 웨어 기반의 분산 공간 데이터베이스에서 공간 데이타 동시 수정을 위한 트랜잭션의 회복 기법에 적용하는 것이다.

참고 문헌

[1]. J. Eliot, B. Moss, "Log-Based Recovery for Nested Transactions", Proceedings of the 13th VLDB Conference, Brighton 1987, pp.427-423, 1987.

[2]. C. Mohan, "ARIES/CSA: A Method for Database Recovery in Client-Server Architectures", ACM SIGMOD, vol23, no2, pp.55-66, 1994.

[3]. 최진오, 홍봉희, "분산된 지리정보시스템에서 새로운 잠금 기법을 이용한 중복된 공간 데이터의 변경 전파",한국정보과학회 논문지, vol26, no9, pp.1061-1072, 1999.

[4]. 신영상, 최진오, 조대수, 홍봉희, "클라이언트 변경 트랜잭션에서 동시성 및 일관성 제어", '99 한국정보과학회 가을 학술발표논문집, vol26, no2, pp.323-325, 1999.

[5]. 박재관, 김동현, 홍봉희, "클라이언트-서버 환경에서 공간 데이터 변경 트랜잭션의 회복 기법", '2000 한국정보과학회 봄 학술발표논문집, vol27, no1, pp.125-127, 2000.