트랜잭션 우선 순위 상승을 이용한 분산 실시간 병행수행제어 기법

(A Distributed Real-Time Concurrency Control Scheme using Transaction the Rise of Priority)

이 종 설 † 신 재 룡 †† 조 기 형 ††† 유 재 수 †††

(Jong Sul Lee) (Jae Ryong Shin) (Ki Hyung Cho) (Jae Soo Yoo)

요 약 실시간 데이타베이스 시스템이 분산환경으로 확장됨에 따라 기존의 실시간 병행수행 제어 기법을 분산환경으로 적용할 필요성이 대두되었다. 이에 본 논문에서는 중복 저장(replication)을 지원하는 분산 실시간 데이타베이스 시스템을 위한 효율적인 병행수행 제어 기법을 제안한다. 제안하는 기법은 중복 저장을 지원하는 분산 실시간 환경에서 완료준비 단계에 도달한 트랜잭션의 우선 순위를 상승시킴으로써 트랜잭션의 완료를 최대한 보장하며 완료 준비 단계에서 재시작 되는 트랜잭션을 줄이고 잠금 지연 시간을 최소화한다. 또한, 우선 순위가 상승된 트랜잭션이 점유한 데이타에 대한 대여(lending)를 허용함으로써 데이타를 차용(borrowing)한 트랜잭션의 대기시간이 감소되고 전체적인 시스템 성능이 향상되었다. 제안한 기법에 대한 성능 평가에서는 Firm 실시간 데이타베이스 환경에서 2 단계 완료 기법을 기반으로 DO2PL_ PA와 MIRROR 기법을 대상으로 트랜잭션의 도착율, 크기, 쓰기 가능성 및 데이타 중복율에 대한 마감시한 초과비율을 비교한다.

Abstract As real-time database systems are extended to the distributed computing environment, the need to apply the existing real-time concurrency control schemes to the distributed computing environment has been made. In this paper we propose an efficient concurrency control scheme for distributed real-time database system. Our proposed scheme guarantees a transaction to commit at its maximum, reduces the restart of a transaction that is on the prepared commit phase, and minimizes the time of the lock holding. This is because it raises the priority of the transaction that is on the prepared commit phase in the distributed real-time computing environment. In addition, it reduces the waiting time of a transaction that owns borrowed data and improves the performance of the system, as a result of lending the data that the transaction with the raised priority holds.

We compare the proposed scheme with DO2PL_PA(Distributed Optimistic Two-Phase Locking) and MIRROR(Managing Isolation in Replicated Real-time Object Repositories) protocol in terms of the arrival rate of transactions, the size of transactions, the write probability of transactions, and the replication degree of data in a firm-deadline real-time database system based on two-phase commit protocol. It is shown through the performance evaluation that our scheme outperforms the existing schemes.

· 본 연구는 한국과학재단 특정기초과제(과제번호:1999-1-303-007-3) 연구비 지원에 의해 수행되었음.

† 비 회 원 : 삼성전자 무선사업부 연구원
leejs98@samsung.co.kr
†† 비 회 원 : 충북대학교 정보통신공학과
jrshin@netdb.chungbuk.ac.kr
††† 종신회원 : 충북대학교 전기전자공학부 교수
khjoe@cbucc.chungbuk.ac.kr
yjs@cbucc.chungbuk.ac.kr

논문접수 : 2000년 8월 14일
심사완료 : 2001년 7월 4일

1. 서 론

컴퓨터 네트워크의 발달에 따라 통신 시스템과 군사 시스템 그리고 전자상거래와 같은 실시간 데이타베이스 시스템이 분산, 중복환경으로 확장되어졌다[1,2,3]. 이와 같은 실시간 데이타베이스 시스템의 분산, 중복환경으로의 확장은 데이타 지역성의 향상과 병렬 질의 실행, 그리고 부하 분산 등을 통해 전체 시스템의 성능을 향상시킬 수 있다는 장점을 갖는다. 그러나 분산 환경으로 확장됨에 따라 병행수행제어와 회복 기법은 더욱 복잡해졌

다[4,5]. 따라서 실시간 데이타베이스의 직렬성 및 시간 제약조건의 보장과 시스템 성능 향상을 위해 기존의 실시간 병행수행제어 기법 및 완료 기법을 분산환경에 적용하기 위한 다양한 연구들이 진행되어왔다[3,6,7,8].

기존의 실시간 병행수행제어 기법을 분산환경에 적용하기 위한 연구로는 크게 비관적 접근 방법과 낙관적 접근방법을 분산환경으로 확장하는 것이다. 비관적 접근 방법은 잠금을 이용하는 것으로 분산 병행수행제어 방법인 D2PL(Distributed Two Phase Locking)과 DO2PL (Distributed Optimistic Two Phase Locking)을 실시간 병행수행제어 기법으로 확장하는 방법[9]과 MIRROR (Managing Isolation in Replicated Real-time Object Repositories)[8] 등이 있다. 낙관적 접근 방법으로는 OPT-Sacrifice, OPT-Wait, Wait 50등[10]을 분산환경으로 확장하는 방법이 있다[6]. 또한, 분산환경의 완료 기법을 개선한 OPT(OPTimistic commit protocol) 방법[11]이 있다. MIRROR 프로토콜은 다른 비관적 접근 방법에 비해 향상된 성능을 갖는다. 그러나 완료 단계에서의 처리 시간이 길어질 경우 우선 순위 역전현상 발생 및 트랜잭션의 대기 시간 증가가 불가피하다. 또한, 낙관적 접근 방법은 재시작에 의한 자원의 낭비가 심하다.

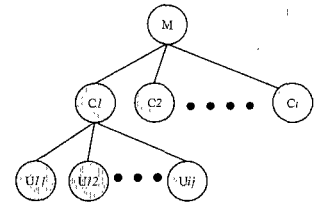
본 논문에서는 이와 같은 기존 기법들의 문제점을 해결한 분산 실시간 데이타베이스 시스템을 위한 병행수행제어 기법을 제안한다. 제안하는 병행수행제어 기법은 시간 제약 조건을 만족시키지 못하는 트랜잭션을 무시하는 Firm 실시간 데이타베이스와 2 단계 완료 기법을 기반으로 한다. 또한 제안하는 기법은 분산 실시간 환경에서 완료준비 단계에 도달한 트랜잭션의 우선 순위를 상승시킴으로써 트랜잭션의 완료를 최대한 보장하며 완료 준비 단계에서 재시작 되는 트랜잭션을 줄이고 잠금 지연 시간을 최소화한다. 뿐만 아니라, 우선 순위가 상승된 트랜잭션이 점유한 데이타에 대한 대여를 허용함으로써 데이타를 차용한 트랜잭션의 대기시간을 감소시키고 전체적인 시스템 성능을 향상시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 분산 실시간 병행수행제어 기법과 분산 실시간 완료 기법에 대한 기존 연구를 살펴보고, 3장에서는 새롭게 제안하는 우선 순위 상승 기법에 대해 기술한다. 4장에서는 기존 기법들과의 성능평가를 통해 제안한 병행수행 제어 기법의 우수성을 입증한다. 마지막 5장에서는 결론을 맺고 향후 연구 방향을 제시한다.

2. 관련 연구

2.1 중복 저장을 지원하는 분산 트랜잭션의 구조

일반적으로 분산환경에서는 트랜잭션이 제시된 주 사이트에서 실행되는 주(master) 트랜잭션(M)과 주(master) 트랜잭션에 의해 다른 여러 사이트에서 실행되는 종(cohort) 트랜잭션(C_i)들로 구성된다. 일반적으로 데이타는 중복될 수 있으므로 데이타를 변경하는 모든 종 트랜잭션은 다른 사이트에 대해 중복된 데이타를 변경하는 하나 이상의 다른 사이트에 대한 원격 갱신(remote update) 트랜잭션(U_{ij})들을 갖는다. 특히, 종 트랜잭션이 변경하고자 하는 데이타의 복사본을 저장하고 있는 원격 사이트(remote site)는 종 트랜잭션과의 원만한 병행수행 제어를 통해 해당 데이타에 대한 변경을 실행하여야 한다. 이를 위해 종 트랜잭션은 완료 프로토콜의 첫 번째 단계인 완료 준비 단계(voting phase)에서 변경할 데이타에 대한 사본을 원격 갱신 트랜잭션에 전달한다. (그림 1)은 중복 저장을 허용하는 분산 환경을 나타낸다.



M(Master Transaction) : 트랜잭션이 제시된 주 사이트에서 실행되는 프로세스
C (Cohort Transaction) : Master 트랜잭션의 하위 트랜잭션으로 여러 사이트에서 나누어 실행되는 프로세스

그림 1 중복 저장을 지원하는 분산환경

이와 같은 분산환경의 병행수행제어를 위해 사용되는 여러 기법들을 살펴보면 다음과 같다.

2.2 D2PL

D2PL에서 트랜잭션이 읽기 잠금을 요구할 때는 데이타의 복사본 중에 하나의 사본에 대해서만 읽기 잠금을 설정한다[9]. 그러나 중복된 데이타에 대해 쓰기 잠금을 요구할 때는 데이타의 모든 복사본들에 대해서 쓰기 잠금을 설정한다. D2PL은 중복된 데이타의 변경 시 모든 복사본들에 대해서 잠금을 획득할 때까지 트랜잭션의 실행이 중지되며, 실시간 환경으로 확장할 경우 재 시작되는 트랜잭션으로 인한 자원의 낭비가 심각해지는 단점을 갖는다.

2.3 DO2PL

DO2PL에서도 데이타에 대해 읽기 잠금을 요구할 때 D2PL과 마찬가지로 여러 사이트에 존재하는 데이타 사본 중에서 하나의 사본에 대해 읽기 잠금을 설정한다

[9]. 그러나 중복된 데이타에 대해서 쓰기 잠금을 요구할 때에는 각 종 트랜잭션은 지역 사본에 대해서만 쓰기 잠금을 설정한다. 그리고 다른 사이트의 원격 사본에 대한 쓰기 잠금은 prepare 메시지가 전송될 때 획득되어진다.

DO2PL에서는 트랜잭션의 실행이 끝난 뒤 완료 준비 단계에서 주 트랜잭션의 prepare 메시지가 도착하면 각 종 트랜잭션은 변경될 데이타 항목과 prepare 메시지를 해당 원격 갱신 트랜잭션에 전달한다. 각 원격 갱신 트랜잭션에서는 변경할 데이타에 대해서 쓰기 잠금을 설정한 뒤 prepared 상태가 되며 자신의 종 트랜잭션에게 prepared 메시지를 전달한다. 모든 원격 갱신 트랜잭션에서 보내온 prepared 메시지를 받은 종 트랜잭션은 prepared 상태가 되고, 자신의 주 트랜잭션에게 prepared 메시지를 전달한다. 결정 단계에서 주 트랜잭션은 종 트랜잭션에게 완료(commit) 메시지를 전달하고, 완료 로그를 추가한다. DO2PL은 트랜잭션의 실행이 끝난 뒤에 원격 사본에 대해서 쓰기 잠금을 허용함으로써 D2PL에 비해서 자원의 낭비를 줄일 수 있다.

D2PL과 DO2PL을 실시간 환경으로 확장하기 위해서는 실시간 충돌 해결 방법을 적용해야 한다. 실시간 충돌 해결 방법으로 PA(Priority Abort)를 적용할 경우 우선 순위가 낮은 트랜잭션은 완료 단계에서 철회될 가능성이 있으며 PB(Priority Blocking)를 적용할 경우 완료준비 단계의 장기 트랜잭션에 의해 대기 시간이 길어질 수 있다. PI(Priority Inheritance)를 적용할 경우에는 너무 복잡해지는 단점이 있다.

2.4 MIRROR

DO2PL을 기반으로 한 분산 실시간 병행수행 제어 기법인 MIRROR의 가장 큰 특징은 트랜잭션의 상태에 따라 데이타의 충돌에 대한 해결 방법을 다르게 적용하는 것이다[8]. 종 트랜잭션과 원격 갱신 트랜잭션의 상태는 경계점(demarcation points)으로 구분된다. 여기에 서 경계점이란 트랜잭션의 상태를 구분하는 기준점이다. 예를 들어 종 트랜잭션에서는 prepare 메시지를 받는 시점이며 원격 갱신 트랜잭션은 prepare 메시지와 함께 전달된 변경할 모든 데이타에 대한 쓰기 잠금을 획득하는 시점을 일컫는다.

MIRROR에서는 충돌 해결을 위해 트랜잭션 상태에 따라 경계점 이전에는 PA 방법을 적용하여 낮은 우선순위를 갖는 트랜잭션이 철회될 수 있으며, 경계점 이후에는 완료 준비 상태에 있으므로 충돌되는 다른 트랜잭션들의 우선 순위와 상관없이 계속 수행이 보장 되도록 PB 방법을 적용한다. MIRROR의 특징은 상태 정보를 얻기 위해 사이트간의 추가적인 통신이나 동기화가 불필요하고, 완료 기법의 경우 2단계 완료 기법을 변경할 필요가 없다는 것이다. 따라서 보통의 데이타 변경이 발생하는 중복 저장을 허용하는 분산환경에서 우수한 성능을 보인다.

그러나 경계점 이후에서 PB를 적용함으로써 발생하는 문제점은 완료 단계의 기간이 길어질 경우 우선 순위 역전현상이 발생하고, 트랜잭션의 대기 시간이 길어지며 교착 상태가 발생할 수 있다. firm 실시간 데이타베이스 시스템에서 우선 순위는 마감시간과 밀접한 관련을 갖고 있으므로 우선 순위 역전 현상이 자주 발생되면 그만큼 전체적인 마감시간 초과비율이 증가될 수 밖에 없다. 따라서, 우선 순위 역전 현상 방지 및 교착 상태를 발생시키지 않은 보완책이 필요하다.

2.5 OPT(PROMPT: Permits Reading Of Modified Prepared-data for Timeliness)

[11]에서 제안된 OPT 기법은 트랜잭션 T_j가 필요로 하는 데이타가 완료 준비 상태(prepared state)인 트랜잭션 T_i에 의해 점유되었을 경우 트랜잭션 T_i가 사용중인 데이타에 대해서 트랜잭션 T_j에게 해당 데이타의 사본을 빌려준다[11]. 이때 트랜잭션 T_i, T_j를 각각 대여자(lender), 차용자(borrower)라 한다. 이 경우 대여자 트랜잭션과 차용자 트랜잭션간에는 다음과 같은 세 가지의 실행 결과가 발생할 수 있다. 첫째는 차용자가 데이타에 대한 실행을 완료하기 전에 대여자가 결정단계를 통해 완료 또는 철회가 결정되어질 경우이다. 이 경우에는 대여자가 완료되면 차용자는 실행을 계속하게 되며, 대여자가 철회되면 차용자 또한 철회된다. 둘째는 대여자가 결정단계를 통해 완료 또는 철회가 결정되어지기 전에 차용자의 실행이 끝날 경우이다. 이 경우에는 차용자는 대여자의 완료 또는 철회가 결정되거나 자신의 마감시간이 초과될 때까지 대기하게 된다. 대여자가 철회되거나 마감시간이 초과되면 차용자는 철회되며, 대여자가 완료되면 차용자 또한 즉시 완료 절차를 수행한다. 마지막으로 대여자의 완료 또는 철회가 결정되기 전에 차용자가 실행 중에 철회되는 경우이다. 이 경우에는 차용자와 대여자의 대여 및 차용 기록은 무효화되며 차용자는 철회되어진다.

따라서, OPT 기법은 완료준비 단계의 트랜잭션의 정상적인 완료를 보장하며, 대부분의 대여자 트랜잭션이 완료된다고 가정했을 때 완료 준비상태에 있는 트랜잭션이 점유한 데이타를 필요로 하는 다른 높은 우선 순위 트랜잭션에게 대여함으로써 대기 시간을 줄일 수 있는 장점을 갖는다.

3. 제안하는 병행수행제어 기법

3.1 개요

실시간 데이타베이스 시스템에서는 자원에 대한 충돌 해결과 트랜잭션의 시간제약 조건을 만족시키기 위해 각 트랜잭션에 우선 순위를 할당한다. 본 논문에서는 Firm 실시간 데이타베이스를 대상으로 하며 우선 순위 할당 방법으로 마감 시간에 가장 근접한 트랜잭션에 높은 우선 순위를 할당하는 EDF기법[4]을 사용한다. 또한 충돌 해결을 위해 PA와 2PL을 사용하며, 분산 완료 기법인 2 단계 완료 기법을 이용한다.

본 논문에서 제안하는 병행수행제어 기법은 DO2PL_PA 기법을 확장하여 완료 준비 단계에 있는 낮은 우선 순위 트랜잭션의 철회를 방지하여 재시작에 의한 자원 낭비를 줄이고, 대여/차용 방법을 적용하여 데이타의 활용을 높임으로써 전체적인 시스템 성능을 향상시킨다. 이를 위해 전체 우선 순위의 영역을 일반 우선 순위 영역과 상승 우선 순위 영역으로 이분한다. 일반 우선 순위 영역은 트랜잭션이 처음 시작될 때 할당받는 우선 순위 영역으로 항상 상승 우선 순위 보다 낮다. 반대로 상승 우선 순위 영역은 실행을 마친 트랜잭션이 완료 준비 단계로 진입할 때 재 할당받는 우선 순위 영역으로 항상 일반 우선 순위 영역 보다 높은 우선 순위를 가지므로 완료를 최대한 보장받게 된다. 여기에서 우선 순위 상승은 전체 우선 순위 등급의 절반만큼을 높여주는 것이다.

제안하는 기법의 전체적인 처리 과정은 다음과 같다. 처음 시작되는 트랜잭션은 일반 우선 순위를 할당받고 읽기 단계에 들어간다. 읽기 단계에서 데이타에 대한 충돌이 발생할 경우 데이타를 점유한 트랜잭션이 상승 우선 순위이면 해당 데이타를 차용하여 처리를 계속한다. 그러나, 데이타를 점유한 트랜잭션이 일반 우선 순위인 트랜잭션일 경우에는 PA를 적용하여 두 트랜잭션 간의 충돌을 해결한다. 트랜잭션의 실행이 끝나면 완료 준비 단계에 들어간다. 완료 준비 단계에서 트랜잭션은 완료를 최대한 보장받기 위해서 상승 우선 순위를 재 할당받는다. 상승 우선 순위를 재 할당받은 트랜잭션은 모든 하위 트랜잭션에 대해서 처리 완료 여부를 확인하게 된다. 전체 하위 트랜잭션의 완료 여부가 결정되면, 최종 결정 단계로 들어간다. 결정 단계에서는 전체 트랜잭션의 완료 여부를 판단하여 트랜잭션의 완료 및 철회를 최종적으로 결정하고 그 결과를 각 하위 트랜잭션에게 전달한다.

3.2 우선 순위 영역의 분할

본 논문에서 제안하는 기법의 특징인 우선 순위 영역의 분할 방법은 다음과 같다. 시스템이 할당 가능한 우선 순위의 영역을 (그림 2)와 같이 중간 값을 기준으로 상승 우선 순위 영역과 일반 우선 순위 영역으로 구분하여 적용한다. 중간 값은 시스템이 할당할 수 있는 최고 우선 순위와 최저 우선 순위의 중간 값이다. 처음 시작되는 트랜잭션의 경우 일반 우선 순위 영역에 할당되고, 트랜잭션이 prepare 메시지를 전달받은 뒤에 상승 우선 순위 영역에 새로운 우선 순위를 재 할당받게 된다. 새롭게 할당받는 우선 순위는 최초 할당받은 일반 우선 순위에서 중간 값을 뺀 값이 된다. 예를 들어 데이타베이스에서 할당할 수 있는 우선 순위의 영역이 최고 우선 순위가 1이고 최저 우선 순위가 10이라고 가정할 때 중간 값은 ⌊11/2⌋=5가 된다. 따라서, 처음 시작되는 트랜잭션에 할당되는 일반 우선 순위 영역은 6~10이며 상승 우선 순위 영역은 1~5가 된다. 처음 시작되는 트랜잭션이 일반 우선 순위 7을 할당받고 실행이 끝나게 되면 완료 준비 단계에서 상승 우선 순위 2(일반 우선순위−중간 값)를 재 할당받는다.

완료 준비 단계에서 상승 우선 순위를 재 할당받은 트랜잭션은 충돌이 발생하는 다른 일반 우선 순위를 할당받은 트랜잭션 보다 항상 우선 순위가 높으므로 철회되지 않으며 완료가 보장된다. 그러나, 동일한 우선 순위를 갖는 두 트랜잭션이 서로 다른 사이트에서 동일한 데이타에 대해 동시에 완료준비단계에 들어갈 경우에는 두 트랜잭션 중에서 하나의 트랜잭션을 선택하여야 한다.

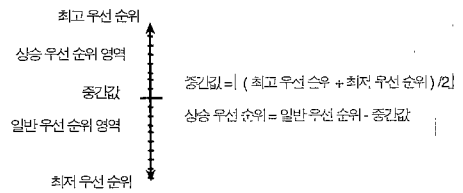


그림 2 우선 순위 영역 분할

3.3 데이타의 대여와 차용

실시간 완료 기법인 OPT[11]에서는 완료 준비 상태인 트랜잭션이 점유한 데이타에 대해서 다른 트랜잭션이 데이타를 필요로 하는 경우 해당 데이타에 대해 대여를 허용한다. 본 논문에서 제안하는 병행수행제어 기법에서는 OPT기법을 변형해서 적용한다. 제안하는 기법에서는 prepare 메시지를 전달받은 종 트랜잭션은 상승 우선 순위를 재 할당받는다. 이 시점에서 트랜잭션이 완료되기 이전까지 점유한 데이타에 대해서 일반 우선 순위를 할당받은 다른 트랜잭션이 필요로 할 경우에

는 데이타의 대여를 허용한다. (그림 3)에서 상위 우선 순위 트랜잭션 T_i는 점유한 데이타를 일반 우선 순위 트랜잭션 T_j가 필요로 할 경우 해당 데이타에 대한 대여를 허용한다. 이때 데이타에 대해서 대여를 필요로 하는 트랜잭션들간에는 PA를 적용하여 충돌을 해결하며, 대여자 및 차용자에 대한 기록은 TBT(Trade Brief Table)에 기록된다. 제안된 데이타의 대여와 차용은 OPT[11]와 마찬가지로 다음의 세 가지 경우를 갖는다. 첫째, 일반 우선 순위인 차용자가 데이타에 대한 실행을 완료하기 전에 상승 우선 순위인 대여자가 결정단계를 통해 완료 또는 철회가 결정되어질 경우이다. 이 경우에는 대여자가 완료되면 TBT에서 대여 및 차용의 기록을 제거한 뒤 차용자는 실행을 계속하게 되며, 만약, 대여자가 철회되면 데이타에 대한 일관성 유지를 위해 차용자 또한 철회된다. 둘째, 대여자가 결정단계를 통해 완료 또는 철회가 결정되어지기 전에 차용자의 실행이 끝나면 차용자는 대여자의 완료 또는 철회가 결정되거나 자신의 마감시간이 초과될 때까지 대기하게 된다. 만약 대여자가 철회되거나 마감시간이 초과되면 차용자는 철회되며, 대여자가 완료되면 차용자 또한 즉시 완료 절차를 수행한다. 셋째, 대여자의 완료 또는 철회가 결정되기 전에 차용자가 실행 도중에 철회되는 경우이다. 이 경우에는 차용자와 대여자의 대여 및 차용 기록은 무효화되며 차용자는 철회되어진다.

이와 같이 상승 우선 순위 트랜잭션과 충돌이 발생하는 일반 우선 순위 트랜잭션에게 해당 데이타에 대한 대여를 허용함으로써 데이타를 차용한 일반 우선 순위 트랜잭션은 블럭킹 되지 않고 수행될 수 있다. 이를 통해 완료 준비 단계인 트랜잭션이 점유한 데이타를 필요로 하는 일반 우선 순위 트랜잭션들의 대기 시간을 줄일 수 있다.

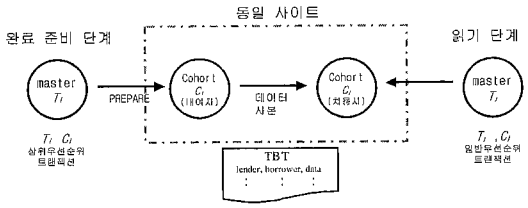


그림 3 데이타의 대여와 차용

3.4 실행 단계

(그림 4)에서와 같이 처음 시작되는 일반 우선 순위 트랜잭션 T_i는 각 종 트랜잭션들에 대해서 하위 트랜잭션을 실행한다. T_i가 잠금을 요구하는 데이타가 사용중이지 않을 경우에는 해당 데이타를 사용할 수 있지만 일반 우선 순위 트랜잭션 T_j에 의해 사용중인 경우, 두 트랜잭션에 대해서 PA를 적용하여 충돌을 해결한다. 만약 데이타를 사용중인 T_j가 상승 우선 순위를 갖는 경우에는 T_j가 점유한 데이타를 차용한 트랜잭션이 있는지를 확인한다. 차용중인 경우에는 해당 트랜잭션과 T_i는 PA를 적용하여 충돌을 해결해야 하고, 그렇지 않을 경우 T_i는 T_j가 점유한 데이타를 차용하여 블럭킹 없이 계속 수행되어진다. 이때 데이타를 차용한 트랜잭션을 차용자 트랜잭션이라고 하고 데이타를 대여한 트랜잭션을 대여자 트랜잭션이라 한다.

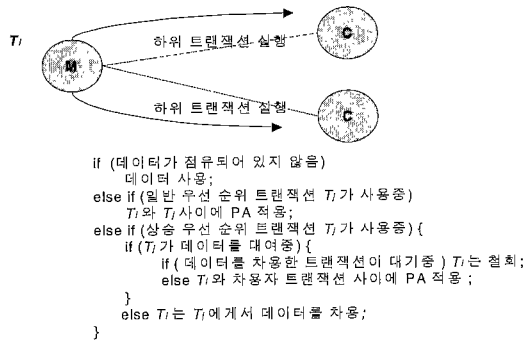


그림 4 실행 단계

3.5 완료 준비 단계

완료 준비 단계에서 일반 우선 순위를 갖는 종 트랜잭션은 (그림 5)와 같이 prepare 메시지를 전달받은 후, 상승 우선 순위를 재 할당받는다. 이 종 트랜잭션은 차용한 데이타의 대여자 트랜잭션의 완료가 결정되지 않았을 경우에는 대여자 트랜잭션의 완료 또는 철회 시까지 대기해야 한다. 대여자 트랜잭션이 완료된 경우, 대기중인 차용자 트랜잭션은 변경할 데이타 항목과 prepare 메시지를 원격 갱신 트랜잭션에게 전달한다. 만약, 대여자 트랜잭션이 철회되는 경우, 차용자 트랜잭션은 주 트랜잭션에 철회(abort) 메시지를 전달한다. 그러나, 차용한 데이타가 없을 경우에는 변경할 데이타 항목과 prepare 메시지를 원격 갱신 트랜잭션에게 전달한다. 이때 원격 갱신 트랜잭션은 변경할 데이타에 대해 쓰기 잠금을 설정한다. 만약 충돌이 발생할 경우 PA를 적용하여 충돌을 해결한다. 모든 데이타에 대해 잠금을 획득한 원격 갱신 트랜잭션은 자신의 종 트랜잭션에게 prepared 메시지를 전달한다. 모든 원격 갱신

트랜잭션에게 prepared 메시지를 전달받은 종 트랜잭션은 주 트랜잭션에게 prepared 메시지를 전달한다.

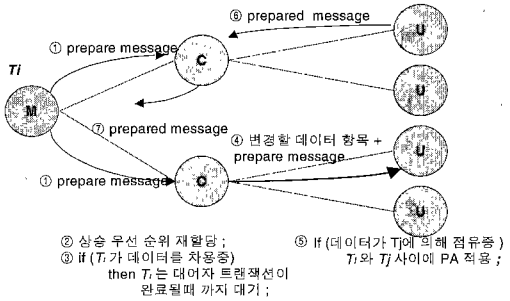


그림 5 완료 준비 단계

3.6 결정 단계

(그림 6)에서는 제안하는 기법의 마지막 단계인 결정 단계를 보여준다. 그림에서와 같이 완료준비 단계에서 검증을 성공적으로 마친 T_i는 commit 메시지를 종 트랜잭션에게 전달한다. 종 트랜잭션은 완료 로그 레코드를 추가한 뒤 원격 갱신 트랜잭션에게 완료 메시지를 전달한다. 그리고, 데이타를 차용한 트랜잭션들의 경우 대여/차용 기록을 삭제하고 대기중인 차용자 트랜잭션을 실행시킨다. 원격 갱신 트랜잭션은 완료 로그 레코드를 추가한 후에 종 트랜잭션에게 확인 메시지를 전달한다. 그리고 모든 원격 갱신 트랜잭션에게서 확인 메시지를 전달받은 종 트랜잭션은 주 트랜잭션에게 확인 메시지를 전달한다. 마지막으로 모든 종 트랜잭션의 확인 메시지를 받은 주 트랜잭션은 완료 종료를 로그에 기록한다.

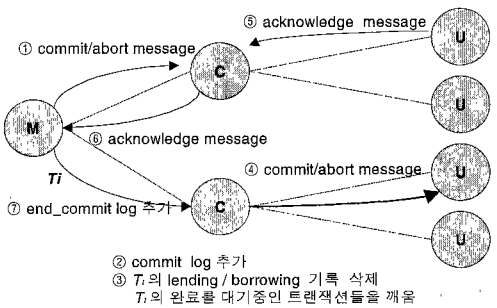


그림 6 결정 단계

3.7 구현의 복잡성과 교착상태 회피

본 절에서는 제안하는 분산 병행수행제어 기법의 구현 복잡성과 교착상태 회피에 대해 기술한다.

3.7.1 구현의 복잡성

비교 대상인 MIRROR[8]에서는 아래와 같은 구현 이득에 대한 특성을 갖는다.

- 경계점을 설정하기 위해 사이트간의 통신 또는 동기화가 불필요하다.
- 2단계 완료 프로토콜과 관련된 메시지, 로그 및 핸드셰이크 순서의 변경이 필요 없다.
- 프로토콜 구현측면에서 각 로컬 사이트들의 잠금 관리자 변경이 간단하다.

본 논문에서 제안하는 기법은 MIRROR와 마찬가지로 위의 세 가지의 구현 이득을 모두 갖는다. 게다가 MIRROR와 달리 PB를 위한 우선 순위 대기 큐가 필요하지 않으므로 MIRROR보다 구현이 용이하다.

3.7.2 교착상태 회피

MIRROR를 포함한 대부분의 병행수행 제어 기법은 2단계 잠금 기법을 기반으로 하기 때문에 교착상태를 피할 수 없다. 따라서 MIRROR 에서는 교착상태의 발생 빈도가 적다는 가정 하에 적용 부담이 적은 시간 초과(time out) 방법을 사용하였다. 제안하는 기법에서는 서로 다른 사이트에서 실행된 종 트랜잭션 *C1, C2*가 서로를 원격 갱신 트랜잭션 *U11, U21* 로 필요할 경우 교착상태가 발생 될 수 있다. 이와 같은 상황을 해결하기 위해 우선 순위를 이용한다. 만약, 동일 우선 순위를 갖는 경우에는 각 트랜잭션의 도착 시간(Arrival time)을 비교하여 도착 시간이 늦은 트랜잭션을 희생시킨다. 최악의 경우, 각 트랜잭션의 도착 시간도 동일한 경우에는 두 트랜잭션을 모두 재시작 시킨다. 그러나 이와 같은 상황은 발생 빈도가 매우 낮기 때문에 거의 발생되지 않는다.

4. 성능 평가

4.1 시뮬레이션 모델

본 논문에서 제안한 분산 실시간 병행수행 제어 기법에 대하여 DO2PL_PA와 MIRROR[8]를 대상으로 성능 평가를 실시하였다. 성능 평가에 사용된 시뮬레이션 모델은 MIRROR에서 적용한 모델을 기반으로 하였다. 실험에 사용된 컴퓨터는 Sun Enterprise 250 기종에 1GB의 주기억 공간을 가지고 있고 OS는 Solaris 2.7, 컴파일러는 gcc 2.8을 사용하였다.

9개의 사이트에 각각 1000개의 페이지를 두고, 페이지들에 대한 접근 성향은 정규분포가 되도록 하였다. 또한 초당 도착하는 트랜잭션의 개수인 트랜잭션의 도착 비율은 10~100개까지 선형적으로 증가시켜 보았다. 성

표 1 시스템 파라미터

파라미터	내용	설정값
NumSites	전체 사이트의 수	9
DBSize	데이타베이스 크기	1000 pages
NumCPUs	CPU 수	2
NumDataDisks	데이타 디스크의 수	4
PageDisk	Disk page 접근 시간	20ms
PageCPU	CPU page 접근 시간	10ms
NumLogDisks	로그 디스크의 수	1
BuffHitRatio	버퍼 적중률	0.1
InitWriteCPU	디스크 쓰기 초기화 시간	2ms
LogDisk	로그 반영 시간	5ms
MsgCPU	CPU 메시지의 송수신 시간	1ms
TransSize	트랜잭션당 page 수	16*(0.5 ~ 1.5)
Deadline	트랜잭션의 마감시한	도착시간+실행시간*Slack_Factor
ArrivalRate	트랜잭션의 도착율	40 Trans/Second
WriteProbability	트랜잭션의 쓰기 가능성	30%
ReplDegree	데이타의 중복율	10 ~ 100%

능 평가에 사용된 파라미터는 (표 1)과 같으며, 트랜잭션의 마감 시간은 (식 1)과 같다.

마감시간=도착시간+실행시간*Slack_Factor (1)

시뮬레이션 모델의 각 구성 요소는 (그림 7)과 같다. 트랜잭션 생성기에서는 시스템 파라미터를 바탕으로 시뮬레이션에 사용될 트랜잭션을 생성한다. 데이타베이스 관리기에서는 데이타베이스를 생성하고 각 데이타베이스를 관리한다. 트랜잭션 관리기는 트랜잭션 생성기에서 생성한 트랜잭션에 대한 주 트랜잭션을 생성하고 실행한다. 주 트랜잭션 관리기, 종 트랜잭션 관리기, 원격 갱신 트랜잭션 관리기에서는 각 트랜잭션에 의해 생성되는 하위 트랜잭션의 생성, 실행, 완료를 관리한다. 트랜잭션간에 충돌 해결은 충돌 관리기에서 제공하는 PA 및 PB를 사용한다.

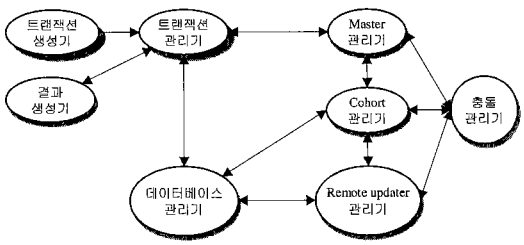


그림 7 시뮬레이션 모델의 각 구성요소

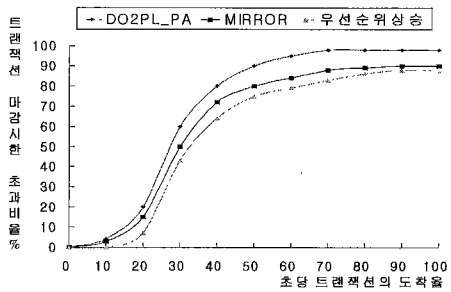
4.2 실험 결과

첫 번째 실험에서는 초당 트랜잭션의 도착율이 증가함에 따른 마감시한 초과비율의 변화를 알아보았다. 이 실험에서 데이타 중복율은 40%, 트랜잭션 쓰기 가능성은 30%이며 기타 파라미터는 (표 1)과 같다. 대부분의 시뮬레이션에서 각 트랜잭션은 실행시간의 2~6배만큼의 여유시간을 갖는다. 예를 들어, 비교 대상인 MIRROR의 경우 Slack_Factor를 6으로 고정하여 실험한 결과를 제시하고 있다. 그러나 본 논문에서는 Slack_Factor를 2배, 4배, 6배의 값으로 모두 실험하고 가장 우수한 성능을 보이는 Slack_Factor를 찾아보았다. 그리고 이 결과를 바탕으로 나머지 실험에서는 가장 우수한 성능을 나타내는 Slack_Factor를 사용하였다.

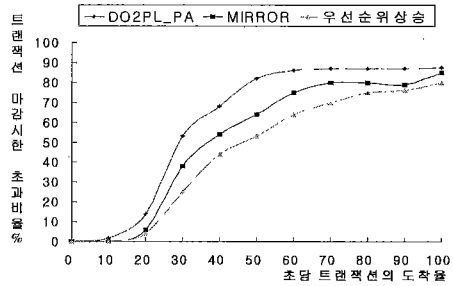
(그림 8)의 (a), (b), (c)는 Slack_Factor를 각각 2배, 4배, 6배로 설정하여 시뮬레이션을 수행한 결과 그래프이다. 이 세 가지 결과 모두에서 제안하는 기법이 가장 우수한 성능을 보임을 알 수 있었다. 또한 그림 (d)와 같이 Slack_Factor와 성능 향상의 관계 그래프를 얻을 수 있었다. 그림 (d)와 같이 Slack_Factor가 2인 경우에는 DO2PL_PA에 비해 제안하는 기법이 21.6%의 성능향상을 보인다. 또한 MIRROR에 비해서는 11.6%의 성능 향상을 보인다. 그리고 Slack_Factor가 4인 경우에는 각각 30.6%와 16.3%의 성능 향상을 보인다. 마지막으로 Slack_Factor가 6인 경우, 각각 35.3%와 17.2%의 성능 향상을 보임으로써 가장 적합한 Slack_Factor임을 알 수 있었다.

다른 실험들에서는 트랜잭션의 쓰기 비율이 30%로 고정된 환경에서 실험을 실시했다. 반면에, 두 번째 실험에서는 트랜잭션의 쓰기 가능성의 변화에 따른 트랜잭션 마감시한 초과비율을 알아보았다. 실험에서 적용한 데이타 중복율은 40%이고 초당 트랜잭션 도착율은 40 Trans/Second 이다. (그림 9)와 같이 트랜잭션의 쓰기 가능성이 증가함에 따라 트랜잭션 마감시한 초과비율이 증가되는데 제안하는 기법이 DO2PL_PA, MIRROR에 비해 성능이 향상되었음을 알 수 있다.

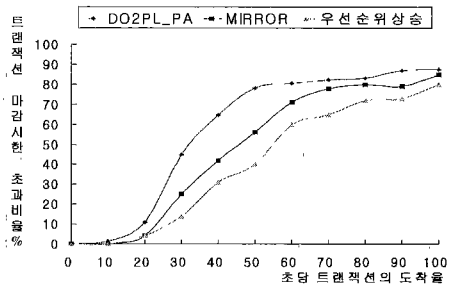
트랜잭션의 도착율과 쓰기 가능성뿐만 아니라 트랜잭션의 크기에 따라서 마감시한 초과율이 변할 수 있다. 따라서, 세 번째 실험에서는 (그림 10)과 같이 트랜잭션의 크기에 따른 마감시한 초과율의 변화를 알아보았다. 실험에서 데이타 중복율은 40%이고 초당 트랜잭션의 도착율은 40 Trans/Second 이며 트랜잭션의 쓰기 가능성은 30% 이다. 실험 결과에서 다른 실험들과 마찬가지로 마감 시한 초과 비율이 상대적으로 감소되는 것을 알 수 있다.



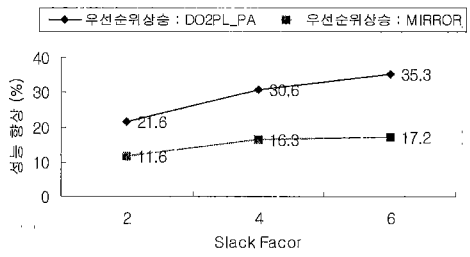
(a) Slack_Factor=2인 경우



(b) Slack_Factor=4인 경우



(c) Slack_Factor=6인 경우



(d) Slack_Factor와 성능 향상의 관계 그래프

그림 8 트랜잭션의 도착율에 따른 마감시한 초과비율

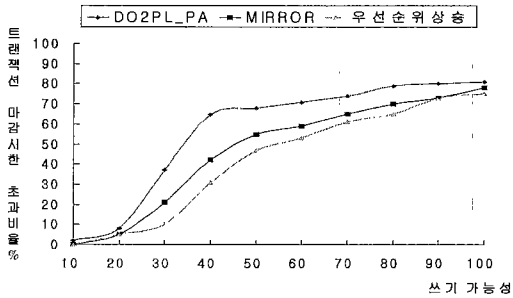


그림 9 쓰기 가능성의 변화에 따른 마감시한 초과비율

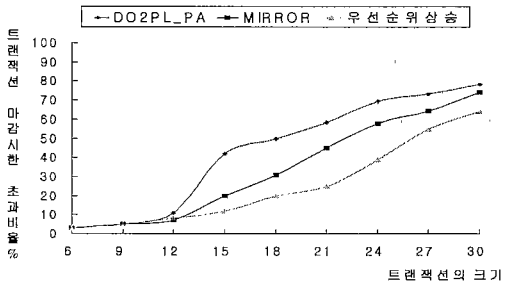


그림 10 트랜잭션의 크기에 따른 마감시한 초과비율

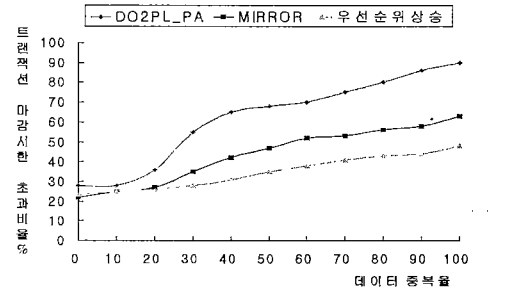


그림 11 데이타의 중복율에 따른 마감시한 초과비율

분산 데이타베이스 시스템에서 중복율 또한 데이타베이스의 성능과 깊은 관계를 가진다. 따라서, 마지막 실험에서는 (그림 11)과 같이 데이타베이스의 중복비율의 변화에 따른 각 기법의 마감시한 초과비율을 알아보았다. 중복율의 변화는 데이타베이스에서 중복 페이지의 수를 나타낸다. 실험에서 트랜잭션의 도착율은 40 Trans/Second이고 트랜잭션의 크기는 16pages/transaction이다. 트랜잭션의 Write probability는 30%이며 기타 사용된 파라미터는 (표 1)과 같다. 실험 결과, 다른 기법에 비해 중복율에 크게 영향을 받지 않음을 알 수 있다.

트랜잭션의 도착율, 트랜잭션의 쓰기 가능성 및 트랜잭션의 크기가 증가함에 따라 각 기법의 마감시한 초과 비율이 증가하는 것은 각 파라미터가 증가함에 따라 트랜잭션의 충돌빈도가 높아지고 충돌을 통한 재시작 트랜잭션의 누적에 따라 시스템 작업량이 증가하기 때문이다.

실험 결과, 제안한 기법은 트랜잭션의 도착 비율이 30~70 일 때 가장 우수한 성능을 보인다. 그리고 트랜잭션의 크기가 15~27 일 때와 트랜잭션의 쓰기 가능성이 40~80 일 때 가장 높은 성능을 보임을 알 수 있다. 또한, 트랜잭션의 도착 비율, 쓰기 가능성, 크기에 대한 실험 결과와 마찬가지로 DO2PL_PA에서는 중복율이 증가함에 따라 마감시한의 초과비율이 급격하게 증가하는 것을 알 수 있다. 이것은 완료단계인 트랜잭션에 대한 완료 보장을 하지 못하므로 철회비율이 높아지고, 따라서 자원 낭비를 초과하여 전체적인 성능이 저하된 것이다. 반면에 제안된 기법에서는 트랜잭션의 완료를 보장함으로써 성능을 향상시킬 수 있었다. 이와 같이 중복 데이타를 허용하는 분산환경에서는 다른 기법에 비해서 우수한 성능을 가지는 것을 확인 할 수 있다.

5. 결 론

본 연구에서는 중복 데이타를 허용하는 분산 실시간 데이타베이스 시스템을 위한 효율적인 병행수행 제어 기법을 제안하였다. 새롭게 제안된 기법은 트랜잭션의 완료 단계에서 우선 순위를 상승시킴으로써 트랜잭션의 재시작에 의한 낭비를 줄일 수 있고 트랜잭션의 완료를 최대한 보장하며, 잠금 지연 시간을 최소화하는 장점을 갖는다. 또한 완료 준비 단계의 트랜잭션이 소유한 데이타를 필요로 하는 경우 해당 데이타에 대한 차용을 허용함으로써 차용자 트랜잭션의 지연시간을 줄일 수 있었고 이에 따라서 전체적인 시스템 성능이 향상되었다.

제안하는 병행수행 제어 기법은 충돌 해결을 위해 PA만을 사용한다. 따라서 PA와 PB를 사용하는 MIRROR와 달리 우선 순위 대기 큐가 필요하지 않으므로 MIRROR보다 구현이 용이하다. 그리고 MIRROR는 낮은 우선 순위를 갖는 트랜잭션이 완료준비단계에 있을 때 우선 순위 역전 문제를 발생시킬 수 있다. 그러나 제안하는 기법은 트랜잭션 우선 순위 상승 방법을 통해 이 문제를 해결하고 있으므로 우선 순위 역전 문제가 발생되지 않는다.

제안하는 기법은 DO2PL_PA 및 MIRROR와의 성능 평가를 통해 초당 트랜잭션의 도착율, 트랜잭션의 쓰기 가능성, 트랜잭션의 크기, 데이타의 중복율 변화에 따른 마감시한 초과율을 비교하였다. 실험 결과에서 초당 트랜잭션의 도착율이 30~70 Trans/Second, 쓰기 가능성이 40~80%, 트랜잭션의 크기가 15~27, 데이타 중복율이 30% 이상일 때 트랜잭션의 마감시한 초과율이 가장 낮아짐을 알 수 있었다.

향후, 낙관적 병행수행 제어 기법인 OPT-Sacrifice, OPT-Wait, Wait-50으로 확장하고 성능평가를 수행할 계획이다.

참 고 문 헌

[1] C. Mohan, B. Lindsay and R. Obermarck, "Transaction Management in the R* Distributed Database Management System," ACM TODS, 11(4), 1986.

[2] Franaszek, P.A., Haritsa, J.R., Robinson, J.T. and Thomasian, A., "Distributed concurrency control based on limited wait-depth," Parallel and Distributed Systems, IEEE Transactions on Volume: 4, Page(s): 1246-1264, 1993.

[3] Kwok-Wa Lam, Lee, V.C.S., Kam-Yiu Lam and Sheung-Lun Hung "Parallel and Distributed Real-Time Systems," Proceedings of the 4th International Workshop on, Page(s): 122-125, 1996.

[4] Michael J. Carey. and Miron Livny. "Distributed Concurrency Control Performance: A Study of Algorithm, Distribution, and Replication," Proceedings of the 14th VLDB Conference, Los Angels, California Page(s): 13-25, 1988.

[5] Thomasian, A., "Distributed optimistic concurrency control methods for high-performance transaction processing," Knowledge and Data Engineering, IEEE Transactions on Volume: 10, 1, Jan.-Feb., 1998.

[6] Baothman, F., Sarje, A.K. and Joshi, R.C., "On optimistic concurrency control for RTDBS," TENCON '98. 1998 IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control Volume: 2, Page(s): 615-618 vol.2, 1998.

[7] Son, S., "Advances In Real-Time Systems," Prentice Hall, 1995.

[8] Xiong, M., Ramamritham, K., Haritsa, J. and Stankovic, J., "MIRROR: A State-Conscious Concurrency Control Protocol in Replicated Real-time Database," Technical Report 98-36, Department of Computer Science, University of Massachusetts at Amherst, 1998.

[9] Carey, M., and Linvy, M., "Conflict Detection

Tradeoffs for Replicated Data," ACM Transactions on Database Systems, Vol. 16, Page(s): 703-746, 1991.

[10] Haritsa, J. R., Carey, M.J. and Livny, M., "Dynamic real-time optimistic concurrency control," Real-Time Systems Symposium, Proceedings., 11th, Page(s): 94-103, 1990.

[11] Gupta, R., Haritsa, J., Ramamritham, K. and Seshadri, S., "Commit processing in distributed real-time database systems," Real-Time Systems Symposium, 17th IEEE, Page(s): 220-229, 1996.

이 종 설

1996년 충북대학교 정보통신공학과(공학사). 2001년 충북대학교 정보통신공학과(공학석사). 2001년 ~ 현재 삼성전자 무선사업부 연구원. 관심분야는 실시간 데이타베이스, 분산 데이타베이스, 병행수행 제어, XML 등

신 재 룡

1996년 충북대학교 정보통신공학과(공학사). 1998년 충북대학교 정보통신공학과(공학석사). 2001년 충북대학교 정보통신공학과(박사과정 수료). 관심분야는 데이타베이스 시스템, 실시간 시스템, 멀티미디어 데이타베이스 등

조 기 형

1966년 인하대학교 전기공학과(공학사). 1984년 청주대학교 산업공학과(공학석사). 1992년 경희대학교 전자공학과(공학박사). 1988년 ~ 현재 충북대학교 공과대학 전기전자공학부 교수. 관심분야는 데이타베이스, 화상처리 및 통신, GIS, 통신프로토콜

유 재 수

1989년 전북대학교 공과대학 컴퓨터공학과(학사). 1991년 한국과학기술원 전산학과(공학석사). 1995년 한국과학기술원 전산학과(공학박사). 1995년 ~ 1996년 목포대학교 전산통계학과 전임강사. 1996년 ~ 현재 충북대학교 공과대학 전기전자공학부 부교수. 관심분야는 데이타베이스 시스템, 정보검색, 멀티미디어 데이타베이스, 분산 객체 컴퓨팅