Journal of the Korea Academia-Industrial cooperation Society
Vol. 13, No. 10 pp. 4786-4793, 2012

http://dx.doi.org/10.5762/KAIS.2012.13.10.4786

모바일 컴퓨팅 환경에서 BISLD를 이용한 회복모델 개선

조성제[1*], 한성수[2]
[1]동방대학원대학교 동방문화학과, [2]동방대학원대학교 문화정보학과

Recovery Model Improvement using BISLD in Mobile Computing Environment

Sung-Je Cho[1*] and Seong-Soo Han[2]
[1]**Division of Dongbang Culture , Dongbang Graduate University**
[2]**Division of Culture Information, Dongbang Graduate University**

요 약 최근 모바일 컴퓨팅 환경에서 실시간 트랜잭션 처리에 대한 트랜잭션 모델에 대한 연구가 활발히 진행되고 있다. 그러나 회복모델에 대한 연구는 상대적으로 미흡하다. 이러한 모바일 컴퓨팅 환경에서는 핸드오프가 발생하였을 경우 데이터 손실이 발생하므로 데이터베이스의 안정성을 위한 모바일 회복기법은 매우 중요하다. 그래서 모바일 컴퓨팅 환경에서는 입출력을 줄일 수 있는 효율적인 회복기법에 대한 연구가 절실히 요구된다. SLL기법은 순차파일로 구성되어 있기 때문에 입출력시간이 증가된다. 본 연구에서는 모바일 컴퓨팅 환경에서 효율적인 모바일 회복 기법을 제안하였다. 마지막으로 제한한 회복기법의 성능평가를 수행하여 효율성을 입증하였다.

Abstract In a recent mobile computing environment, the research for a transaction model have been actively carried out to deal with real-time transaction. However, the research for recovery model is relatively insufficient. Therefore, it would be very important for mobile recovery methods for database stability, just in case of data loss due to hand-off in mobile computing system. So, further efficient research should be keenly done to reduce I/O hand-off in mobile computing systems. SLL method increases I/O time due to sequential file. This paper have been suggested to efficient mobile recovery method in mobile computing system. Finally, it has been clearly demonstrated for system efficiency, conducting the performance assessment that currently proposed recovery system.

Key Words : Mobile, Model, Transaction, Algorithm, Recovery, Database

1. 서론

최근 하드웨어와 초고속 인터넷 기술이 향상됨에 따라 이동장비에 의한 전자상거래, 뱅킹, 항공권 발매 등과 같은 분야에 사용률이 급격히 증가하고 있다. 이동통신 응용분야에서 효율적인 트랜잭션 처리를 위한 실시간 처리가 절실히 요구되고 있다. 이를 위해 모바일 컴퓨팅 시스템에 대한 연구가 진행되고 있다.[1-3,10]

일반적인 모바일 컴퓨팅 시스템에서는 통신장애로 인한 예기치 않은 고장이 발생된다. 그러므로 배터리 백업 기법을 사용하여 데이터베이스의 로그파일과 데이터를 저장하고, 이 파일을 회복기법에 적용한다.[2-4].

일반적으로 데이터베이스시스템의 회복시스템은 회복을 위하여 주기적으로 트랜잭션 처리에 대한 리두(REDO)정보와 언두(UNDO)정보를 로그파일에 기록하고 주기적으로 체크포인트를 수행하여 메인메모리상에 있는 데이터베이스의 상태를 디스크상의 백업 데이터베이스시스템에 반영하게 된다. 그러나 모바일 컴퓨팅 시스템에

*Corresponding Author : Sung-Je Cho
Tel: +82-10-3668-9861 email: chosj715@yahoo.co.kr
접수일 12년 09월 10일 수정일 (1차 12년 09월 26일, 2차 12년 10월 04일, 3차 12년 10월 09일) 게재확정일 12년 10월 11일

대한 회복기법에 대한 연구가 미흡한 상태이다. 본 연구에서는 모바일 컴퓨팅시스템에 B+트리를 적용한 회복기법을 제안하고자 한다. Segment Log List 기법(Segment Log List 기법 :이하 SLL기법이라고 한다)은 로그파일을 디렉터리형태로 구축하여 회복동작 때에 로그들을 분석한다. SLL기법의 문제점은 로그분석을 위해 마치 순차파일과 유사함으로써 회복시간이 많이 소요된다[2].

EHPLD기법은 SLL기법의 문제점을 개선하기 위해 확장성 해싱기법을 적용하였다[5]. 확장성 해싱을 위한 접근시간의 성능은 디렉터리가 메모리에 저장되어 있는 경우에는 한 번의 탐색으로 가능하다. 디렉터리가 디스크에 존재하는 경우에는 최악의 경우 성능은 두 배의 탐색이 요구된다.

본 연구에서는 SLL기법과 EHPLD기법을 개선하기 위해 B+트리 인덱스기법을 이용한 로그들을 클라이언트별로 구성한다. 그렇게 함으로써 장애가 발생할 때 신속히 로그디렉터리에 접근하여 회복을 수행함으로써 회복시간의 오버헤드를 제거하였다. 이 제안된 기법을 B+트리 인덱스 세그먼트 로그디렉터리(B+-tree Index Segment Log Directory : 이하 BISLD이라고 한다)기법이라 하며, 이 기법을 이용하여 회복기법을 수행하게 된다. 그 이유는 Cache Sensitive T-tree(Cache Sensitive T-tree:이하 CST이라고 한다)의 논문에서 B+-트리가 다른 인덱스보다 더 빠르다는 것을 기술하였다[3].

본 논문의 구성은 다음과 같다. 제2장에서는 관련연구로 모바일 컴퓨팅 시스템에서의 회복을 위한 기법들을 기술한다. 제3장에서는 모바일 컴퓨팅 시스템을 위한 B+트리를 이용한 회복시스템 모델을 제안하고, 제4장에서는 BISLD를 적용한 회복시스템의 알고리즘을 제안하였다. 제5장은 기존 기법과 제안된 기법을 성능분석 하였고, 6장에서는 결론을 기술하였다.

2. 관련 연구

2.1 Segment Log List 기법

이 기법은 클라이언트가 해당 트랜잭션을 언두 동작하여 서버의 병목현상을 감소하고, 더티(dirty)된 세그먼트를 이용하여 REDO 하여 트랜잭션을 재수행하는 기법이다. 각 클라이언트는 동시에 하나의 트랜잭션을 수행하며, 클라이언트에서 서버로부터 해당 세그먼트를 전송받아 트랜잭션을 수행하는 환경에서 트랜잭션 완료 후 로그를 서버에 전송하면, 서버는 세그먼트 로그 리스트(Segment Log List)를 통해 충돌 여부를 점검하여 비충돌이면 완료하고, 충돌이면 클라이언트에게 재 수행을 지시한다. 클라이언트가 트랜잭션을 재 수행 하고자 할 때 낙관적 동시성 제어 프로토콜에 의해 해당 데이터 세그먼트를 서버로부터 재전송 받아 수행해야 한다[2]. 이 기법의 단점은 세그먼트 로그 리스트의 디렉터리 크기가 클수록 로그파일을 찾는 시간이 길어지므로 그에 따른 오버헤드가 많이 발생한다.

2.2 EHPLD기법

이 기법은 SLL기법의 문제점을 개선하기 위하여 확장성 해싱기법을 적용한 회복기법을 제한하였다. 이 기법의 특징은 다음과 같다.

첫째, WAL(Write Ahead Log)기법을 통해서 입출력을 줄이고자 하였다.

둘째, 전체시스템 회복시간을 단축하기 위해 확장성 해싱 페이지 로그 디렉터리(Extendible Hashing Page Log Directory: 이하 EHPLD이라고 한다)기법을 제안하였다[5].

2.3 Pradhan 기법

이 기법은 스테이트 세이빙(State Saving) 전략과 핸드오프(Handoff) 전략으로 구성되어 있다. 이 전략의 특징은 로깅(Logging)과 노 로깅(No Logging) 기법을 이용한 회복기법을 제안하였다. 이 논문의 주요 목적은 새로운 모바일 컴퓨팅 환경의 제약조건을 기술하고, 그에 알맞은 회복 알고리즘을 설계하는 것이다. 로깅(Logging) 기법은 체크포인트가 주기적으로 일어나며, 쓰기 트랜잭션은 체크포인트에 의해서 로그에 기록된다. 장애로부터 모바일 클라이언트를 회복하고자 할 때 로그 엔트리에 대해서 체크포인트를 이용한다[6]. 이 기법의 단점은 트랜잭션 수행 완료 후 일관성을 위해 로깅에 대한 오버헤드가 발생된다.

2.4 Neves and Fuchs 기법

이 기법은 분산 시스템을 위한 체크포인트 기반의 회복기법을 제안하였다.

이 프로토콜은 주기적으로 안정된 저장장비의 트랜잭션 상태를 저장한다. 과거에 제안된 프로토콜은 통신단절 때문에 모바일 환경에 적합하지 않다. 고장이 발생할 경우 가장 최근에 저장된 트랜잭션의 상태로 롤백하고 커밋(Commit)이 이루어지지 않은 경우는 UNDO 수행을 실시한다. 이 프로토콜은 응용 체크포인트를 생성하는 동안 메시지를 교환한다[7]. 이 기법의 단점은 로깅을 위해 수시로 체크포인트를 수행하므로 통신 오버헤드가 발생된다.

2.5 George and Chen 기법

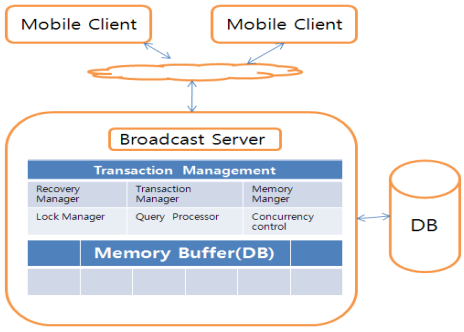
이 기법은 체크포인팅 및 로깅 기반으로 하여 모바일 어플리케이션의 효율적인 회복기법을 제안하였다. 현재의 접근 방식은 사용자의 이동과 관계없이 정기적으로 체크포인트를 시행한다. 사용자의 움직임을 기반으로 체크포인팅 제도는 핸드오프의 경계가 초과된 후에만 체크포인트가 걸린다. 최적의 경계는 실패율 및 어플리케이션과 모바일 클라이언트의 이동속도에 의해 적용된다. 각각의 어플리케이션과 모바일 클라이언트에 기반을 둔 체크포인팅 속도의 조정이 가능하다. 최적의 이동성 한계를 확인한다는 것은 모바일 노드의 이동속도, 실패율과 로그 도착율의 함수로 실패 당 복구비용을 최소화 하는 것이다. 만족한 복구시간에 복구가 행해질 수 있는 가능성을 계산하고 그 접근 방식의 적용에 대해 설명한다[8]. 이 기법의 단점은 로깅을 위해 수시로 체크포인트를 수행하므로 통신 오버헤드가 발생된다.

2.6 Man Hong Wong 기법

이 기법은 모바일 컴퓨팅 환경에서 읽기 전용 트랜잭션을 처리하기 위한 모바일 캐싱기법을 제시하였다. 이 기법은 캐시 내의 일관성을 유지하기 위한 방법으로 무효화 브로드캐스팅 전파를 하는 기법이다. 무효화 브로드캐스팅 전파는 기지국에서 갱신 데이터 아이템을 모바일 클라이언트에 전파하는 기법이다. 이 기법은 모바일클라이언트가 핸드오버 현상으로 인하여 캐시 내 데이터 일관성을 유지하기 어렵다[9]. 이 기법의 문제점은 다음과 같다. 첫째는 캐시 내의 데이터 일관성을 유지하기 어렵다. 둘째는 캐시 내의 데이터 일관성을 유지하기 위하여 버전 기법을 이용하고 있다. 이때, 방송서버와 모바일 클라이언트 사이에 일관성을 보장하기 위해 통신 오버헤드가 많이 발생된다. 두 번째는 읽기 전용 트랜잭션이므로 쓰기 트랜잭션의 경우에는 전혀 사용할 수 없는 단점이 있다.

3. B+ 트리 모바일 컴퓨팅 시스템

본 장에서는 B+-트리를 이용하여 SLL기법을 개선한 새로운 모바일 컴퓨팅시스템을 제안하였다. 이 시스템 구조는 그림 1과 같다.



[그림 1] B^+트리 모바일 컴퓨팅 시스템
[Fig. 1] B^+Tree in mobile computing system

3.1 모바일 컴퓨팅 시스템

그림 1은 방송서버 모듈과 모바일 클라이언트 모듈을 나타낸다. 방송서버 모듈은 트랜잭션관리와 데이터베이스로 구성된다. 회복관리자는 로깅과 체크포인트를 담당한다. 로깅은 트랜잭션을 수행한 후 발생한 더티 데이터(Dirty Data)의 변화에 대한 로그를 안전한 디스크에 저장하여 시스템 장애가 발생할 경우 회복에 사용한다. 체크포인트는 일정 주기마다 DB를 보조기억장치에 백업을 담당한다. 트랜잭션관리자는 트랜잭션에 필요한 데이터와 트랜잭션 수행을 담당한다. 메모리 관리자는 공유메모리에 데이터를 관리 위해 메모리 할당 및 회수를 담당한다. 록 관리자는 트랜잭션을 수행을 할 때 동시성 제어 기능을 구현에 사용된다. 질의어 처리기는 트랜잭션을 수행할 때 질의를 분석하여 수행을 담당한다.

3.2 방송서버

모바일 컴퓨팅시스템의 방송서버 역할은 트랜잭션관리(Transaction Management)와 메모리 버퍼관리(Memory Buffer Management) 담당한다. 트랜잭션관리의 구성은 회복관리자, 트랜잭션관리자, 메모리관리자, 록 관리자, 질의처리기, 그리고 동시성제어로 되어있다.

메모리 버퍼의 구성은 DB 버퍼, 로그버퍼, 인덱스 버퍼, 공유버퍼로 되어있다.

방송서버와 모바일 클라이언트 구성 중에 본 논문에서 필요한 관리자 및 버퍼만 기술한다.

3.2.1 회복관리

BISLD는 크게 B^+-트리기법과 디렉터리 로그 버퍼를 이용하여 회복한다. 모바일 클라이언트들 중에서 고장 난 클라이언트만 회복을 수행함으로서 기존 기법보다 회복시간이 훨씬 단축된다.

3.2.2 로그버퍼

본 연구에서는 회복을 위한 여러 로깅기법 중에서 지연기법을 적용한다. 그 이유는 모바일 클라이언트에서 트랜잭션 완료 후 갱신된 연산을 가지고 있으므로 재 수행만을 실시하는 장점을 가지고 있다. 로그버퍼의 역할은 각 모바일 클라이언트에서 트랜잭션을 완료 후 전송 한 로그 레코드를 저장한다.

3.2.3 메모리 버퍼

방송서버에 데이터베이스 복사본을 저장하기 위한 공간을 메모리 버퍼라고 한다. 메모리버퍼는 데이터 캐시버퍼와 로그버퍼로 구성된다. 데이터 캐시버퍼는 데이터베이스 복사본이 저장되고, 로그버퍼는 BISLD가 저장된다.

3.2.4 동시성제어

BISLD의 동시성제어 기법은 낙관적 동시성제어기법을 적용한다. 그 이유는 각 클라이언트가 방송서버를 통해 트랜잭션에 필요한 세그먼트 전송받아 무조건 수행한다. 같은 세그먼트 중에 같은 데이터를 사용하는 경우가 희박하기 때문이다. 록 단위를 세그먼트인 경우 트랜잭션 수행시간이 많이 소요된다. 동시성제어 기법을 낙관적 제어기법을 적용함으로써 시스템 병렬성을 향상할 수 있다.

방송서버는 방송 주기별로 모바일 클라이언트로 방송전파한다. 해당 모바일 클라이언트는 필요한 세그먼트를 캐시버퍼에 저장하고 트랜잭션을 수행한다. 트랜잭션을 수행하면서 발생하는 리두로그(Redo log)들은 로그버퍼에 저장한다.

3.2.5 질의처리 관리

모바일 컴퓨팅 시스템에서 방송서버는 세그먼트를 모바일 클라이언트에 방송전파만 담당하고 질의처리는 각 모바일 클라이언트의 DBMS에서 독립적으로 수행한다.

3.2.6 백업관리

모바일 컴퓨팅시스템 환경이므로 주기적으로 체크포인트에 의해서 백업 DBMS의 갱신을 담당한다. 시스템 장애시 빠른 회복을 위해 주기적으로 데이터베이스를 안정된 저장장치에 기록하여야 한다. 이때 백업관리는 BISLD크기를 고려하여 일정한 주기로 백업처리를 담당한다. 백업방법은 로그버퍼 내에 있는 로그 레코드를 이용하여 백업하며, 백업이 끝난 후 로그버퍼 내의 로그 레코드를 모두 제거한다.

3.2.7 BISLD

B+트리를 이용한 디렉터리로 구성되어 있다. 기존기법[2]는 순차파일 형태의 디렉터리로 구성되어 있기 때문에 회복을 위하여 로그 분석시간이 많이 소요된다. 본 논문에서 제안하는 기법은 B+ 인덱스를 이용한 회복을 수행함으로써 신속한 로그분석을 수행한다. 따라서 기존기법보다 빠른 회복을 할 수 있다.

3.3 모바일 클라이언트

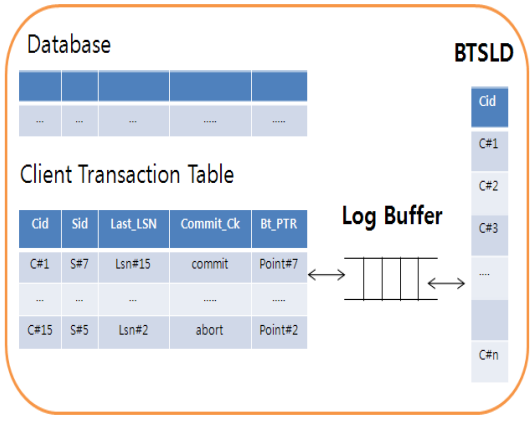
모바일 클라이언트는 로그버퍼와 세그먼트 버퍼로 구성되어 있다. 로그버퍼는 모바일 클라이언트에서 트랜잭션 수행 중에 발생하는 리두 로그 파일만 저장한다. 세그먼트 버퍼는 각 모바일 클라이언트에서 트랜잭션 수행에 필요한 세그먼트 파일을 방송전파를 통해 전송받아 저장하는 공간이다. 모바일 클라이언트에서 트랜잭션 수행 완료 후 로그버퍼를 모바일 컴퓨팅시스템으로 보낸 후 방송서버는 BISLD에 저장한다.

3.4 백업 DBMS

DBMS의 갱신을 위하여 주기적으로 체크포인트를 수행하여 메인메모리상에 있는 데이터베이스의 상태를 디스크상의 백업 데이터베이스시스템에 반영한다.

4. BISLD를 이용한 회복모델

BISLD를 이용한 회복모델은 그림 2와 같다. 그 구조의 구성은 클라이언트 트랜잭션 테이블(Client Transaction Table :이하 CTT), 로그버퍼, BISLD로 구성되어 있다.



[그림 2] BISLD를 이용한 회복모델
[Fig. 2] Recovery Model using BISLD

로그를 검색하기 위해서는 B^+트리 인덱스를 사용하여 로그 세그먼트 디렉터리의 리프에 대한 포인터를 가리킨다. BISLD 버퍼구조는 모바일 컴퓨팅시스템의 로그 인덱스를 기록한다. 그리고 세그먼트 로그 디렉터리에는 각 세그먼트별로 리두 로그파일을 두어 모바일 클라이언트별로 회복동작 때 사용한다.

SLL기법의 단점인 순차파일 형태의 디렉터리를 B^+ 트리 인덱스를 이용한 디렉터리로 구성함으로써 로그정보를 신속히 검색할 수 있다.

이와 같이 할 경우 전체 데이터베이스를 회복할 때 걸리는 시간에 대한 오버헤드를 제거할 수 있다.

BISLD구조는 모바일 클라이언트의 번호의 값을 갖는 헤더와 REDO로그의 정보에 대한 포인터로 되어 있으며, REDO로그는 로그파일들이 저장되는 엔트리 블록으로 구성되어 있다. BISLD는 B^+트리 인덱스 디렉터리와 REDO 로그의 집합으로 구성되어 있으며, 인덱스로 사용하여 디렉터리에서 엔트리 블록을 찾아서 해당 엔트리에 접근한다. 이때 엔트리 블록에서 오버플로가 발생하면 B^+트리 기법에 의해 분할되거나 디렉터리가 증가하며 확장된다. B^+ 트리는 오버플로가 발생한 블록만 분할하므로 분할 횟수를 줄이고 블록의 적재율을 높이는 장점이 있다.

4.1 트랜잭션 정의

본 연구에서는 모바일 컴퓨팅시스템에서 모바일 클라이언트로 전송하는 단위는 세그먼트이고, 트랜잭션은 모바일 클라이언트에서만 처리하는 것으로 가정한다[2].

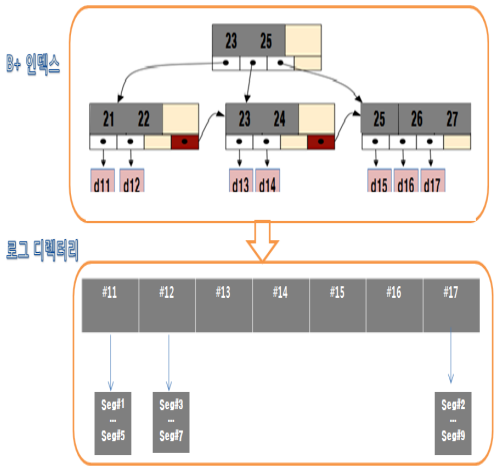
모바일 클라이언트에서 트랜잭션을 수행할 때 발생하는 모바일 클라이언트의 로그버퍼에 기록한다. 트랜잭션 수행할 때 마다 세그먼트의 갱신을 기록하기 위해 모바일 클라이언트번호, 세그먼트번호, 로그 일련번호, 갱신값을 저장한다.

4.2 BISLD 회복 시스템 구조

BISLD 회복구조는 그림 3과 같다. 삽입과 검색은 B+ 트리 알고리즘을 적용한다.

이 알고리즘을 적용함으로써 특정 클라이언트가 장애가 발생할 경우 리두 로그를 신속하게 검색하여 빠른 회복을 수행할 수 있도록 설계하였다.

그 이유는 B+ 트리의 특징은 특정 데이터의 검색을 할 경우 다른 인덱스들 보다 우수하기 때문이다[3].



[그림 3] BISLD 회복 구조
[Fig. 3] Recovery Structure in BISLD

4.3 회복 트랜잭션 가정

제안하는 회복 알고리즘을 위해 사용되는 가정들은 다음과 같다.[2]

첫째, 데이터베이스 회복시간 단축을 위하여 각 모바일 클라이언트에서 수행한 리두정보만 방송서버에 저장한다. 이렇게 함으로써 리두로그 정보와 더티 세그먼트를 전송하는 기존의 오버헤드의 문제점을 해결할 수 있다.

둘째, 모바일 클라이언트에서는 트랜잭션을 수행하고 방송서버에서는 각 모바일 클라이언트에서 수행 완료한 로그정보를 관리한다.

셋째, 장애는 ACK 메시지를 보낸 후 일정한 시간동안 ACK에 대한 메시지가 없으면 장애로 간주한다. 넷째, 본 논문에서는 모바일 컴퓨팅 환경에서의 발생 가능한 모바일 클라이언트와 방송서버 장애에 대한 회복기법을 제안한다.

4.4 자료구조

4.4.1 모바일 클라이언트의 로그 구조

모바일 클라이언트 내에는 로그를 저장할 수 있는 로그버퍼가 있고, 로그버퍼는 REDO 로그를 저장하며 그 구조는 그림 4와 같다.

[그림 4] 로그레코드 구조
[Fig. 4] Log Record Structure

로그구조의 설명은 다음과 같다.

① MC#은 모바일 클라이언트번호를 기록한다. 이것은 클라이언트별로 회복할 때 사용한다.

② SEG#은 세그먼트 번호를 기록한다. 모바일 클라이언트에게 전송하는 크기를 말한다,

③ LSN은 로그 일련번호를 기록한다. 트랜잭션 수행 시 발생하는 로그들을 순차적으로 일련번호를 부여하여 리두시 사용한다.

④ NEW VALUE은 세그먼트의 더티 값을 기록한다. 모바일 클라이언트가 방송서버에 전송하면 체크포인트에 의해 데이터베이스에 반영한다.

4.4.2 Client Transaction Table의 구조

CTT는 그림 5와 같은 형태로 방송서버에 유지되며, 방송서버는 텍스트와 멀티미디어 데이터를 유지할 수 있는 세그먼트를 저장한다. 그리고, CTT구조는 클라이언트 번호, 세그먼트번호, Last_LSN, Commit_Check 여부, Point 번호순으로 저장한다.

Cid	Sid	Last_LSN	Commit_Ck	PTR
C#1	S#7	Lsn#15	commit	Point#7
...	...	...	.....	.....
C#15	S#5	Lsn#2	abort	Point#2

[그림 5] CTT 구조
[Fig. 5] CTT Structure

4.5 BISLD를 이용한 회복모델

4.5.1 정상적인 상태에서 알고리즘

모바일 클라이언트는 트랜잭션을 처리하기 위해 데이터 아이템 X가 속한 세그먼트를 방송서버로부터 전송받으며, 처리된 트랜잭션의 결과를 방송서버로 보낸다. 구체적인 알고리즘을 살펴보면 다음과 같다.

① 모바일 클라이언트는 트랜잭션을 처리하기 위해 필요한 데이터 아이템 X가 속한 세그먼트를 방송서버로부터 전송받아 메모리 버퍼에 기록한다.

② 트랜잭션 처리기에 의해 메모리 버퍼에 있는 세그먼트를 이용하여 트랜잭션 수행 후 로그 레코드를 생성하여 로그 버퍼에 기록한다.

③ 트랜잭션이 Commit 직전에 로그버퍼에 있는 리두 로그레코드들을 방송서버로 전송한다.

④ 방송서버는 리두 로그레코드들을 전송받으면 로그 버퍼 해당 위치에 로그를 저장하고 이 로그정보를 이용하여 BISLD를 구축한다.

4.5.2 모바일 클라이언트의 로깅동작 알고리즘

모바일 클라이언트의 로깅동작 알고리즘은 다음과 같다.

① 모바일 클라이언트는 트랜잭션에 필요한 세그먼트들을 방송서버를 통해 전송 받는다.

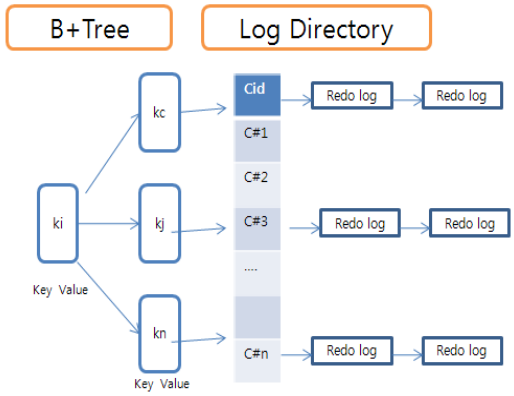
② 모바일 클라이언트는 트랜잭션을 수행하면서 리두 로그레코드와 언두 로그레코드를 생성해서 각각의 로그버퍼에 분리시켜 저장한다.

③ 만약 트랜잭션 철회 시에는 자신의 언두 로그버퍼를 이용하여 직접 철회동작을 수행한다.

④ 트랜잭션 수행 완료 시에는 방송서버로 재 수행 로그버퍼 내의 로그 레코드들을 보내고 언두 로그버퍼는 제거한다.

4.5.3 BISLD의 삽입 알고리즘

BISLD는 CTT정보에 의해서 충돌을 점검 후 로그버퍼에 모바일 클라이언트별로 리두 로그를 저장한다. 그리고 로그 레코드 정보를 이용하여 BISLD를 생성한다.



[그림 6] BISLD 회복 구조
[Fig. 6] Recovery Structure in BISLD

그림 6은 모바일 클라이언트 #1에서 생성한 로그레코드가 방송서버에 처음 도착하였기 때문에 Last_LSN에 일련번호를 부여하고 CTT내의 해당 위치에 로그 레코드를 기록한다. 이 로그정보에 있는 클라이언트번호, 세그먼트를 이용하여 BISLD에 로그정보를 기록한다. BISLD 정보는 세그먼트 재수행시 이 정보를 이용하여 로그버퍼에 해당 로그정보를 재 수행 할 수 있다.

BISLD의 삽입 알고리즘은 다음과 같다.

① 삽입할 리프노드를 검색한다.
② 인덱스 노드에 표시된 킷값보다 적은 노드를 검색한다.
③ 삽입할 키가 현 인덱스 노드의 모든 킷값보다 크다면 큰 노드를 찾아간다.
④ 다음 노드를 검색한다.
⑤ 삽입할 리프노드를 찾을 때까지 반복한다.

4.5.4 회복에 대한 로그정보 검색 알고리즘

방송서버에 장애가 발생 후 재가동되면 분석단계에서 로그버퍼를 분석하여 BISLD를 재구성한다. 방송서버에서는 모바일 클라이언트에게 방송전파를 통해 시스템 복구 메시지 전송한다. 그 후 각 모바일 클라이언트들은 리두 로그를 방송서버에 전송 후 commit 메시지를 회신 못한 경우 해당 트랜잭션 회복에 대한 요청한다.

요청을 받은 방송서버는 신청한 해당 클라이언트의 로그정보가 BISLD에 존재하면 트랜잭션 로그 레코드의 시작주소를 이용해서 트랜잭션 단위로 리두동작을 수행한다. 완료 후 모바일 클라이언트에게 요청된 commit 정보를 전파한다. BISLD를 이용한 회복에 대한 로그정보 검색 절차는 다음과 같다.

① 로그정보검색은 항상 B+Tree 인덱스의 루트부터 시작한다.
② 검색 모바일 클라이언트노드가 가장 작은 값보다 적으면 오른쪽 서버트리로 이동하여 검색
③ 검색 모바일 클라이언트노드가 가장 큰 값보다 크면 왼쪽 서버트리로 이동하여 검색
④ 검색할 때 까지 ②번과 ③번 단계를 반복한다.

5. 성능 평가

본 논문에서 기존기법(SLL기법, EHPLD기법)과 제안된 BISLD기법에 대한 세 가지의 구조를 구현하여 성능평가를 실시하였다. 성능평가를 위한 시험의 시스템 환경은 인텔 CPU 3.2GHz, 1GB RAM 환경에서 실시하였다. 각각의 구조를 C 언어로 구현하였다.

성능평가를 위한 연산형태는 해당로그를 검색 할 때 소요되는 시간을 실시하였다.

시뮬레이션 방법은 노드의 크기를 변경시키면서 회복을 위해 로그정보 검색 처리시간의 변화를 조사하여 세 기법을 분석하였다.

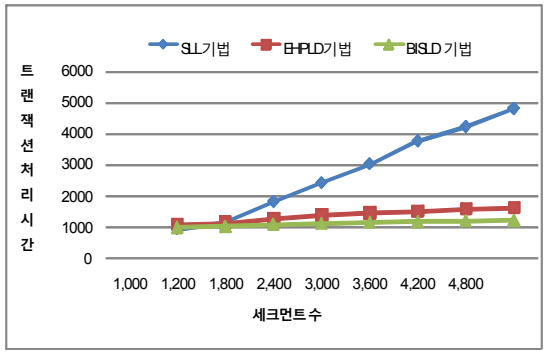
5.1 성능 비교 분석

본 논문에서 제시한 BISLD를 적용한 회복기법과 기존기법을 성능평가를 하였다. 그 결과는 표 1과 같다.

[표 1] 회복시간표 (단위 : ms)
[Table 1] Recovery Time Table

페이지수	SLL기법	EHPLD 기법	BISLD 기법
1,000	940	1,090	996
1,200	1,150	1,140	1,022
1,800	1,810	1,270	1,081
2,400	2,410	1,370	1,122
3,000	3,010	1,450	1,155
3,600	3,750	1,510	1,181
4,200	4,210	1,570	1,203
4,800	4,790	1,610	1,222

입력된 세그먼트의 수는 1000에서 4800을 발생시켰다. 회복처리시간(단위: 초)이 0에서 6000까지 증가됨에 따른 처리시간을 나타낸 것이다.



[그림 7] 회복 서비스시간
[Fig. 7] Recovery Service Time

그림 7은 기존기법과 제안된 기법의 성능결과를 그래프로 나타냈다. 그 결과를 살펴보면, BISLD기법이 SLL기법보다는 1000세그먼트수의 이하인 경우는 회복성능이 저조하지만 세그먼트의 수가 1200부터 회복성능이 우수 하였다. 세그먼트의 수가 적을 경우 BISLD기법이 인덱스 구성에 따른 오버헤드가 크다.

그러나 BISLD기법이 세그먼트의 수가 증가할 경우는 성능이 우수함을 그림 7에서 나타났다. 그리고 EHPLD기법은 SLL기법보다는 1000이하인 경우는 회복성능이 저조하지만 페이지의 수가 1200부터 회복성능이 우수 하였다. 그러나 EHPLD기법은 전반적으로 BISLD기법보다 회복성능이 저조하다. 그 이유는 CST의 연구결과를 보면 B+-트리의 인덱스구조가 다른 인덱스구조들 보다 우수함을 증명하였다.

BISLD기법이 성능이 뛰어난 이유는 SLL기법 의 회복 동작은 순차파일구조 이루어져 처리시간이 너무 많이 소요되기 때문이다. BISLD기법은 B+트리의 인덱스를 적용함으로써 회복에 소요되는 시간은 세그먼트 수가 증가함에 따라 상당한 시간이 단축되었다.

6. 결 론

본 논문에서는 모바일 컴퓨팅시스템 환경에서 발생하는 여러 가지 문제점들을 제시하고, 이 문제점을 해결을 위한 효율적인 회복기법을 제안하였다.

이 회복기법은 SLL기법과 EHPLD기법의 문제점들을 개선하기 위해 로그디렉터리를 B+인덱스를 적용한 로그 디렉터리를 구성하였다.

기존 기법들과 제안기법을 비교 분석한 결과 데이터가 적은 경우는 B+트리의 인덱스가 없기 때문에 제안기법이 기존기법보다 검색시간이 증가하였으나 1800세그먼트 이후부터 로그검색 시간이 상당히 감소하였다.

그 결과 회복동작에서 기존기법보다 우수함으로 나타났다. 그 이유는 순차파일에서 로그를 검색하는 것 보다 B+ 트리의 인덱스를 아용하는 것이 검색효율이 우수하기 때문이다[3]. 향후 모바일 컴퓨팅 환경에서 동시성제어에 대한 연구를 하고자 한다.

References

[1] DongHee-Kang and Sangkeun-Lee, “Memory Usage Scheme in Mobile Web Browser”, Journal of KIISE, Vol. 18 No. 06, pp.479 ~ 0483, 2012.

[2] SungJe-Cho, “An Efficient Recovery Method for Mobile Main Memory Database System”, Journal of the korea Society of IT Services, Vol.7, No.2, pp.131-143, 2008.

[3] Ig-hoon Lee and Sang-goo Lee, “Cache Sensitive T-tree Index Structure”,Journal of KIISE, Vol.32 No.1, pp.12-23, 2005.

[4] Kam-Yiu, Lam and Mei-Wai Au, “Broadcasting Consistent Data to Read-only Transaction from Mobile Clients”, The Computer Journal, vol. 45, No.2, pp.29 -46, 2002.

[5] K.K Lee and S.J Cho," Mobile Main Memory Database Recovery System Implement using EHPLD", The Journal of Korea Institute of Information Technology, Vol. 9 No.12, pp.73~92, 2012.

[6] Dhiraj K.Pradhan, Krishna,P .and Nitin H. Vaidya, “Recovery in Mobile Wireless Environment: Trade off Analysis”, 26th IEEE International Symposium on Fault Tolerance Computing, pp.16-25, 1996.

[7] Nuno Neves and Fuchs, W.K, “Adaptive recovery for mobile environments”, Communication of the ACM, Vol.40, Issue 1, pp.68-74, 1997.

[8] Sapna E .George,Ing-Ray Chen and Jing Jin, “Movement based Checkpointing and Logging for Recovery in Mobile Computing Systems”, Proceeding of the 5th ACM International Workshop on Data Engineering for Wireless and Mobile Access, 2006.

[9] Man Hong Wong, and Wing Man Leungm, “A Caching Policy to Support Read-only Transaction in a Mobile Computing Environment”, CS-TR-95-07, 1995.

[10] Jong-Ho Choi, "Design of Portable Signal Analysis System for Mobile WiMax Base Station", Journal of Korea Institute of Information Electronic Communication Technology Sciences, Vol. 4 No.1, pp.39~45, 2011.

조 성 제(Sung-Je Cho) **[정회원]**

- 1997년 2월 : 홍익대학교 전자계산학과(이학박사)
- 2007년 3월 ~ 현재 : 동방대학원대학교 동방문화학과 교수

<관심분야>
메인메모리 DBMS 및 모바일컴퓨팅, 문화콘텐츠 개발, 정보보안

한 성 수(Seong-Soo Han) **[정회원]**

- 2000년 2월 : 경상대학교 정보통신공학과(공학사)
- 2005년 8월 : 순천향대학교 산업정보대학원 정보통신과(공학석사)
- 2011년 9월 ~ 현재 : 동방대학원대학교 문화정보학과 박사과정

<관심분야>
컴퓨터 네트워킹 및 모바일컴퓨팅, 이동통신