다중 무선 방송채널에서의 효과적인 모바일 트랜잭션 처리 기법

(An Efficient Mobile Transaction Processing Scheme over Multiple Wireless Broadcast Channels)

정 호 련 † 정 성 원 †† 박 성 욱 †††

(Horyun Jeong) (Sungwon Jung) (Sungwook Park)

요 약 무선 방송 환경은 모바일 클라이언트 수에 상관없이 다수의 클라이언트에게 데이타를 보낼 수 있다는 특징이 있다. 이와 같은 특징으로 무선 방송은 많은 분야에 적용되고 있는데 이러한 응용분야에서는 대부분이 읽기 작업을 수행하며 데이타를 일관성 있게 관리하기 위하여 트랜잭션 단위의 동시성 제어 기법을 사용하고 있다. 기존 멀티 무선 방송 채널에서의 데이타 할당 방법으로 구성된 데이타 채널들에서는 단일채널에서 사용하는 동시성 제어 기법으로 트랜잭션의 일관성을 보장해 주기 어렵다. 이는 각 데이타 채널의 브로드캐스트 주기가 서로 다르기 때문에 특정 채널에서 데이타를 수신한 후, 다음 채널로 이동하였을 때 트랜잭션이 접근하는 데이타의 일관성이 깨질 수 있기 때문이다. 본 논문에서는 이러한 문제를 해결할 수 있는 멀티 무선 방송채널에서의 읽기 트랜잭션을 위한 동시성 제어 기법을 제안하였다. 이 논문에서는 기본적으로 인덱스 전용 채널과 데이타 전용채널 구조를 이용한다. 또한 LBCPC(Longest Broadcast Cycle Per Channel)라는 새로운 일관성의 단위를 제시한다. 데이타 전용채널에서는 이 LBCPC만큼 같은 BCPC(Broadcast Cycle Per Channel)내의 데이타를 반복하여 방송한다. 또한 LBCPC마다 전체 데이타에 대한 제어 정보를 이용하여 자체적으로 트랜잭션의 검증을 실시한다. 이로 인해 트랜잭션의 일관성이 유지될 뿐만 아니라 단일채널의 긴 브로드캐스트 주기보다 짧은 LBCPC로 인하여 재실행을 위한 대기 시간을 줄여줌으로써 평균 응답 시간을 줄여줄 수 있다. 또한 단일 채널에 비해 제어 정보를 자주 방송함으로써 읽기 전용 트랜잭션이 접근하는 데이타에 대한 최신성을 보장한다. 마지막으로 실험을 통해 단일 채널과의 트랜잭션의 평균 응답 시간이 현저히 감소함을 보임으로써 제안하는 다중 채널에서의 동시성 제어 기법의 성능을 검증한다.

키워드 : 모바일 데이타베이스, 모바일 트랜잭션, 무선 방송, 다중 무선 방송 채널, 동시성 제어기법

Abstract Wireless broadcast environments has character that a number of mobile client can receive data streaming from central server no matter how they are so many. Because it is asymmetric bandwidth in that uplink and downlink bandwidth are different. This advantage helps wireless broadcast environments is used in many applications. These applications work almost read operation and need control concurrency using transaction unit. Previous concurrency control scheme in single channel is not adapted in multi channel environments because consistency of data are broken when a mobile client tunes in a broadcast cycle in a channel and then move into another channel and listen to different broadcast cycle with already accessed broadcast cycle. In this paper, we propose concurrency control for read-only mobile transactions in multiple wireless broadcast channel. First of all, we adapt index and data dedicated channel and propose LBCPC(Longest Broadcast Cycle Per Channel) as new unit of consistency. In index dedicated channel, it is repeatedly broadcasted data in

† 학생회원 : LG전자 단말연구소 선임연구원
queenking@sogang.ac.kr
†† 종신회원 : 서강대학교 컴퓨터학과 교수
jungsung@sogang.ac.kr
††† 학생회원 : 서강대학교 컴퓨터학과
psw0405@sogang.ac.kr
논문접수 : 2007년 9월 27일
심사완료 : 2008년 2월 27일

정보과학회논문지: 데이타베이스 제35권 제3호(2008.6)

same BCPC(Broadcast Cycle Per Channel) until LBCPC. And mobile transaction executes validation using control information every LBCPC. As a result, consistency of data is kept and average response time is shorter than one in single channel because waiting time for restart reduces. And as control information is broadcasted more frequently than in single channel, it is guaranteed currency about data accessed by transaction. Finally, according to the simulation result, we verify performance of our scheme in multi channel as comparing average response time with single channel.

Key words : Mobile Databases, Mobile Transactions, Wireless Broadcast, Multiple Wireless Broadcast Channels, Concurrency Control

1. 서 론

최근 휴대폰이나 PDA와 같은 컴퓨터 하드웨어와 블루투스와 같은 무선 네트워크 기술의 급격한 발달은 모바일 컴퓨팅을 점차 가능케 하고 있다[1].

그러나 현존하는 기술들은 물리적 제약 조건으로 인해 좁은 통신 대역폭, 잦은 접속 단절, 배터리 부족 등의 문제를 가진다. 이러한 문제들을 해결하기 위해 데이타를 일정한 주기 동안 반복적으로 전송하는 무선 데이타 방송기법에 대한 많은 연구가 있었다[1-3]. 이러한 기법은 클라이언트가 명시적으로 데이타를 요구하지 않아도 원하는 데이타를 수신할 수 있으므로 상향 통신 대역폭을 낭비하지 않고, 서버는 클라이언트의 수에 대한 확장성을 가지는 장점이 있다.

그러한 장점으로 인해 무선 데이타 방송은 경매, 전자 입찰과 같은 전자 상거래 응용 분야와 주식 거래, 기상 정보, 교통 정보 방송과 같은 다양한 응용 분야에 적용되고 있다[4]. 이러한 응용 분야에서는 많은 사용자들이 동시에 하나의 아이템을 읽거나 갱신하는 상황이 발생하고, 따라서 데이타를 일관성 있게 관리하기 위해서는 트랜잭션 단위의 동시성 제어 기법이 필요하다.

그러나 무선 데이타 방송환경의 특성상 단일채널에서의 방송 사이클을 주기로 하여 일관성을 보장하는 전통적인 트랜잭션 처리 기법을 그대로 적용할 수 없다[5]. 따라서 다음과 같은 이유로 멀티채널(Multi-channel) 환경에서의 데이타 방송 기법이 연구 되어 왔다[6].

- 한 기지국이 전송능력을 가진 여러 대의 서버를 가지고 다른 기지국의 방송을 대신할 경우.
- 무선 환경에서 낮은 대역폭으로 인해 하나의 채널만으로는 효과적인 방송에 어려움이 있을 경우.
- 서버에서의 응용프로그램이 사용자의 증가를 수용하면서 안정된 서비스를 제공하기 위하여 별도의 채널을 필요로 할 때.

또한 단일채널에서의 트랜잭션 처리 방법과 다중 채널에서의 트랜잭션 처리 방법은 다음과 같은 이유로 구분되어야 한다. 기존의 단일채널 브로드캐스팅 환경에서는 서버측에서 일정한 브로드캐스트 주기 동안 일관성 있는 데이타들을 방송하였다. 그러나 다중채널에서 각 채널의 브로드캐스트 주기를 기반으로 트랜잭션을 실시할 경우 각 데이타 채널의 브로드캐스트 주기가 서로 다르기 때문에 특정 채널에서 데이타를 수신한 후, 다음 채널로 이동하였을 때 트랜잭션이 접근하는 데이타의 일관성이 깨질 수 있다. 이러한 이유 때문에 다중채널에서는 기존의 단일채널 트랜잭션 환경에서 사용되어 오던 채널당 브로드캐스트 단위로 서버측에서 제공하는 데이타들의 일관성은 유지되지 못한다. 따라서 다중채널에서의 트랜잭션 처리를 위해서는 기존의 브로드캐스트 사이클 단위 이외의 서버측에서 제공하는 데이타들의 일관성의 단위가 필요로 하게 된다.

이를 위하여 본 논문에서는 LBCPC라는 새로운 일관성의 단위를 제시하고 낙관적 동시성 제어기법인 검증기법을 통하여 다중 다중채널 환경에서 트랜잭션 처리를 위한 서버와 클라이언트 측의 동시성 제어기법을 제시해 보도록 하겠다. 다중채널 환경에서 트랜잭션을 처리하기 위하여 본 논문에서는 인덱스만을 방송하는 인덱스 전용채널과 데이타만을 방송하는 데이타 전용채널을 이용한다[7]. 본 논문에서 제시한 방법은 단일채널에서의 트랜잭션 처리 기법과 비교해 보아도 단일채널에서의 브로드캐스트 주기보다 본 논문에서 제안한 새로운 일관성의 단위가 상대적으로 많이 짧기 때문에 브로드캐스트 시스템의 평가 요소인 응답시간의 줄어들 뿐만 아니라 브로드캐스트 주기의 시작지점이 아닌 LBCPC의 시작 지점에서 제어 정보가 방송되어 읽기 트랜잭션의 최신성(currency) 또한 보장할 수 있다.

본 논문의 나머지 부분은 다음과 같이 구성된다. 2장에서는 기존의 단일 채널에서 연구되어온 여러 가지 동시성 제어 기법과 본 연구를 진행하게 된 동기, 본 논문에서 사용하는 다중 채널 환경모델에 대하여 설명하도록 하겠다. 3장에서는 다중 무선 채널에서 인덱스 채널 방송과 데이타 채널방송 그리고 다중채널 상에서의 동시성 제어 기법에 대하여 설명하도록 하겠다. 그리고 4장에서는 제안된 기법의 성능을 분석해 보도록 하겠다. 마지막으로 5장에서는 논문의 결론을 논의하겠다.

2. 관련 연구

본 장에서는 기존의 단일 채널에서 연구되어온 여러

가지 동시성 제어 기법과 본 연구를 진행하게 된 동기, 본 논문에서 사용하는 다중 채널 환경모델에 대하여 설명하도록 한다. 브로드캐스트의 기본적인 모델은 Acharya가 제시한 브로드캐스트 디스크(Broadcast Disks)와 같다[1]. 브로드캐스트 시스템은 크게 서버와 모바일 클라이언트로 구성된다. 서버는 주기적으로 데이타베이스의 데이타 아이템을 하나 혹은 그 이상의 무선 채널을 통해서 모바일 클라이언트에게 브로드캐스팅 한다. 그리고 모바일 클라이언트는 필요한 데이타가 있을 때, 해당 데이타가 브로드캐스팅 될 때까지 대기한 뒤 채널을 통해 수신한다. 따라서 모바일 클라이언트의 수가 증가해도 개별 데이타에 대한 접근 시간의 증가, 즉 성능의 저하는 발생하지 않는다. 이러한 장점으로 인해 모바일 트랜잭션은 경매, 전자 입찰과 같은 상거래 응용분야, 주식거래, 기상정보, 교통 정보 방송과 같은 다양한 응용분야에 적용된다. 또한 이러한 응용에서는 일관성 있는 데이타 관리를 위하여 트랜잭션 단위의 동시성 제어 기법이 필요하다. 예를 들어, 주식 거래에서 브로드캐스트 주기 C_i동안 트랜잭션 T_1이 특정 주식의 주식가격, 남은 수량 등에 대하여 업데이트를 수행하고 동일한 브로드캐스트 주기 C_i동안 트랜잭션 T_2가 동일한 주식에 대하여 주식가격을 읽었다고 가정하자. T_1에 의하여 변경된 정보는 C_{i+1}의 브로드캐스트 주기에 반영이 된다. 이때 트랜잭션 T_2가 C_i동안 읽은 주식 가격에 해당하는 주식의 남은 수량을 C_{i+1}동안에 읽게 되었다면 트랜잭션 T_2는 T_1에 의하여 업데이트되기 이전과 이후의 데이타를 읽음으로서 일관성을 유지하지 못하게 된다.

Shanmugasundarm은 무선 방송 환경에서의 동시성 제어기법은 상호 일관성과 최신성을 보장해야 한다고 하였다[8]. 여기서 상호 일관성은 서버가 일관성 있는 데이타를 내보내야 하고 클라이언트 또한 수행하고 있는 트랜잭션이 일관성 있는 데이타를 읽어야 한다는 것이다. 최신성은 클라이언트가 하나의 브로드캐스트 주기 시작 이후의 현재의 데이타를 읽어야 한다는 것으로 모바일 트랜잭션이 읽고 있는 데이타는 최신성을 유지해야한다는 것이다.

이를 위하여 서버는 데이타베이스 내의 데이타의 일관성을 유지할 책임을 가진다. 또한 하나의 브로드캐스트 주기 동안 브로드캐스팅 되는 데이타는 일관성을 가진다. 다시 말해 브로드캐스팅 되는 데이타는 해당 브로드캐스트 주기의 시작 이전의 데이타베이스의 상태만을 반영하고, 브로드캐스트 주기 도중 일어나는 모든 데이타에 대한 갱신은 다음 주기에 반영된다. 또한 서버와 클라이언트는 각각 읽기 전용 트랜잭션과 갱신 트랜잭션을 수행할 수 있다.

일반적으로 동시성 제어 기법은 낙관적 동시성 제어 기법과 비관적 동시성 제어 기법으로 나뉜다. 그런데 다양한 락(lock) 기반의 기법이나 타임스탬프 순서화(timestamp ordering) 기법과 같은 비관적 동시성 제어 기법은 위에서 언급한 무선 브로드캐스트 환경의 특성과 맞지 않는다. 첫째로 클라이언트와 서버 사이의 통신을 지나치게 요구한다. 예를 들어 락 기반의 기법을 사용하면 모바일 트랜잭션이 데이타 아이템에 대한 락이 요구될 때마다 클라이언트와 서버 사이의 통신을 필요로 한다. 이는 상향 대역폭을 사용하고, 전력을 소모하게 되므로 한정된 자원을 가진 모바일 클라이언트에게 부담이 된다. 둘째로 모바일 클라이언트는 무선 환경의 특성상 잦은 접속의 단절 발생할 수 있는데, 모바일 클라이언트가 단절되면, 그 클라이언트에서 수행 중이던 트랜잭션이 가진 락을 기다리는 다른 모든 모바일 트랜잭션들이 지연되므로 전체적인 응답 시간을 길어질 수 있다. 셋째로 모바일 브로드캐스트 환경의 많은 응용 분야의 경우 클라이언트의 수가 매우 많은데 이는 지나치게 많은 락 요구로 서버에게 과부하를 가져올 수 있다[5].

그에 비해 낙관적 동시성 제어 기법은 특성이 무선 브로드캐스트 환경과 잘 맞는다. 낙관적 동시성 제어 기법에서 모바일 트랜잭션은 검증 단계(validation phase)에 이르기 전까지는 서버에게 메시지를 보낼 필요가 없다[9]. 모바일 클라이언트는 서버가 브로드캐스팅하는 데이타 중에서 필요한 데이타를 수신하고, 그를 바탕으로 자신의 작업 공간(local workspace)에서 모바일 트랜잭션을 수행한다. 그리고 모바일 트랜잭션은 모든 실행이 끝나면 검증에 필요한 정보를 모아서 서버에 전송하고 최종 검증 단계를 거친다. 따라서 낙관적 기법은 비관적인 기법에 비해서 모바일 클라이언트에게 더 적은 통신비용을 요구하게 된다.

낙관적 동시성 제어 기법의 성능은 트랜잭션 재실행률에 크게 영향을 받는다. 트랜잭션의 재실행률이 높아지면 높아질수록, 모바일 클라이언트는 상향, 하향 통신 대역폭과 배터리를 소모하게 되고, 응답 시간도 길어지게 된다. 따라서 모바일 트랜잭션의 재실행률을 낮추고, 재실행시 발생하는 자원 소모를 줄이는 것이 프로토콜 디자인의 핵심이라고 할 수 있다.

위와 같은 동시성 제어 기법들은 단일채널을 기반으로 연구 되어왔다. 그러나 다중채널만을 위한 트랜잭션의 동시성 제어 기법에 대한 연구는 제시된 적이 없다. 본 논문에서는 다중 방송 채널 환경에서 읽기 트랜잭션을 위한 동시성 제어 기법을 제시하도록 한다.

이를 위하여 다중 채널환경에서 기존의 브로드캐스트 주기의 개념으로는 임의의 시점에서 서버에서 제공하는 데이타들의 일관성을 보장해 주지 못함을 보이도록 한다. 우선, 다중 채널환경서의 브로드캐스트 주기 개념에

대하여 살펴보도록 하자. 다중채널에서는 응답시간을 줄이기 위한 다중채널에서의 데이타 브로드캐스트 프로그램을 할당하는 연구들이 많이 제시되어 왔다[6,10-12]. 이렇게 생성된 다중채널에서의 데이타 브로드캐스트 프로그램은 각 채널당 브로드캐스트 주기를 생성하게 된다. 우리는 편의를 위하여 각 채널당의 브로드캐스트 주기를 아래와 같이 정의하도록 한다.

정의 1. k개의 다중 채널 중 임의의 알고리즘에 의하여 i번째 채널에 할당되는 완전히 스케줄링된 정보 프레임들을 Broadcast Cycle Per Channel(BCPC)라고 한다. 각 채널 i에서 방송되고 있는 데이타 노드의 개수를 BC, 각 데이타 노드의 크기를 Dsize라고 한다면 BCPC(i)는 $BC * D_{size}$ 이다.

기존 멀티 무선 방송 채널에서의 데이타 할당 방법으로 생성된 BCPC로 구성된 데이타 채널들은 임의의 시점에서 방송 되는 채널들 사이의 데이타들의 일관성을 보장해 주지 못한다.

그림 1은 클라이언트가 BCPC단위로 구성된 다중 채널을 듣고 있는 과정을 화살표를 통해 나타내고 있는 그림이다.

다중 브로드캐스트 채널 환경에서 클라이언트는 매순간 하나의 채널만을 들을 수 있다. 그러므로 그림 1에서처럼 채널1에서 첫 번째 BCPC의 내용을 들은 뒤 두 번째 BCPC의 중간에서 듣기를 시작했으므로 클라이언트는 대기 모드로 대기하다가 다음 BCPC의 시작부터 들어야 할 것이다. 이러한 구조는 채널1의 첫 번째 BCPC의 데이타와 채널2의 두 번째 BCPC의 데이타가 하나의 트랜잭션의 일관성을 요구할 경우 문제가 된다. 클라이언트는 채널1의 첫 번째 사이클을 들은 후 이미 읽은 데이타와 일관성이 있는 채널2의 첫 번째 BCPC

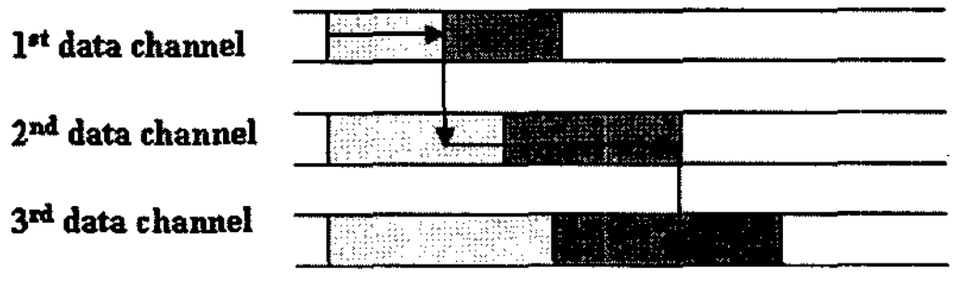

그림 1 BCPC 단위의 다중 채널을 클라이언트가 순차적으로 듣고 있는 경우

가 아니라 일관성이 다른 두 번째 BCPC를 들을 것이기 때문이다.

클라이언트가 그림 1과 같이 가장 온도가 높은 채널에서 낮은 채널로 순차적으로 데이타를 읽지 않고 최적의 채널 순서로 채널을 읽는 다고하더라도 기존의 BCPC을 기반으로는 일관성이 있는 데이타를 제공하지 않는다. 또한 클라이언트는 서버의 브로드캐스트 데이타가 어떠한 주기로 방송될지 모르기 때문에 원하는 데이타의 일관성을 유지하는 최적의 채널순서를 예측할 수 없다.

본 논문에서는 위와 같은 다중 채널에서의 문제점을 위하여 새로운 일관성의 단위를 제시하고 Lee의 [5]에서의 검증 기법을 기반으로 다중채널에서의 일관성 있는 트랜잭션 처리를 위한 동시성 제어 기법을 제안하도록 하겠다.

본 논문의 내용 이해를 돕기 위하여 다음의 예를 이용하여 앞으로의 제안 기법을 설명하도록 한다. 다중채널에서 브로드 캐스트 프로그램을 할당하는 많은 예제 중 우리가 제안하는 방법을 쉽게 설명할 수 있는 Prabhakara의 알고리즘을 이용하여 BCPC의 예를 들어 보도록 하자[6]. 그림 2는 Prabhakara의 방법을 이용한 데이타 채널에서의 브로드 캐스트 스케줄을 작성하기 위한 알고리즘이다. 알고리즘에서 각 데이타에 대한 온도(Temperature)는 그들에 대한 모바일 트랜잭션의 접

```
begin
/*INPUT :
  data temperature, data, the number of channel(k)
  Output :
  k-channel Data Broadcast Cycle
*/
  sort data according to the their temperature.
  average_channel_temperature = ∑(temperature of data in DB) / the number of data channels
   for(i=0; i<the number of data channels; i++){
      do
          allocate data in i_th data_channel;
      while ( ∑(temperature of data∈each channel)< average_channel
  _temperature);
   }
end
```

그림 2 다중채널에서 data broadcast cycle 생성 알고리즘

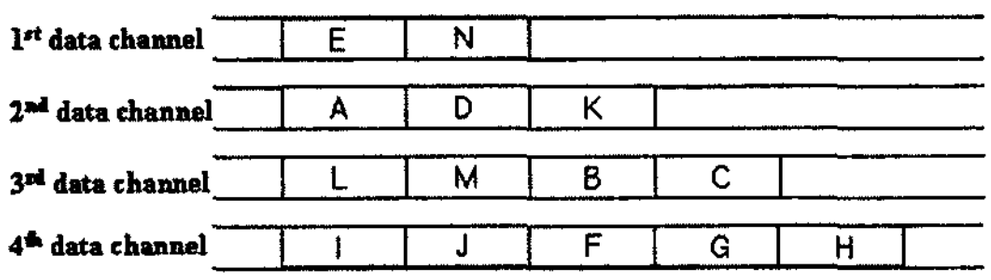

그림 3 다중 채널에 할당된 브로드캐스트 스케줄의 예

근 빈도(Access Frequency)를 나타낸다. 따라서 온도가 높다(Hot)는 것은 데이타에 대한 접근 빈도가 높다는 것을 의미하며, 차갑다(Cold)는 것은 상대적으로 접근 빈도가 낮음을 의미한다.

E, N, A, D, K, L, M, B, C, I, J, F, G, H와 같은 14개의 데이타와 4개의 채널이 주어질 경우 위의 그림 2의 알고리즘을 적용시켜 보자. 데이타는 E, N, A, D, K, L, M, B, C, I, J, F, G, H의 순서로 온도에 따라 정렬되어 있고 평균 채널 온도가 0.25이라고 가정한다. 이때 데이타 E와 N의 온도가 0.25와 같거나 약간 초과한다고 가정한다면 데이타 E와 N은 채널1에 할당될 것이다. 이와 같은 방법으로 채널 2, 3, 4에 [A, D, K], [L, M, B, C], [I, J, F, G, H]와 같이 방송될 것이다. 그림 3은 4개의 채널에 데이타가 할당되어 있는 모습을 보여준다.

각 데이타 노드의 크기, Dsize를 1이라고 가정한다면 그림 3을 통해 채널 1의 BCPC는 2, 채널 2의 BCPC는 3, 채널 3의 BCPC는 4인 것을 확인할 수 있다. 앞서 설명한 것처럼 그림 3의 BCPC로 구성된 브로드캐스트 스케줄은 일관성이 유지되는 정보를 내보내야 한다는 상호 일관성(mutual consistency)조건을 만족시키지 못한다.

다음 장에서는 다중 브로드캐스팅 채널 환경에서 읽기 트랜잭션 처리를 위한 동시성 제어 기법에 대하여 설명하도록 하겠다.

3. 다중채널 방송환경에서 읽기 트랜잭션 처리를 위한 동시성 제어기법

이 장에서는 우리가 제안하는 다중 채널 브로드캐스트 환경에서 읽기 전용 트랜잭션 처리를 위한 동시성 제어기법에 대하여 살펴보도록 한다. 이 기법은 앞장에서 설명한 낙관적 동시성 제어 기법인 검증기법과 새로운 일관성의 단위를 사용함으로서 다중채널에서의 읽기 트랜잭션의 일관성을 유지하도록 한다. 우선 이 장에서 제안하는 방법을 설명하기 전에 새로운 기법을 위해 필요한 가정들과 용어들을 살펴보도록 하겠다.

- 클라이언트는 한순간에 하나의 채널에만 접근이 가능하다.
- 하나의 채널에서 다른 채널로 변환할 때 걸리는 시간은 무시할 수 있다고 가정한다.
- 일반적으로 인덱스 노드의 크기는 데이타 노드의 크기보다 크다. 그러나 모든 데이타 노드들의 크기는 동일하며, 모든 인덱스 노드들의 크기는 동일하다고 가정한다.
- 클라이언트는 자신의 읽기 트랜잭션에서 필요한 데이타를 이미 모두 알고 있다.
- 각 채널에 할당된 데이타는 각 데이타 채널의 브로드캐스트 주기가 변하여도 변하지 않는다. 따라서 인덱스의 구조 또한 변경되지 않는다. 그러나 데이타의 값은 새로운 일관성의 주기가 바뀜에 따라 변경된다.

본 논문에서는 모바일 트랜잭션이 시작하기 전에 필요한 데이타 집합을 알고 있다는 가정과 [7]에서 제안한 인덱스 전용채널과 데이타 전용채널을 구분하는 다중채널 스킴을 이용한다. 이를 통하여 모바일 트랜잭션이 원하는 데이타의 위치를 알아내고 알아낸 위치가 트랜잭션내의 다른 데이타와 중복되는 경우 다음 주기의 데이타 위치를 다시 계산할 수 있다. 또한 인덱스가 데이타와 섞여 있을 때와 비교하여 빠른 응답시간을 얻을 수 있다.

본 논문에서는 다중채널에서의 읽기트랜잭션을 위한 동시성 제어기법을 설명하기 위하여 서버와 클라이언트의 작업을 설명하도록 한다. 3.1에서는 제안하는 다중채널 구조를 위한 인덱스 채널구조에 대해서 간략하게 설명하고 3.2에서는 서버에서 일관성 있게 데이타를 방송하는 방법에 대하여. 3.3에서는 클라이언트가 수행해야 하는 다중채널 상에서의 읽기 트랜잭션을 위한 동시성 제어 기법에 대하여 설명하도록 하겠다.

3.1 인덱스 채널 방송

다중 채널에서 또한 단일 채널과 마찬가지로 배터리의 소모를 줄이기 위하여 인덱스를 제공해야 한다. 이장에서는 [7]에서 제안한 인덱스 전용 채널 구조를 따른다. 이 인덱스 전용채널은 다중채널에 걸쳐 편향된 접근 선호도를 가진 방송 데이타에 대한 새로운 트리기반 인덱스 할당 방법을 이용한 것이다. 본 연구에서 사용하는 다중 채널 인덱스는 [7]에서 제안한 인덱스 채널 구조로서 알파베틱 허프만 트리 구조를 이용하여 인덱스만을 위한 전용 채널을 사용한다. 이때 알파베틱 허프만 트리에서 데이타 노드의 깊이는 그 데이타의 선호도가 반영되어 진다. 즉, 선호도가 낮은 데이타의 깊이는 높은 반면 선호도가 높은 데이타의 깊이는 낮을 것이다. 이러한 특징으로 트리의 낮은 곳에 위치한 데이타를 사용하는데 그렇지 못한 데이타보다 더 짧은 접근시간과 튜닝시간을 가질 것이다.

앞서 다중채널의 예제로 소개하였던 데이타 집합 {A, B, C, D, E, F, G, H, I, J, K, L, M, N}과 그들의 선

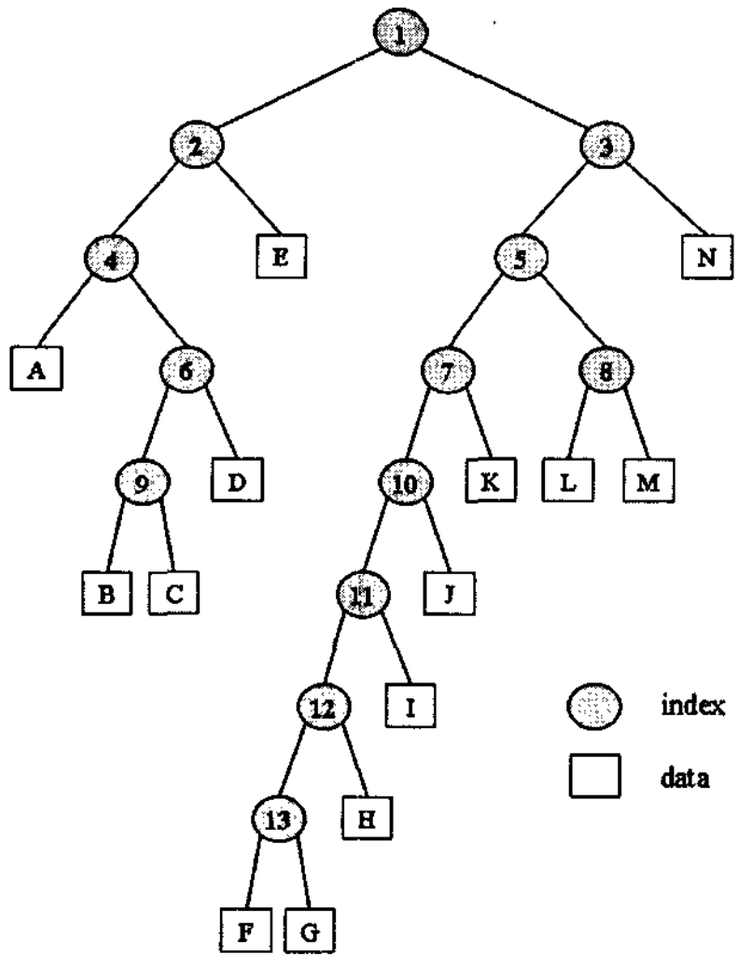

그림 4 알파베틱 허프만 트리의 예제

호도를 바탕으로 알파베틱 허프만 트리를 구성하면 그림 4와 같이 된다. 그림 4에서 인덱스의 set은 {1, 2, 3, ..., 11, 12, 13}이다. 그림 4에서 E와 N은 2장에서의 예제와 같이 가장 높은 선호도를 가지고 있으므로 트리의 깊이가 낮고 F와 G는 선호도가 가장 낮으므로 트리의 깊이가 높은 것을 확인할 수 있다.

이렇게 생성된 데이타에 대한 인덱스는 인덱스 전용 채널을 통해서 방송할 때 선호도가 높은 데이타에 대하여 더 자주 방송함으로서 모바일 클라이언트의 평균 접근 시간을 낮추는데 도움을 준다. 인덱스가 인덱스 전용 채널을 어떻게 구성하는지는 [7]을 참고하기 바란다.

이러한 인덱스 전용채널을 이용하여 데이타를 제공하는 것은 기존의 트리의 깊이만큼의 인덱스 채널을 필요로 하는 기법 [13]이나 데이타와 인덱스를 끼워 넣어 데이타와 인덱스의 크기가 같아야만 했던 단점을 보안할 수 있다. 또한 본 논문의 취지인 다중채널에서의 읽기 트랜잭션의 수행을 용이하게 한다. 데이타와 인덱스가 끼워져 있는 채널 구조는 채널들 사이를 오가며 트랜잭션을 수행해야 하는 클라이언트에게 일관성을 제공해주는 것을 더욱 어렵게 할 것이다.

인덱스 노드에서는 각 데이타가 방송되고 있는 채널의 채널번호, 자식 인덱스 또는 데이타 노드에 대한 시간 오프셋과 다음 제어정보의 시간 오프셋이 제공된다. 모바일 클라이언트는 이러한 정보를 이용하여 자신이 원하는 데이타의 위치를 알아내어 듣게 되고 다음 제어정보의 오프셋에 튜닝하여 검증을 수행한다. 모바일 클라이언트가 원하는 데이타를 인덱스채널을 걸쳐 어떻게 데이타채널에서 읽을지는 본장의 뒷부분에서 설명하도록 한다. 3.2장에서는 다중 무선 채널에서 데이타 채널을 방송하는 방법에 설명하도록 하겠다.

3.2 다중 무선 채널에서 데이타 채널 방송

2장에서도 언급한 것과 같이 무선 브로드캐스팅 환경에서의 동시성 제어 기법은 상호 일관성과 최신성이 보장되어야 한다[8]. 본 논문에서는 모든 채널에서 같은 BCPC의 데이타를 일정 시간만큼 반복하는 것과 제어정보(control information)를 통하여 읽기 트랜잭션의 상호 일관성과 최신성을 보장하도록 하겠다. 이를 위하여 다음의 용어를 정의하도록 한다.

정의 2. k개의 다중 데이타 채널에는 각 채널 i마다 서로 다른 길이의 BCPC(i)가 존재한다. 이때 $\max_{i \in 1,..,k}\{BCPC(i)\}$인 BCPC를 LBCPC(Longest Broadcast Cycle Per Channel)라고 한다. k개의 데이타 채널에서는 LBCPC의 길이만큼 각 i번째 채널의 BCPC(i)를 반복한다. 이때, $LBCPC \bmod BCPC(i) = 0 \quad \text{for } i = 1,k$ 일 필요는 없다.

정의 2에서 정의한 LBCPC는 모든 채널에서 적용되는 다중채널을 위한 새로운 일관성의 단위이다. LBCPC만큼 반복되어 방송되는 각 BCPC의 데이타는 앞서 가정에서 설명한 것과 같이 데이타의 아이디뿐만 아니라 값이 동일한 데이타이다. 모든 채널에서 동일한 LBCPC는 여러 개의 채널들을 이동하며 데이타를 듣고 있는 모바일 읽기 트랜잭션이 여러 채널에 걸쳐 트랜잭션내의 일관성을 유지하는데 도움을 준다.

다음은 기존의 낙관적 검증기법에서 사용되는 제어정보(Control Information)가 본 논문에서 어떻게 정의되는지 살펴보도록 하겠다.

정의 3. 연속적인 LBCPC C_i와 C_{i+1}이 있고 LBCPC C_i동안 서버에서 갱신 트랜잭션 $U_1...U_n$이 발생하여 k개의 데이타들이 업데이트 되어있다고 가정하자. LBCPC C_{i+1}의 시작시점에 방송되는 제어정보, $CI(C_{i+1})$는 이전 LBCPC인 C_i동안 서버에서 발생한 n개의 갱신트랜잭션에서 업데이트한 데이타의 아이디인 k개의 아이디를 가지고 있다. CI는 모든 채널에 걸쳐 매 LBCPC의 시작마다 방송된다.

제어정보는 모든 채널에서 방송되며 그 시점은 모든 채널에서 동일한 LBCPC의 시작 시점이다. 이는 모바일 읽기 트랜잭션의 검증을 위하여 제공되는 것으로 LBCPC와 함께 사용되어 본 논문의 목적인 다중 채널에서의 읽기 트랜잭션의 일관성을 유지하도록 한다.

모든 데이타 채널에 걸쳐 LBCPC동안 변경된 데이타의 양은 많을 것이다. 그러나 제어정보에서는 아이디만을 방송하게 됨으로 다음 LBCPC의 데이타를 읽기 위하여 기다리는 부하는 크지 않다.

그림 5는 다중무선 데이타 채널에서 데이타를 일관성 있게 방송하는 방법이다. 그림 5을 통해 알 수 있듯이

```
begin
/* INPUT:
   k=the number of channel;
   Dset_channel_ij=the set generated by figure 2 in channel i with j_th data position;
   OUTPUT:
   C_ij= data program with j_th data position allocated over i data channel for i=1,k;
*/
   for(i=1;i<=k;i++){
      for(j=1;j<=LBCPC;j++){
         temp = j mod BCPC(i);
         if(temp==0){temp=BCPC(i);}
         C_ij=Dset_channel_itemp;
      }
      broadcast CI over all of channels;
   }
end
```

그림 5 데이타 채널에서의 데이타 방송 방법

데이타는 LBCPC만큼 반복된다. 이때 반복되는 데이타는 업데이트가 반영되기 이전의 데이타로서 제어정보가 나오기 이전까지는 같은 BCPC의 데이타를 반복하여 방송한다. 또한 매 LBCPC마다 제어 정보, CI가 방송되는 것을 확인할 수 있다.

그림 6은 앞서 설명한 Prabhakara의 예제[6]를 바탕으로 그림 5의 알고리즘을 적용한 그림으로 앞서 설명한 LBCPC와 제어정보를 이용하여 데이타 채널을 구성한 모습이다. 채널1의 BCPC 길이는 2, 채널2의 BCPC 길이는 3, 채널3의 BCPC 길이는 4, 채널4의 BCPC 길이는 5이므로 BCPC 길이가 가장 긴 채널4의 BCPC 길이 5가 LBCPC가 된다. 나머지 채널들은 그림 6의 알고리즘에 의하여 LBCPC인 5만큼 같은 BCPC의 데이타를 반복하게 된다.

본 논문에서는 위와 같이 LBCPC와 검증(validation) 기법을 통하여 다중 채널에서의 상호 일관성을 보장해 준다. 제어 정보를 통한 검증기법은 낙관적 동시성제어 기법중 하나로서 2장에서 설명한 것과 같이 상향 대역폭의 사용을 줄어주고 서버의 부담을 줄어주면서 트랜잭션의 일관성을 보장해 주는 방법이다. 그러나 본 논문에서의 환경인 다중채널에서는 검증 기법만으로는 일관성을 보장해주기는 어렵다. 기존의 방법대로 BCPC의 시작 시점에서 제어 정보를 내보내어 검증을 할 경우 각각의 채널에 할당된 BCPC의 길이가 다르기 때문에 여러 채널에 걸쳐 데이타를 읽어야 하는 트랜잭션은 제어 정보만으로 일관성이 있는 데이타를 읽을 수 없다. 그러므로 모든 채널에 걸쳐 동일한 기간인 LBCPC마다 제어 정보를 내보내어 검증함으로서 트랜잭션이 접근하는 데이타의 일관성을 보장해 줄 수 있다. 다음 3.3장에서는 본장에서 제공하는 서버의 브로드캐스팅 환경에 맞는 다중 채널 상에서의 동시성 제어 기법에 대하여 설명하도록 하겠다.

3.3 다중채널 상에서의 동시성 제어 기법

이번 장에서는 원하는 데이타들을 인덱스채널로부터 데이타채널을 통해 읽는 방법에 대하여 설명하도록 하겠다. 3장 도입부에서 가정한 것과 같이 클라이언트는 읽기 트랜잭션에서 필요로 하는 데이타를 트랜잭션 실행 이전에 모두 알고 있다. 따라서 클라이언트는 트랜잭션 수행을 위해 데이타 채널을 듣기 이전에 인덱스 채널에서 읽기를 원하는 데이타 노드의 오프셋을 알아낸 후 데이타 채널을 통해 데이타를 읽을 수 있다. 3.1장에서도 설명했듯이 인덱스채널의 인덱스 노드는 해당 데

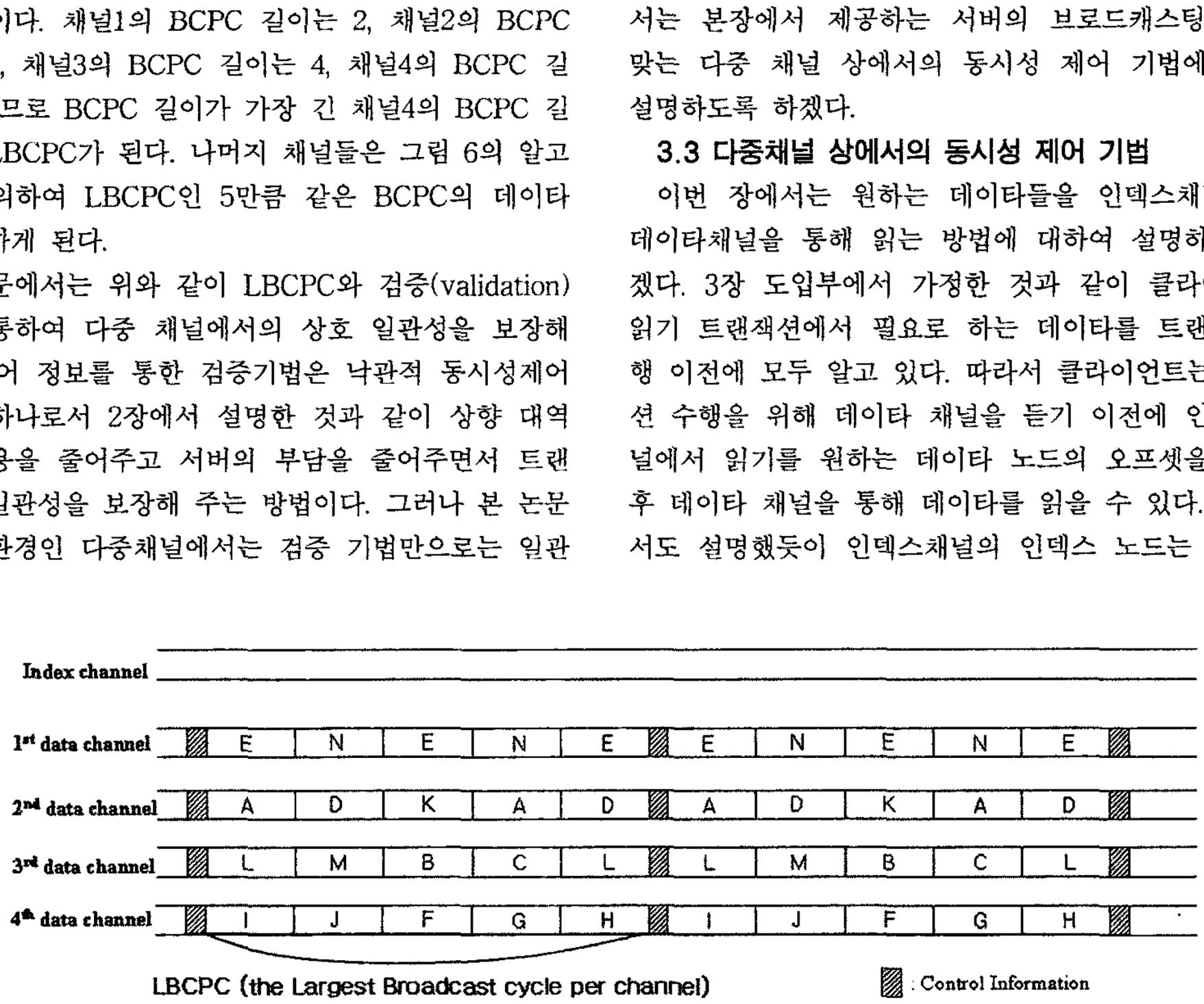

그림 6 서버 측에서 방송되는 인덱스와 데이타 구조의 예

이타의 채널 번호와 자식 인덱스 또는 데이타 노드에 대한 오프셋, 다음 제어 정보의 오프셋 정보를 가지고 있다. 또한 인덱스 노드가 데이타 노드를 가리키고 있는 경우 이 인덱스 노드는 각 채널에서 방송되고 있는 데이타 노드의 개수를 의미하는 BC정보를 가지고 있다.

클라이언트가 임의의 시점에서 트랜잭션을 실행하게 되면 위와 같이 인덱스 전용 채널을 통하여 데이타의 위치를 파악한 후 데이타 채널을 읽게 된다. 이때 클라이언트는 인덱스 채널을 통해 얻은 원하는 데이타의 오프셋을 이용하여 현재의 튜닝시점에서 가장 빨리 들을 수 있는 각 데이타의 위치를 계산하게 된다.

데이타의 위치를 계산하는 방법은 크게 두 가지로 구분할 수 있다. 트랜잭션이 접근해야하는 데이타의 집합이 하나의 LBCPC안에서 접근이 가능할 경우와 여러개의 LBCPC안의 데이타를 접근해야할 경우로 구분될 수 있다. 두 경우 모두 계산 방법으로 아래의 정의를 이용한다.

정리 1. 임의개의 인덱스 전용채널과 k개의 데이타의 전용채널이 존재하는 환경에서 제어정보, CI를 들은 횟수를 CCN, 제어정보의 사이즈를 |CI|, 하나의 LBCPC안에서 반복되는 BCPC의 횟수를 BN, BCPC내의 상대적인 위치를 오프셋이라고 가정하자. 모바일 읽기 트랜잭션이 접근하려는 데이타 x가 i번째 채널에서 방송되고 있다면 계산된 position은 $\{CCN*(LBCPC+|CI|)\}+(BCPC(i)*BN)+OFFSET$ 이다.

증명. 데이타 전용 채널에서는 CI를 방송하고 LBCPC의 길이만큼의 데이타를 방송한다. 따라서 k개의 데이타 채널에서는 BCPC(i)와 상관없이 LBCPC+|CI|의 길이를 반복하여 방송할 것이다. 이때 수행되고 있는 트랜잭션이 제어정보를 들은 횟수를 CCN이라고 하였으므로 LBCPC 단위의 시간은 $\{CCN*(LBCPC+|CI|)\}$이 된다. 한 LBCPC내부에서는 BN번만큼의 BCPC(i)가 반복되므로 $BCPC(i)*BN$으로 BCPC의 위치를 찾을 수 있다. 또한 하나의 BCPC내의 데이타는 인덱스 채널에서 제공하는 특정 데이타의 OFFSET로 찾을 수 있으므로 특정 데이타 x의 다음 방송 시간을 계산하는 position값은 $\{CCN*(LBCPC+|CI|)\}+(BCPC(i)*BN)+OFFSET$ 이 된다. □

그러나 정리 1은 하나의 LBCPC안에서 하나의 트랜잭션이 접근하는 데이타집합의 위치를 계산하는 경우에는 CCN값이 0이 됨으로 position의 값은 $(BCPC(i)*BN)+OFFSET$이 된다. CCN이 0이라는 것은 클라이언트가 임의의 시점에 인덱스 채널로부터 데이타 채널에 튜닝하였다는 것이다. 따라서 BN은 ⌊current_index_position/DSize⌋ 로 결정되어야 한다. 이때 current_index_position은 인덱스 노드를 들은 이후의 시간을 가리킨다. 모바일 트랜잭션이 실행하게 되면 앞서 설명한 것과 같이 인덱스 채널로부터 필요한 정보를 읽고 데이타 채널에 튜닝하게 된다. 이때 트랜잭션은 하나의 LBCPC 안에서 데이타 집합의 위치를 계산하게 되므로 $(BCPC(i)*BN)+OFFSET$의 식을 이용하여 데이타 집합의 위치를 계산하게 된다.

여러 LBCPC에 걸쳐 원하는 데이타 집합의 위치를 계산하는 경우는 CCN의 개념을 포함한 정리 1에서 정의한 position 계산식인 $\{CCN*(LBCPC+|CI|)\}+(BCPC(i)*BN)+OFFSET$를 사용하게 된다. 이때 CCN과 BN은 인덱스 채널에서 제공하는 제어정보 오프셋을 지날 때마다 CCN은 하나씩 증가해야하고 BN은 0으로 초기화되어야 한다.

위와 같이 계산된 트랜잭션이 수행해야 하는 데이타들의 실제 위치는 여러 데이타가 같은 시간을 가르칠 수도 있다. 이러한 상황을 용어로 정의해 보도록 하겠다.

정의 4. 임의개의 다중 데이타채널로 구성된 환경에서 트랜잭션 T가 접근하려는 n개의 데이타들이 c개의 데이타채널($1 \leq c \leq k$)에 분산되어 있다고 가정하자. 이때 정리 1의 계산식에 의하여 계산된 데이타들의 position 값이 같은 위치를 가리키고 있는 데이타가 2에서 n개라면 정리 1에 의해서 계산된 2에서 n개의 데이타의 실제 위치가 겹쳤다(overlap)라고 한다.

이장 처음의 가정에 따라 모바일 클라이언트는 하나의 데이타 채널만을 들을 수 있으므로 모바일 클라이언트가 접근하는 데이타의 실제 위치가 겹치게 된다면 둘 중 하나의 데이타를 포기해야만 한다. 따라서 데이타의 위치가 겹칠 경우 모바일 클라이언트는 데이타의 다음 방송 위치를 재계산해야 한다. 재계산 방법 역시 정리 1의 계산식을 이용한다. 이때 읽어야 하는 데이타의 위치가 겹친다면 위치가 같은 데이타들 중 가장 긴 BCPC을 가진 채널의 데이타를 먼저 읽도록 한다. 위치가 겹치는 나머지 데이타들은 정리 1의 계산법에 의해 새로운 방송 위치를 찾게 된다. 여기서 데이타들 중 가장 긴 BCPC을 가진 데이타에게 읽을 우선권을 먼저 준 이유는 BCPC가 상대적으로 짧은 채널의 데이타의 위치를 다시 계산하는 것이 긴 BCPC을 가진 채널의 데이타를 찾을 때보다 더 빠른 응답시간 안에 원하는 데이타의 집합을 찾을 수 있기 때문이다.

그림 7은 인덱스 노드로부터 모바일 트랜잭션이 접근해야 하는 데이타 x를 접근하는 방법이다.

모바일 읽기 트랜잭션은 그림 7과 같은 방법으로 데이타 채널에서 방송되는 데이타에 대하여 작업을 수행하게 된다. 작업 수행 중 클라이언트 측에서 트랜잭션 수행 중에 제어정보(control information)를 듣게 되었다면 validation을 수행하게 된다. 검증(validation) 방법

```
begin
/* Input:
  CI_position=time offset of Control Information;
Output:
  position=the current time position of data node x over data channel i from index node y;
*/
  if(current_position pass CI_position)
  {
      CCN++;
      BN=0;
  }
  BN++;
  calculate position by Theorem 1;
  if(position is overlapped)
  {
      give priority to channel having the longest BCPC;
      while(position is overlapped || current_index_positon>position)
      {
            BN++;
            recalculate position by Theorem 1;
      }
  }

end
```

그림 7 인덱스 노드로 부터 데이타 노드를 접근하는 방법

은 현재 수행하고 있는 읽기 트랜잭션 T_{PV}중 아직 읽지 못한 데이타의 아이디 집합 $CRS(T_{PV})$가 제어 정보 안에 있는 이전 LBCPC C_i동안 변경된 데이타의 아이디의 집합$CI(C_i)$과 공통된 데이타 아이디를 가지고 있는지 확인한다. 이때 만약 공통된 데이타를 가지고 있지 않다면 트랜잭션을 계속실행하고 만약 공통된 아이디가 있다면 트랜잭션을 중지시키고 재실행한다. 제어정보에 있는 데이타의 아이디는 이전의 LBCPC 동안에 변경된 데이타를 나타내는 것이므로 이전 LBCPC와 진행될 LBCPC 동안 데이타의 일관성이 같지 않다는 것을 의미한다. 따라서 수행은 중지되고 다음 LBCPC에서 재실행 되어야 한다.

본 논문에서는 앞서 설명한 것과 같이 트랜잭션의 검증과정은 제어정보로 제공되는 $CD(C_i)$의 아이디와 현재 실행되고 있는 트랜잭션의 아직 읽지 못한 데이타의 아이디 집합을 비교하는 것이다. 이는 기존 연구에서 언급하였던 Lee의 이미 읽은 데이타와 제어정보의 아이디를 비교하는 것과 차이를 보인다[5]. 이것은 본 논문에서 제시한 트랜잭션 실행 전에 수행에 필요한 데이타를 알고 있다는 가정 때문에 생긴 차이로서 성능에 영향을 미치지 않는다. 그림 8은 모바일 트랜잭션의 검증과정을 간략하게 나타낸 알고리즘이다.

정리 2. 임의개의 인덱스 전용채널과 임의개의 다중 무선 채널 환경에서 임의의 모바일 읽기 전용 트랜잭션 T가 필요로 하는 데이타의 일관성은 트랜잭션 T가 수행되면서 듣게 되는 $C_i, C_{i+1}, ..., C_n$의 LBCPC와 $CI(C_i)$, $CI(C_{i+1}), ..., CI(C_n)$를 통한 검증(validation) 과정으로 유지될 수 있다(i는 임의의 수, n은 무한히 큰 수).

증명. 모바일 클라이언트의 트랜잭션에서 필요로 하는 데이타는 여러 채널에 걸쳐 방송된다. 모바일 클라이언트가 튜닝한 시점이 C_i번째 LBCPC의 데이타 노드라면 C_i 번째의 LBCPC 안의 데이타를 읽기 시작할 것이다. 만약 모바일 클라이언트의 트랜잭션이 필요로 하는 데이타를 C_i 번째 LBCPC 안에 모두 읽게 된다면 트랜잭션은 커미트될 것이다. 그러나 원하는 데이타를 C_i 번째 LBCPC 안에 다 읽지 못했다면 다음 $CI(C_{i+1})$를 확인하여 제어 정보 안에 자신이 읽어야할 데이타의 아이디가 존재한다면 현 트랜잭션이 이미 읽은 데이타의 일관성과 앞으로 읽어야 할 C_{i+1}번째 LBCPC의 데이타의 일관성이 다르므로 트랜잭션은 실패되고 재실행될 것이다. 이러한 방법으로 트랜잭션 T가 계속 재실행이 일어난다고 하더라도 무한히 큰 수인 n에 따라 Cn번째 LBCPC에서는 성공하게 될 것이다. 만약 제어정보 안에 자신이 읽어야 하는 데이타의 아이디가 존재하지 않는다면 트랜잭션은 계속 실행되어 C_{i+1}번째의 LBCPC의 데이타를 읽을 것이다. 따라서 모바일 클라이언트 T의 일관성은 유지될 것이다. □

지금까지 설명한 동시성 제어 기법을 예를 통해 설명해 보도록 하자. 그림 9의 스케줄을 따르는 아래의 일련의 트랜잭션 집합을 고려해 보자.

서버 갱신 트랜잭션 U_1 : r(A)w(A)r(D)w(D)

모바일 읽기 트랜잭션 T_2 : r(N)r(E)r(D)r(L)

클라이언트는 제일 먼저 인덱스를 통하여 원하는 데이타 {N,E,D,L}에 대한 데이타 채널번호와 실제 방송

```
begin
/*input:
TPV : the validating mobile transaction
CD(Ci) : the set of data objects that have been commited(updated) in the last LBCPC
Ci over whole channels at the server
CRS(TPV) : the set of data objects that did not read by TPV from previous LBCPC
*/
validation(TPV)
{
    if CD(Ci)∩CRS(TPV)≠{}
        abort(TPV);
    else
    {
        record the value of Ci;
        TPV is allowed to continue;
    }
}
end
```

그림 8 트랜잭션의 validation 과정

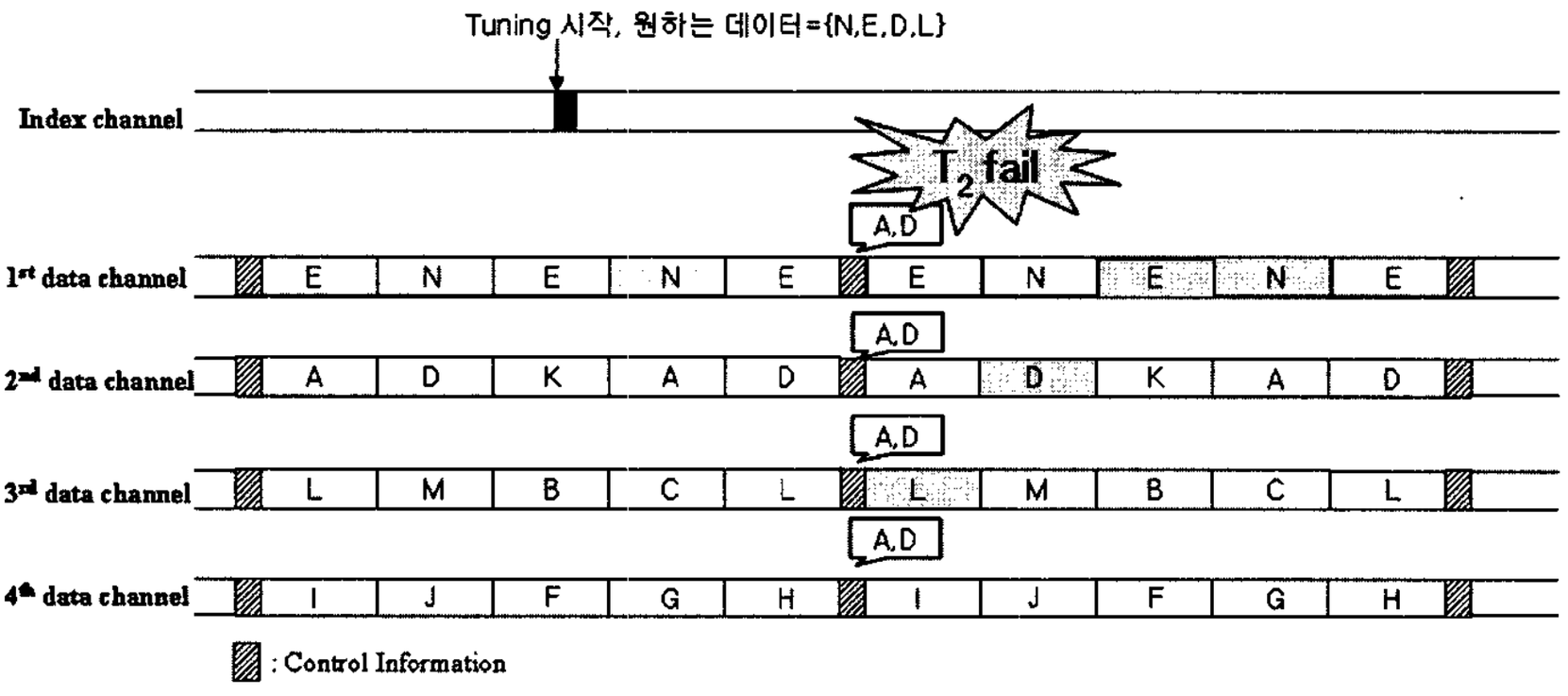

그림 9 트랜잭션 수행과정의 예제

위치 오프셋, 다음 제어정보의 오프셋을 알아낸다. 구체적으로 그림 7의 다중채널 상에서 모바일 트랜잭션의 접근방법 알고리즘에 따라 트랜잭션 T_2의 데이타의 위치 오프셋을 구해 보면 다음과 같다.

일단, 튜닝 시점(current_position)을 25, Bcast(1)=2, Bcast(2)=3, Bcast(3)=4, 그리고 데이타 노드의 크기 Dsize를 10이라고 제어 정보의 크기인 CI_size를 1이라고 가정해보자. 트랜잭션 T_2의 데이타 N, E, D, L의 데이타 오프셋은 다음과 같다.

N = 20*1+10 = 30

E = 20*2+0 = 40

D = 30*1+10 = 40

L = 40*1+0 = 40

위의 계산 된 데이타 오프셋을 살펴보면 N은 30으로 클라이언트가 가장 먼저 데이타를 읽게 되고 E,D,L은 40으로 데이타의 시간 위치가 겹치는 것을 확인할 수 있다. 그러나 멀티 다중채널의 특성상 클라이언트는 하나의 채널만을 들을 수 있다. 따라서 그림 7의 알고리즘에 따라 가장 긴 BCPC을 갖고 있는 3번째 채널의 L을 타임 오프셋 40에 먼저 읽고 CI 검증(validation) 검사 후 E=51+20*0+0=51, D=51+30*0+10=61 시간에 데이타를 읽게 된다. 만약 E와 D의 위치가 또 다시 겹친다면 알고리즘에 따라 위치가 겹치지 않을 때까지 위치를 다시 계산하게 된다.

트랜잭션 T_2의 실행도중 시간 30에 N과 시간 40에 L를 읽은 다음 시간 50에 제어 정보를 읽게 된다. 클라이언트는 인덱스 채널에서 얻은 제어 정보 offset을 통해 제어 정보를 읽게 된다. 이때 제어 정보는 서버에서 일어난 갱신 트랜잭션 U1의 데이타 아이디가 {A, D}이므로 50 시점에서 방송되는 제어 정보는 {A, D}가 된다.

이렇게 방송된 제어 정보는 수행되고 있는 트랜잭션 T_2가 앞으로 읽기 작업을 수행해야 하는 데이타 집합 {E, D}와 겹치게 됨으로 그림 8의 과정에 따라 트랜잭션 T_2는 실패하고 다음 LBCPC에서 다시 시작하게 된다.

그림 9는 트랜잭션는 T_2가 실패하고 다음 LBCPC에서 다시 시작하여 T_2의 트랜잭션을 다시 실행하는 모습을 그림으로 나타낸 것이다. 다시 시작한 T_2 트랜잭션은 앞서 한 것과 같은 그림 7의 과정을 다시 실행하게 된다. 이에 따라 {N,E,D,L}의 데이타 위치는 다음과 같다.

N = 51+20*0+10 = 61

E = 51+20*0+0 = 51

D = 51+30*0+10 = 61

L = 51+40*0+0 = 51

이때 N,D와 E,L의 시간 위치가 겹치므로 알고리즘에 따라 L,D가 제 시간에 수행되게 되고 E와 N은 재계산을 통한 시간 값을 얻어 낸다. 결국 트랜잭션 T_2는 E와 N을 E = 51+20*1+0 = 71, N= 51+20*1+10 = 81의 시간에 데이타를 읽을 수 있다.

트랜잭션 T_2가 중단되고 재실행 되었다고 하더라도 단일채널에서의 제어 정보를 통한 모바일 읽기 트랜잭션의 검증방법보다는 빠른 응답시간을 보일 것이다. 왜냐하면 본 논문에서의 LBCPC는 단일채널에서의 브로드캐스트 주기길이보다 훨씬 작기 때문에 단일채널에서의 재실행으로 인한 응답시간 증가와 비교하여 보았을 때 응답시간은 훨씬 짧아진다. 또한 데이타의 편향성이 심해짐에 따라 응답시간이 더욱 짧아 질 수 있다.

4. 성능 분석

본 논문에서는 다중 채널에서의 읽기 트랜잭션 처리를 위한 동시성 제어 기법을 제안하였다. 다중채널에서 트랜잭션을 처리하는 것은 기존에 제시된 적이 없는 문제로서 본 논문에서 처음으로 제시한다고 본다고 해도 무리가 없을 것이다.

본 장에서는 몇 가지 실험을 통해 본 논문에서 제시하는 다중 채널에서의 읽기 트랜잭션 처리 기법이 효과적임을 보인다. 단일채널에서의 Lee가 제안한 검증을 이용한 낙관적 동시성 제어 기법[5]과 비교를 통해서 본 논문에서 제안하는 방법의 성능을 분석한다. 앞서 말한 것과 같이 다중 채널에서의 트랜잭션 처리 기법이 존재하지 않기 때문에 본 논문에서의 대조군은 단일 채널에서의 Lee가 제안한 검증 기법을 이용할 것이다. 앞으로는 편의를 위해서 이 논문에서 제안하는 기법을 TPMC (Transaction Processing in Multi Channel), Lee가 제안한 동시성 제어 기법을 FBOCC로 쓰도록 하겠다. TPMC가 선호도를 반영한 다중채널 환경에서의 트랜잭션 처리 기법이므로 FBOCC 또한 단일채널에서 선호도를 반영한 Acharya가 제안한 멀티디스크 방법[1]으로 브로드캐스트를 구성하였다.

본 논문의 모든 실험은 Intel Pentium(R) 3.00GHz 프로세서와 1GB 메모리상에서 시뮬레이션 되었다. 실험 변수 중에 하나인 모바일 클라이언트들의 비정규 분포의 접근 패턴을 모델링하기 위해서 매개 변수 θ를 갖는 Zipf 분포를 사용하였다. Zipf 분포는 모바일 브로드캐스트 환경에서 모바일 클라이언트의 편향된 분포의 접근 패턴을 모델링 하는데 널리 사용된다.

기본적으로 모델은 모바일 클라이언트, 서버, 그리고 인덱스 정보를 내보내기 위한 인덱스 채널과 데이타와 제어정보를 전송하기 위한 데이타 채널의 브로드 캐스트 디스크로 구성된다. 모바일 클라이언트 측에서는 읽기 전용트랜잭션이 발생한다. 서버 트랜잭션은 읽기 작업과 갱신 작업을 아래의 비율로 수행하게 된다. 그리고 트랜잭션은 커미트 될 때 까지 계속되고 마감시간을 고려하지 않는다.

앞으로의 실험은 각 실험에서 변화를 주어야 하는 변수와 변화를 주어야 하는 변수 이외의 고정 되어야 하는 기본 변수 모두 위의 표 1을 바탕으로 할 것이다. 본 논문에서는 트랜잭션의 응답 시간을 성능 지표로 삼았다. 여기서의 응답시간은 접근시간(Acess time)을 말하는 것으로서 클라이언트가 원하는 데이타를 얻기 위하여 브로드 캐스트 채널에 처음 탐색(initial probe)을 시작한 순간부터 최종적으로 원하는 데이타를 획득할 때까지의 총 시간을 말한다. 결국 클라이언트가 데이타를 획득하기 위해서는 접근시간(access time) 만큼을 소비해야 한다. 이 때문에 접근 시간은 브로드 캐스트 프로그램의 성능을 측정하는 데에 사용된다. 이장에서 측정의 단위는 비트 시간(bit-time)을 기준으로 하였다. 예를 들어 하나의 서버가 하나의 데이타 아이템을 브로드캐스팅 하는 데는 8,000비트 시간이 소요된다. 또한 인덱스의 크기는 데이타 크기의 0.1배라고 가정하였다. 따라서 위와 같이 하나의 데이타 아이템을 브로드캐스팅 할 때 8,000비트의 시간이 소요된다면 인덱스 전용채널에서는 하나의 인덱스 아이템을 브로드캐스팅 하는데 800비트의 시간이 소요될 것이다.

본 실험에서는 채널수 실험을 제외한 모든 시뮬레이션에서 인덱스 채널을 1, 데이타 채널을 3으로 설정한 이유는 대조군인 FBOCC가 단일 채널을 바탕으로 하였기 때문에 본 논문의 환경에서 평균 응답시간이 가장 나쁜 환경을 채택하도록 하였다. 4.5장을 통해서도 알 수 있듯이 인덱스 채널이 1일 때 가장 큰 응답시간을 낼 수 있다. 따라서 본 논문은 다중채널에서의 최대의 응답시간을 낼 수 있는 환경에서의 TPMC의 결과를 단일채널에서의 FBOCC와 비교하도록 한다.

표 1 시뮬레이션 매개변수

시뮬레이션 매개 변수	범위
일반	
시뮬레이션 횟수	10,000
FBOCC의 브로드캐스트 디스크의 수	3
Zipf 매개 변수 θ	0.0~1.0
인덱스 채널의 갯수	1
데이타 채널의 갯수	3
데이타베이스 내의 데이타의 수	1000
데이타 오브젝트의 크기	8000 bits
모바일 트랜잭션(모바일 클라이언트)	
트랜잭션의 길이(연산의 수)	8
읽기 전용 트랜잭션의 비율	1
연산 사이의 평균지연시간	65,536 bit-times(exponentially distributed)
트랜잭션 사이의 평균지연시간	131,072 bit-times(exponentially distributed)
접근 불변성의 정도	0.95, 0.0~1.0
서버 트랜잭션(서버)	
트랜잭션의 길이	8
트랜잭션의 도착 빈도	1 per 1,000,000 to per 333,333 bit-times
읽기 연산의 확률	0.5
동시성 제어 기법	OCC with forward validation

다음 장에서 접근 패턴의 편향성, 서버 데이타베이스 내의 데이타의 수, 서버트랜잭션 갱신률, 모바일 트랜잭션이 수행하는 데이타의 개수, 인덱스 채널과 데이타 채널의 비율에 따른 평균 응답시간 구하여 성능을 분석하도록 한다.

4.1 접근 패턴의 편향 정도에 따른 성능

일반적으로 다중 무선 채널에서의 브로드 캐스팅 기법은 선호도를 반영하여 각 채널의 데이타 개수를 달리하는 것이다. 이에 따라 데이타 아이템에 대한 접근 패턴이 편향될수록 응답시간이 단축되는 장점을 가지고 있다. 이 장에서는 Zipf 매개 변수 θ를 0.1에서 1.0까지 증가 시키면서 TPMC와 FBOCC의 접근 패턴의 변화에 어떻게 적응하는지를 분석한다. Zipf 매개 변수의 값이 크면 인덱스 채널에서는 선호도가 높은 데이타에 대하여 더 자주 인덱스를 내보내게 되고 데이타 채널에서는 채널들 간의 할당된 데이타 개수의 차이가 많아지게 된다. 또한 트랜잭션의 접근 패턴 또한 편향되게 된다.

그림 10은 접근 편향성을 나타내는 zipf factor에 따른 TPMC와 FBOCC의 응답시간을 나타낸 것이다. FBOCC는 앞에서도 설명했듯이 다중채널을 기반으로 하는 TPMC와의 비교를 위하여 브로드캐스트 스케줄링을 선호도가 반영되는 Acharya가 제안한 멀티디스크 방법[1]으로 구성하였다. 따라서 zipf factor가 증가함에 따라 응답시간이 감소하는 것을 확인할 수 있다. TPMC는 기본적으로 다중 채널을 기반으로 하는 것이기 때문에 접근 편향성에 대하여 zipf factor가 커짐에 따라 응답시간이 감소하는 것을 확인할 수 있다.

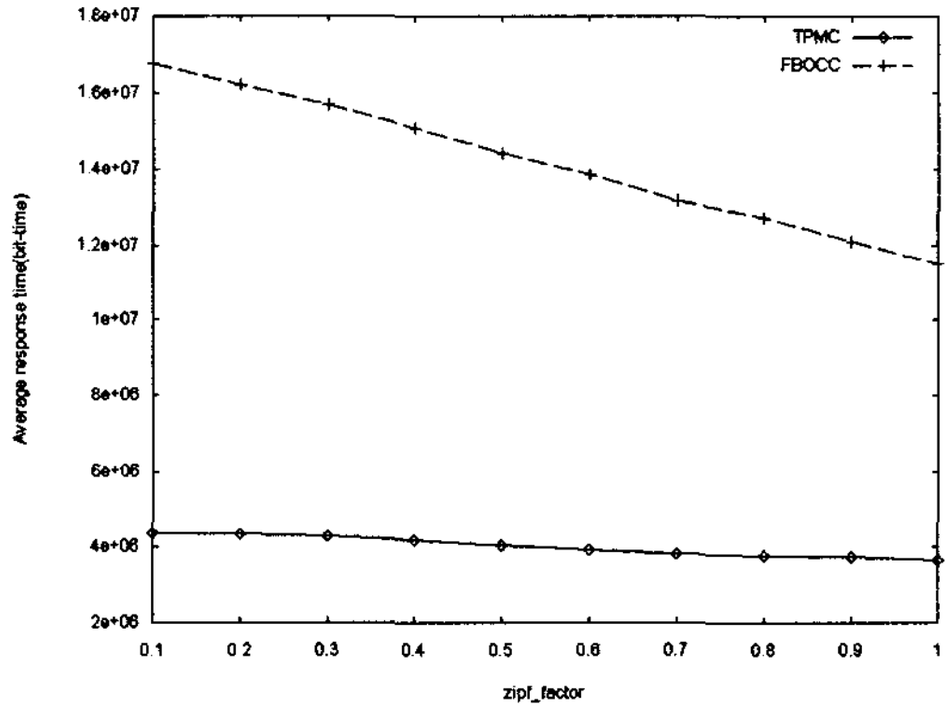

그림 10 접근 편향성에 따른 평균 응답 시간

그림 11은 접근 편향성이 변화에 따라 TPMC와 FBOCC의 실행중 중지된 트랜잭션의 개수를 타나낸 그림이다. 그림에서 확인할 수 있듯이 TPMC와 FBOCC 모두 zipf factor가 커짐에 따라 데이타의 충돌이 커지게 되어 중지되는 트랜잭션의 개수가 많아지는 것을 확인할 수 있다. 또한 TPMC의 트랜잭션 중지 개수가 FBOCC에 비하여 상당히 많은 것을 확인할 수 있는데 이는 다중채널에서의 LBCPC가 단일채널에서의 브로드캐스트 주기보다 짧기 때문이다.

그러나 그림 10과 그림 11을 종합하여 보았을 때 그림 11과 같이 TPMC의 트랜잭션의 중지 개수가 많음에도 불과하고 TPMC의 응답시간이 FBOCC의 응답시간보다 확연히 짧은 이유는 LBCPC가 브로드캐스트 주기보다 짧기 때문에 재실행을 위한 대기시간이 확연히 짧

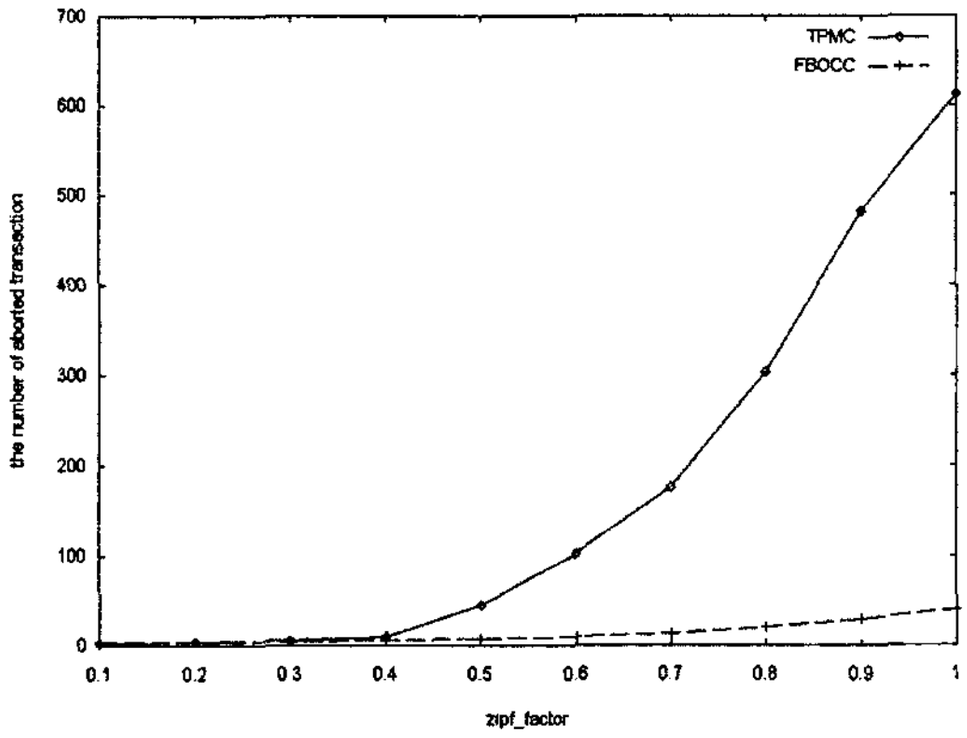

그림 11 접근 편향성에 따른 중단된 트랜잭션의 개수

아 졌기 때문이다.

4.2 서버 데이타베이스의 데이타 개수에 따른 성능

본 장에서는 서버에서 유지하고 있는 데이타베이스의 데이타 개수에 따라 변화하는 TPMC와 FBOCC의 성능을 비교해 보도록 하겠다. 서버에서 방송해야하는 데이타의 양이 커진다는 것은 모바일 트랜잭션 입장에서 필요로 하는 데이타를 얻는데 많은 시간이 걸린다는 것을 의미한다. 왜냐하면 필요로 하는 데이타를 얻기 위해서는 필요하지 않은 데이타를 기다려야 하는데 데이타의 개수가 많아지면 필요하지 않은 데이타의 개수 또한 증가할 것이기 때문이다.

그림 12는 zipf factor값이 0.1일 때와 0.95일 때를 기준으로 데이타베이스의 데이타 개수에 따른 TPMC와 FBOCC의 응답시간을 나타낸 그림이다.

그림 12를 통해 알 수 있듯이 데이타베이스의 데이타 아이템의 개수가 증가함에 따라 TPMC와 FBOCC의 평균 응답시간 모두 증가 하는 것을 확인할 수 있다. 또한 데이타 아이템의 개수가 증가함에 따라 TPMC는 FBOCC에 비하여 좋은 성능을 보이고 있다. 이는 TPMC의 LBCPC가 FBOCC의 브로드캐스트 주기보다 짧기 때문에 원하는 데이타를 얻는데 필요한 응답시간이 짧아지기 때문이다.

동일한 TPMC나 FBOCC기법 일지라도 트랜잭션이 원하는 데이타가 편향되지 않고 유니폼(uniform)할 때 즉, 접근 편향성을 나타내는 Θ가 0.1일 때의 평균 응답시간은 Θ가 0.95인 TPMC나 FBOCC의 평균 응답시간보다 오래 걸리는 것을 확인할 수 있다. 이는 4.1에서 확인한 것과 같이 모바일 트랜잭션의 데이타 패턴이 편향될수록 응답시간이 짧아지기 때문이다.

4.3 서버 트랜잭션의 갱신률에 따른 응답시간

본 장은 서버 트랜잭션의 갱신률에 따른 TPMC와 FBOCC의 응답시간을 비교함으로서 TPMC의 성능을 증명하도록 한다. 서버 트랜잭션은 클라이언트에서 발생하는 모바일 트랜잭션과 달리 업데이트가 발생한다. 이때 업데이트가 발생하는 비율을 0부터 1까지로 설정하여 0일 때는 읽기 전용으로 1일 때는 쓰기 전용 트랜잭션이 되도록 하였다.

그림 13은 TPMC와 FBOCC의 서버 트랜잭션 갱신 비율에 따른 평균 응답시간을 비교한 것이다. 그림에서 확인할 수 있는 것과 같이 TPMC의 증가폭이 크지 않아 확연히 증가하는 모습을 확인할 수 없지만 TPMC와 FBOCC 모두 서버 트랜잭션의 쓰기 비율이 증가함에 따라 평균 응답시간이 증가하는 것을 확인할 수 있다. 또한 TPMC의 평균 응답시간이 FBOCC의 평균 응답시간이 확연히 감소하는 것을 확인할 수 있는데 이는 LBCPC가 단일 채널의 브로드캐스트 주기보다 짧기 때문에 성공된 트랜잭션의 수행시간이 단축되는 것뿐만 아니라 TPMC의 재실행을 위한 대기시간이 짧은 것 또한 수행시간을 단축하는 데 큰 영향을 미치기 때문이다.

4.4 모바일 트랜잭션 데이타 개수에 따른 응답 시간

모바일 트랜잭션에서 수행해야 하는 데이타의 개수는 트랜잭션의 평균 응답시간에 영향을 미친다. 본장에서는

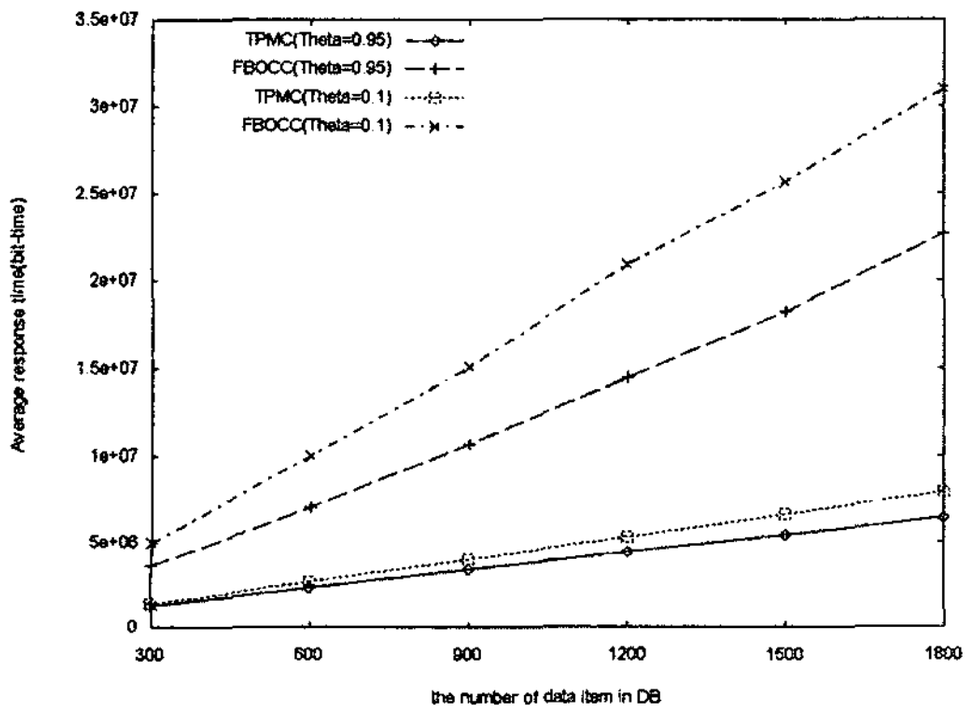

그림 12 데이타베이스의 데이타 개수에 따른 평균 응답 시간

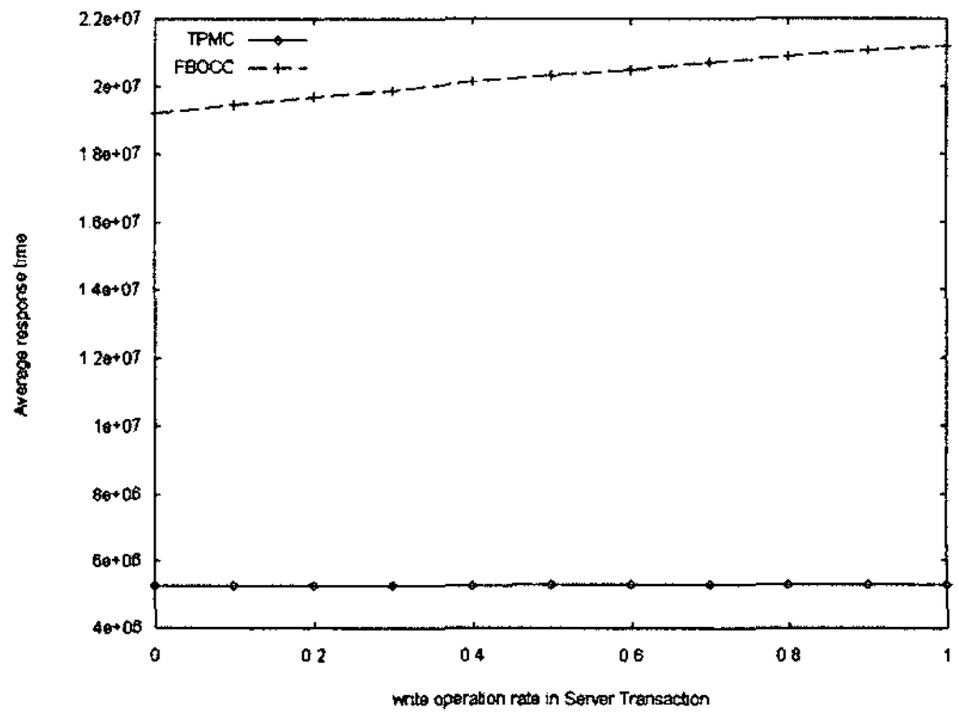

그림 13 서버 트랜잭션 갱신률에 따른 평균 응답 시간

트랜잭션의 데이타 개수의 증가에 따른 TPMC와 FBOCC의 평균 응답시간을 비교해 보도록 하겠다.

그림 14는 트랜잭션 데이타 개수에 따른 평균 응답시간을 나타낸 그림이다. 그림을 통해서 알 수 있듯이 트랜잭션의 데이타 개수가 증가할수록 TPMC의 평균 응답시간이 FBOCC의 평균 응답시간보다 좋은 것을 확인할 수 있다. 이는 짧은 LBCPC의 TPMC가 보다 긴 브로드캐스트 주기로 재실행을 실행하는 FBOCC기법보다 재실행을 위한 대기시간이 짧아지기 때문이다.

4.5 인덱스 채널과 데이타 채널의 개수에 따른 응답시간

본 장에서는 인덱스 채널과 데이타 채널의 개수에 따른 성능을 비교해 보도록 하였다. 그림 15에서는 인덱스 채널 1, 2, 3개에 따른 데이타 채널을 5, 10, 15, 20개로 증가시키면서 TPMC의 평균 응답시간을 측정한 결과이다.

그림 15를 살펴보면 우선 인덱스 채널의 개수에 상관없이 데이타 채널의 개수가 증가함에 따라 응답시간이 감소하는 것을 확인할 수 있다. 이는 데이타 채널의 개수가 증가할수록 하나의 데이타 채널에 할당되는 데이타의 개수가 적어지기 때문에 가장 긴 채널의 브로드캐스트 주기로 결정되는 LBCPC 또한 짧아지게 된다. 이 때문에 전반적으로 데이타 채널의 개수가 증가함에 따라 평균 응답시간이 줄어들게 된다.

그림 15에서의 가장 큰 특징은 데이타 전용채널의 개수가 7개 이하일 때 인덱스 전용채널의 개수는 1개로 구성된 TPMC의 성능이 가장 좋았다는 것이다. 또한 데이타 전용채널이 8개에서 12개일 때는 인덱스 전용채널이 2개로 구성된 TPMC의 성능이 좋은 것을 확인할 수 있다. 이는 [7]에서 10개의 채널을 2:8의 비율로 인덱스 전용채널과 데이타 전용채널로 나누었을 때 가장 좋은 성능을 보인 결과가 같은 것을 확인할 수 있다. 그림 15에서 데이타 전용채널이 13개 이상일 때는 인덱스 전용채널이 3개일 때 가장 빠른 평균 응답시간을 보였다.

본 장에서는 전체적으로 데이타 채널의 개수가 증가하면서 성능 지표인 응답시간이 줄어드는 것을 확인할 수 있었다. 또한 각 데이타 채널개수에 따른 응답시간을 최소로 하는 최적의 인덱스 채널 개수가 있음을 확인할 수 있었다.

5. 결 론

본 논문에서 우리는 멀티 무선 방송 채널에서 읽기 트랜잭션을 위한 동시성 제어 기법을 제안하였다. 지금까지 제안된 트랜잭션 기법들은 단일채널을 기반으로 연구 되어 왔다. 그러나 멀티 채널에서는 기존에 연구되어온 데이타 할당 스킴으로 생성된 데이타 채널들이 임의의 시점에서의 각 데이타 채널의 브로드캐스트 주기, 즉 BCPC가 서로 다르기 때문에 특정 채널에서 데이타를 수신한 후, 다음 채널로 이동하였을 때 트랜잭션이 접근하는 데이타들의 일관성이 깨질 수 있다. 이를 해결하기 위하여 데이타 전용채널에서는 LBCPC만큼 동일한 BCPC의 데이타를 반복적으로 방송하고 매 LBCPC마다 전체 채널에서의 업데이트 정보를 내보내고 클라이언트는 검증을 실시하게 되어 일관성 있는 데이타를 읽을 수 있다. 읽기 트랜잭션의 검증은 클라이언트 측에서 자체적으로 실행되기 때문에 클라이언트에서 서버로 읽기 트랜잭션을 위한 상향 대역폭은 사용되지 않는다. 또한 본 논문에서의 LBCPC가 단일채널에서의 브로드캐스트 주기보다 짧기 때문의 모바일 트랜잭션들의 최신성(Currency)을 보장해 줄 수 있다.

기존의 단일채널에서의 동시성 제어 기법들과의 비교를 통하여 여러 조건하에서 본 논문에서 제안한 기법의 성능을 분석하였다. 시뮬레이션 결과 TPMC의 평균응답시간이 확연히 감소하는 것을 확인할 수 있었다. 또한

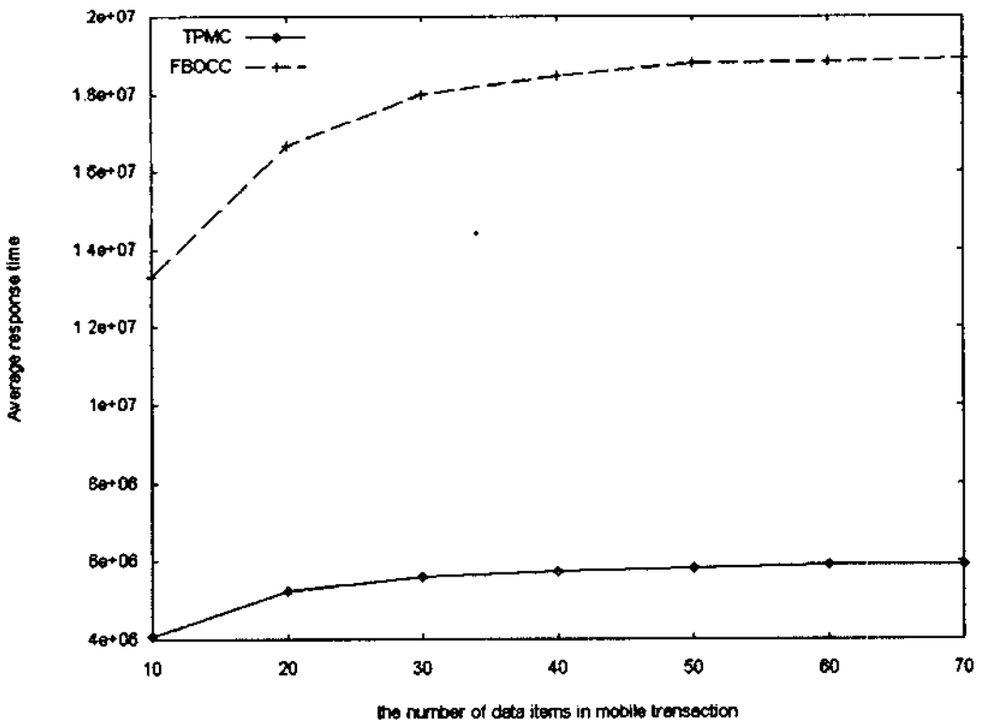

그림 14 모바일 트랜잭션의 데이타 개수에 따른 평균 응답시간

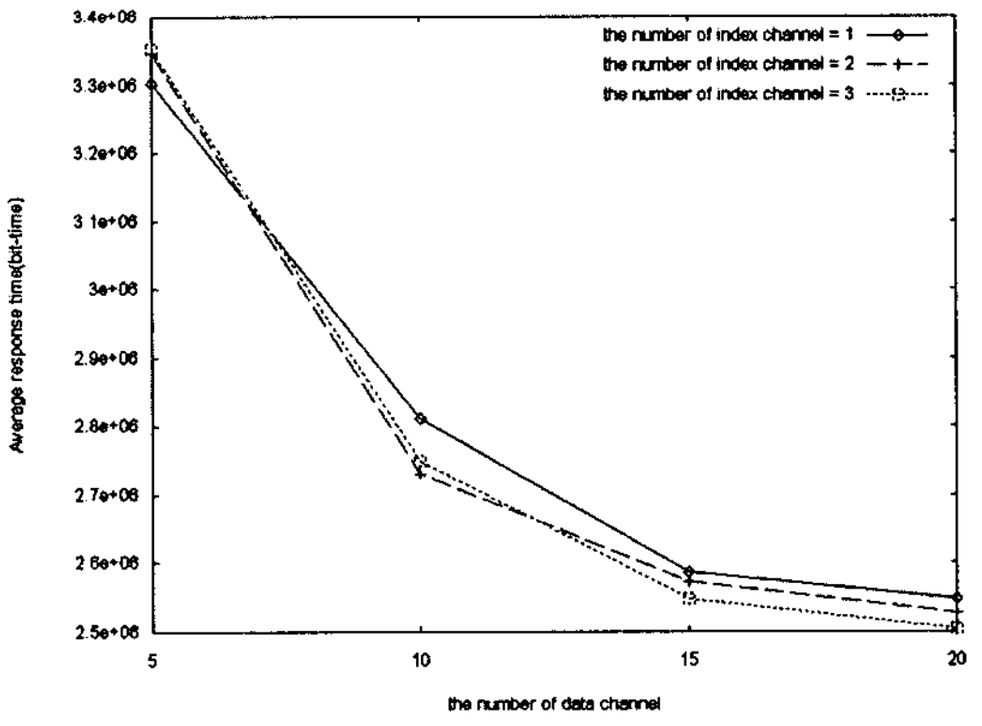

그림 15 인덱스 채널과 데이타 채널의 개수에 따른 평균 응답 시간

본 논문에서 제안한 TPMC의 좋은 성능을 위한 인덱스 전용 채널과 데이타 전용채널의 채널 비율 또한 확인할 수 있었다.

본 논문에서는 기존에 연구되지 않았던 다중 브로드캐스트 채널 환경을 기반으로 읽기 트랜잭션을 위한 동시성 제어 기법을 제시하였다. 앞으로 다중 채널을 바탕으로 일관성을 유지하는 갱신 트랜잭션에 대한 연구 또한 진행되어야 한다.

참 고 문 헌

[1] S. Acharya, M. Franklin, S. Zdonik, and R. Alonso, "Broadcast Disks: Data Management for Asymmetric Communication Environments," Proc. ACM SIGMOD International Conference on Management of Data, pp. 199-210, 1995.

[2] S. Acharya, M. Franklin, and S. Zdonik, "Balancing Push and Pull for Data Broadcast," Proc. ACM SIGMOD Int. Conf. on Management of Data, pp. 183-194, 1997.

[3] P. A. Bernstein, V. Hadzilacos, N. Goodman, "Concurrency Control and Recovery in Database System," Addison Wesley, Massachusetts, 1987.

[4] H. Cho, "Concurrency Control for Read-Only Client Transactions in Broadcast Disks," IEICE Trans. Commun., Vol.E86-B, No.10, 2003.

[5] V. Lee, K-W. Lam, T-W Kuo, "Efficient validation of mobile transactions in wireless environments," The Journal of Systems and Software, pp. 183-193, 2004.

[6] K. Prabhakara, K.Hua, and J. Oh, "Multi-Level Multi-Channel Air Cache Design for Broadcasting in a Mobile Environment," in Proceedings. 16th International Conference on Data Engineering, 2000.

[7] Sungwon Jung, Byungkyu Lee and Sakti Pramanik," A Tree-Structure Index Alocation Method with Replication over Multiple Broadcast Channels in Wireless Environments," IEEE transactions on knowledge and data engineering, Vol.17, No.3, March 2005.

[8] Jayavel Shanmugasundarm, Arvind Nirthakashyap, Rajendran Sivasankaran, "Efficient Concurrency Control for Broadcast Environments," ACM SIGMOD Volume 28, Issue 2 (June 1999) Pages: 85-96. 1999.

[9] Kung, H. T., Robinson, J. T., "On optimistic methods for concurrency control," ACM Transactions on Database Systems 6(2), pp. 213-226, 1981.

[10] W.C. Peng and M.S. Chen, "Dynamic Generation of Data Broadcasting Programs for Broadcast Disk Arrays in a Mobile Computing Environment," Proc. ACM Conf. Information and Knowledge Management (CIKM), pp. 35-45, Nov. 2000.

[11] Wai Gen Yee, IEEE, Shamkant B. Navathe, Edward Omiecinski, and Christopher Jermaine, "Efficient Data Allocation over Multiple Channels at Broadcast Servers," IEEE Transaction on computers, Vol.51, No.10, October 2002.

[12] Dimitrios Katsaros, Yannis Manolopoulos, "Broadcast program generation for Webcasting," Data & Knowledge Engineering 2003.

[13] N. Shivakumar and S. Venkatasubramanian, "Efficient Indexing for Broadcast Based Wireless Systems," Mobile Networks and Applications, pp. 433-446, 1996.

[14] Il Young Chung, Bhargava, B., Mahoui, M., Lilien, L., "Autonomous transaction processing using data dependency in mobile environments," Distributed Computing Systems, 2003.FTDCS 2003. Proceedings. The Ninth IEEE Workshop on Future Trends, 28-30, pp. 138-144, 2003.

[15] Lee, V.C.S., Kwok-Wa Lam, Son, S.H., "On transaction processing with partial validation and timestamp ordering in mobile broadcast environments," IEEE Transactions on knowledge and data engineering, 51, 10, pp. 1196-1211, 2002.

정 호 련

1999년 3월~2003년 2월 경희대학교 컴퓨터공학과 학사. 2004년 3월~2006년 2월 서강대학교 컴퓨터학과 석사. 2006년 2월~현재 LG전자 MC사업본부 선임연구원. 관심분야는 이동통신, 모바일 컴퓨팅 시스템, 모바일 데이타베이스, 모바일 트랜잭션, 모바일 에이전트

정 성 원

1988년 서강대학교 전자계산학 학사. 1990년 M.S. in Computer Science at Michigan State University. 1995년 Ph.D. in Computer Science at Michigan State University. 1997년~2000년 한국전산원 선임연구원. 2000년~현재 서강대학교 컴퓨터학과 부교수. 관심분야는 Mobile Databases, Mobile Computing Systems, Spatial Databases, Telematics, LBS, Mobile Agents

박 성 욱

2006년 2월 서강대학교 컴퓨터학과 학사. 2008년 2월 서강대학교 컴퓨터공학과 석사. 2008년 2월~현재 삼성전자 정보통신총괄 연구원. 관심분야는 Mobile Computing Systems, WCDMA, GSM, WiBro, Mobile Database