주문형 데이타 방송 시스템을 위한 인덱스 및 방송 데이타 구성

(Index and Broadcast Data Organization for On-Demand Data Broadcast Systems)

강 선 희 † 이 상 돈 ††
(Sunny Kang) (Sangdon Lee)

요 약 무선 이동 컴퓨팅 환경에서 배터리 용량은 클라이언트의 유용성을 결정하는 중요한 자원이다. 그러므로 무선 컴퓨팅 환경에서 데이타 방송 기법은 데이타를 빠르게 전달하는 것은 물론 이동 클라이언트의 배터리 소모를 감소시킬 수 있는 효율적인 데이타 전달 기법을 필요로 한다.

본 논문에서는 주문형 데이타 방송 환경을 위한 인덱스 구성과 방송 데이타 구성 방안에 대하여 제안한다. 제안 기법은 방송 서버 내에서 우선순위가 높은 데이타를 대상으로 개별 데이타의 방송 시간이 포함된 인덱스 정보를 구성하여 먼저 방송 시킴으로써 이동 중인 이동 클라이언트의 적응 시간을 줄인다. 또한 다음 인덱스 정보가 방송될 시점을 데이타 방송시 포함시킴으로써 인덱스 정보의 수신을 위해 필요한 적응 시간을 감소시킨다. 제안 기법이 효과적으로 이동 클라이언트의 적응 시간을 감소시키는 것을 실험을 통해 검증한다.

키워드 : 방송 인덱스, 주문형 데이타 방송, 적응 시간, 이동 컴퓨팅

Abstract Battery capacity of mobile clients in wireless mobile computing environments is one of the important resources that determine the availability of mobile clients. So, data broadcast techniques in mobile computing environments need not only rapid delivery of requested data but also efficient data delivery mechanisms which can reduce battery consumption of mobile clients.

This paper proposes organization of an index and broadcast data for on-demand data broadcast. It organizes an index containing broadcast time of each data using some of high-priority data in a broadcast server. By sending such index information prior to the corresponding data, the proposed approach reduces tuning time of mobile clients. It also includes when the next index information will be broadcasted in each data broadcast. This can reduce tuning time of mobile clients waiting for an index to be broadcasted. Experiments show that the proposed approach effectively reduces tuning time of mobile clients.

Key words : Broadcast Index, On-demand broadcast, Tuning time, Mobile computing

1. 서 론

무선 컴퓨팅과 이동 컴퓨팅의 진보는 새로운 정보서비스의 변화를 가져왔다. 휴대용 컴퓨터 기술의 발전과 무선통신의 진보에 힘입어 시간과 장소에 제한 받지 않고 컴퓨팅 서비스를 받을 수 있는 이동 컴퓨팅에 대한 연구가 활발히 진행되고 있다. 배터리 전력을 가진 휴대용 컴퓨터나 휴대용 단말은 무선통신 채널을 통하여 다양한 응용 서비스를 받고 있다. 그러한 응용 서비스로는 모바일 뉴스, 모바일 경매, 모바일 경품, 모바일 주식, 모바일 게임, 날씨 정보, 교통정보, 여행 정보 등을 포함한다. 이러한 응용 서비스의 규모와 사용자의 수가 계속적으로 증가하고 있는 추세이다.

무선 통신환경에서 물리적인 제약은 서버와 클라이언트의 자원에 있어 비대칭으로부터 온다. 무선통신 채널에서 클라이언트의 비대칭은 CPU, 메모리, 배터리, 통신대역폭 등과 같은 유용한 자원을 포함하여 매우 제한된 자원을 갖는다는 것이다. 특히나 배터리 용량은 클라

· This paper was supported in part by research funds of mokpo national university
† 정 회 원 : 목포과학대학 컴퓨터정보과 교수
isun32@hanmail.net
†† 종신회원 : 목포대학교 정보공학부 교수
sdlee@mokpo.ac.kr
논문접수 : 2005년 3월 10일
심사완료 : 2005년 5월 26일

이언트의 유용성을 결정하는 중요한 자원 중의 하나이다. 이러한 자원의 비대칭으로부터 서버의 높은 자원을 공유할 수 있는 방법은 데이타 방송 기법이다. 데이타 방송은 정보를 가지고 있는 서버가 이동 클라이언트에게 데이타를 방송하면 클라이언트가 자신이 원하는 데이타가 방송채널에 나타날 때 이를 검색하는 방법이다. 이러한 이동환경에서 데이타 방송은 이동 클라이언트의 전력소모를 감소할 수 있는 매력적인 데이타 전달기법이다[1].

데이타 방송의 효율성을 측정하는 기준은 접근 시간(access time)과 적응 시간(tuning time)이다. 접근 시간은 클라이언트가 질의를 요청한 순간에서부터 모든 요청한 데이타 항목을 얻을 때까지 소요되는 시간을 말한다. 적응 시간은 클라이언트가 요청한 데이타를 얻을 때까지 실제로 방송채널을 수신하면서 소모하는 시간을 말한다. 효율적인 데이타 전달을 위해서는 이러한 두 가지 기준을 최소화시키는 것이 요구된다[2-5]. 이동 클라이언트는 관심 있는 데이타 항목을 검색하기 위하여 활동 모드(active mode)를 유지하는데 활동 모드에서 소모되는 전력 소비량은 수면 모드(doze mode)에서 전력 소모의 수천배에 이른다[4]. 그러므로 클라이언트의 전력소모를 감소시키기 위해서는 적응 시간을 줄이는 것이 요구된다.

적응 시간을 줄이는 일반적인 접근 방법은 클라이언트가 요청한 데이타를 전달 받는 시점을 미리 알 수 있도록 사전에 알려주는 인덱스 기법을 적용하는 것이다. 인덱스 기법의 기본 전제는 요청한 데이타가 언제, 어떤 데이타가 방송될 것인지 데이타가 실제 방송되어지기 전에 알 수 있어야 한다는 것이다.

데이타 방송은 크게 주기적(Periodic) 데이타 방송과 주문형(On-Demand) 데이타 방송으로 구분된다. 주기적 데이타 방송[6]에서는 데이타 항목의 방송 시간이 방송 전에 미리 결정되어 정해진 순서에 따라 방송 대상 데이타 집합이 반복적으로 방송된다. 그러므로 주기적 방송의 경우 인덱스 정보를 얻는 것이 손쉬운 일이다. 반면에 주문형 데이타 방송[7,8]은 방송의 전체 순서가 미리 결정되어지는 것이 아니라 클라이언트의 요청에 따라 방송 중에 동적으로 결정된다. 클라이언트는 상향 채널을 통하여 데이타를 요청하고 서버는 클라이언트에게 요청한 데이타를 방송을 통해 보냄으로써 응답한다. 동일한 데이타에 대한 여러 클라이언트의 데이타 요청은 방송서버가 해당 데이타를 방송함에 따라 동시에 모든 해당 클라이언트들에서 수신된다. 주문형 데이타 방송은 방송 중에 결정되어지기 때문에 언제 어떤 데이타가 방송될 것인지를 결정하는 인덱스 정보를 정의하는 것이 쉽지 않다.

이러한 이유로 기존의 인덱싱 기법[4-6, 9-11]은 주기적 방송 환경에 주로 초점을 맞추어 왔다. 그러나 클라이언트의 가변적인 접근 요구에 따라 동적 방송 스케줄이 결정되어야 하는 주문형 방송 환경을 위한 인덱스 기법에 대한 연구는 극히 드물다. 한편 주기적 데이타 방송 환경에서 제안되었던 인덱스 기법들을 주문형 데이타 방송 환경에 직접 적용하는 것은 가능하지 않다[12].

주문형 데이타 방송 환경을 위하여 최초로 제안된 인덱스 접근 방법[12]은 방송 서버에 데이타의 요구가 쌓이게 되고, 이 데이타 요구들을 기반으로 방송 순서가 결정된다는 점에 착안하여 각 개별 데이타의 방송 시간을 예측하는 대신에, 일정 시간 동안에 방송이 될 가능성이 큰 데이타 집합을 선정하고 이를 클라이언트에게 알려주는 인덱스 힌트 기법이다. 이 접근 방법에서는 인덱스 정보에 개별 데이타 항목에 대한 방송 시간 정보를 포함하지 않기 때문에 클라이언트는 요구한 데이타가 일정 시간 구간 동안에 방송될지 여부만을 알 수 있고, 이로 인해 해당 구간동안 계속해서 방송 채널을 감시해야만 하였다. 본 논문에서는 인덱스 정보에 개별 데이타의 방송 시간 정보까지도 포함하도록 함으로써 클라이언트가 보다 세밀하게 방송 채널의 감시 여부를 판단할 수 있도록 하여 낭비되는 적응 시간을 보다 줄이도록 한다.

2장에서는 연구 배경을 살펴보고 3장에서는 방송을 위한 인덱스 구성을 설명하고 4장에서는 방송 데이타 구성 기법에 대하여 기술한다. 5장에서는 제안 기법의 성능을 평가하기 위한 실험 결과를 분석하고, 마지막으로 6장에서 결론을 맺는다.

2. 연구 배경 및 필요성

2.1 시스템 환경

논문에서 가정하는 시스템 환경은 그림 1과 같이 방송 서버와 많은 수의 클라이언트가 위성통신 환경을 포함한 무선 네트워크를 기반으로 데이타 방송이 이루어지는 환경을 가정하고 있다. 클라이언트는 데이타 요청을 보낼 때 상향 채널을 사용하고 서버로부터 데이타를 수신할 때 하향 채널을 청취한다. 클라이언트의 요청은 서버의 큐에 저장되며 서버에서는 방송 스케줄링 알고리즘을 사용하여 다음 방송될 데이타 항목을 동적으로 결정한다. 클라이언트는 데이타 요청을 보낸 후 요청한 데이타를 수신할 때까지 하향 채널을 감시한다. 하향 채널을 통해 데이타 항목이 방송될 때 데이타 항목을 요청한 모든 활동 중인 클라이언트는 동시에 데이타를 접근한다.

방송 서버는 큐 내에 있는 데이타 중에서 다음에 방

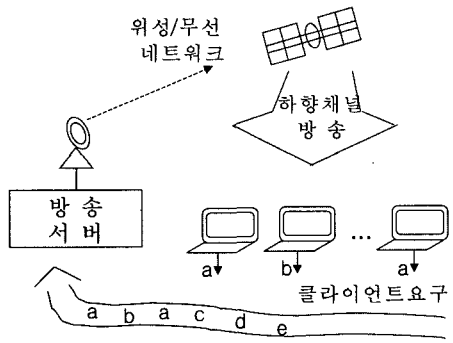

그림 1 주문형 방송 환경의 예

송할 데이타를 위한 인덱스를 생성하고 이를 방송한 후 해당 데이타들을 방송한다. 클라이언트는 요청한 데이타나 인덱스를 수신할 때까지 하향 채널을 감시하며, 클라이언트가 인덱스를 받았을 때 인덱스 내에 요청한 데이타가 포함되어 있는지를 확인한다. 만일 거기에 없다면, 클라이언트는 다음 인덱스를 받을 때까지 수면 모드로 들어간다. 만일 인덱스 내에 클라이언트가 요청한 데이타가 있으면, 클라이언트는 인덱스 정보 중 해당 데이타가 방송될 시점까지 수면 모드에 있다가 방송 시점에 활동 모드로 들어가 해당 데이타를 수신한다.

2.2 관련 연구 및 접근 방법

데이타 방송은 이동통신 환경에서 통신 대역폭의 비대칭성을 이용하여 높은 대역폭 하향 채널을 통해 데이타를 방송하고, 클라이언트는 원하는 데이타가 방송채널에 나타날 때 이를 접근함으로써 클라이언트의 수에 무관하게 효과적으로 데이타를 전달할 수 있는 확장성이 큰 데이타 전달 방법이다[1].

데이타 방송은 크게 두 종류로 구분이 가능하다. 주기적 데이타 방송은 방송 대상 데이타 항목의 순서와 방송의 빈도 등 클라이언트의 접근 요구가 방송 전에 미리 결정되어 전체적인 방송 스케줄이 결정되는 정적인 방송 스케줄을 사용한다[2]. 반면에 주문형 데이타 방송은 방송의 전체 순서가 미리 결정되어지는 것이 아니라 클라이언트의 동적인 데이타 요청에 따라 방송 중에 결정되는 동적 방송 스케줄을 갖는다[7,8,13].

주기적 방송 환경에서는 데이타가 방송되기 이전에 이미 데이타의 방송 시점이 결정된다. 그러므로 이를 사용하여 전체 방송 스케줄에 대한 인덱스를 구성하고, 실제 데이타의 방송에 앞서 인덱스를 먼저 방송하는 방식으로 클라이언트의 적응 시간을 크게 줄일 수 있다. 주기적 데이타 방송을 위해 효율적인 인덱스의 구조들에 대한 연구가 이루어져 왔으며, 해싱, 트리 구조 또는 시그너처(signature)를 사용한 기법들이 제안된 바 있다[4-6,9,11].

주문형 방송 환경에서 인덱스의 적용을 위해서는 클라이언트가 요청한 데이타가 실제 방송되기 전에 방송 시점을 미리 정확히 아는 것이 필요하다. 그러나 주문형 데이타 방송은 방송 중에 방송 스케줄이 동적으로 결정되어지기 때문에 전체 방송 스케줄에 대해서 데이타가 언제 방송되어질 것인지를 정확히 정의하는 것이 쉽지 않다. 그러므로 주기적 방송 환경을 위해 제안되었던 인덱스 기법들이 그대로 주문형 방송 환경에 적용될 수 없다.

주문형 데이타 방송 환경을 위해 최초로 제안된 인덱스 접근 방법[12]은 일정 시간 이후에 방송될 것이 기대되는 데이타들을 예측하고 이동 클라이언트에게 알려주는 인덱스 힌트 기법이다. 그림 2는 [12]에서 제안한 인덱스 힌트를 포함한 방송 스케줄의 예이다. 방송 데이타 항목 중 키에 해당하는 부분은 밑줄로서 나타내었다. 키는 이동 클라이언트가 요청한 데이타를 구별짓는 식별자이다. 인덱스 힌트는 이후 3이라는 시간 동안에 KT, MS, IBM이라는 키 값을 갖는 3개의 데이타가 방송될 것이라는 것을 나타낸다. 데이타 항목1은 KT를 키로 갖는다. 데이타 항목2은 MS를 키로 갖는다. 데이타 항목3은 IBM을 키로 갖는다. 여기서 각 데이타 항목을 방송하는데 1만큼의 시간이 소요됨을 가정하고 있다. 키 값 KT, MS, IBM 이외의 데이타를 요구한 이동 클라이언트는 인덱스 힌트를 참조하여 이후 3의 시간 동안 자신이 필요로 하는 데이타가 방송되지 않을 것이므로 수면 모드에 들어감으로써 적응 시간을 줄이는 방법을 사용한다.

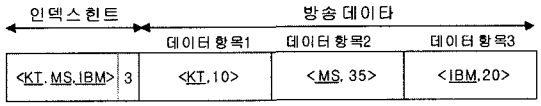

그림 2 인덱스 힌트의 구성 예

이 접근 방법에서는 인덱스에 개별 데이타 항목에 대한 키 정보만을 포함하고 있다. 방송 시간 정보를 포함하지 않기 때문에 클라이언트는 요구한 데이타가 일정 시간 구간동안에 방송될지 여부만을 알 수 있고, 이로 인해 데이타가 방송되는 구간동안 계속해서 방송 채널을 감시해야만 한다.

만일 방송 인덱스를 구성할 때 각 데이타가 언제 방송될 것인지 방송 시간 정보를 제시해 준다면 클라이언트가 보다 세밀하게 방송 채널의 감시 여부를 판단할 수 있도록 하여 낭비되는 적응시간을 줄일 수 있을 것이다. 그림 3은 그림 2에 대해서 각 항목별로 방송 시간 정보를 포함하도록 한 인덱스 구성의 예이다. 클라이언

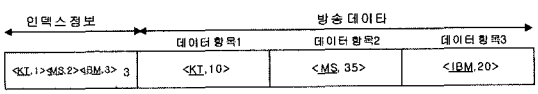

그림 3 방송 시간 정보를 포함하는 인덱스 구성의 예

트는 인덱스 정보를 참조하여 데이타 항목의 정확한 방송 시점을 인지하고 그 시간 전에 깨어나서 해당 데이타를 수신함으로써 보다 더 적응 시간을 줄이게 될 것이다.

이러한 인덱스를 효과적으로 구성하기 위해서는 우선 일정 시간 구간동안에 방송될 데이타의 집합과 방송 예정시간을 식별하는 것이 요구된다. 방송될 데이타 집합의 예측은 [12]에서 제시된 바와 같이 현재 방송 서버 내에 대기 중인 데이타 요청들을 관리하는 서버 내 우선순위 큐의 현재 상태를 고려하여 가장 우선순위가 높은 N개의 데이타 항목을 선정하는 방법과, 각 데이타 항목의 접근 패턴이 알려진 경우 서버 큐의 상태를 예측하여 일정 시간 후 가장 우선순위가 높게 될 것으로 예측되는 N개의 데이타 항목을 선정하는 방법을 사용한다. 방송 예정시간의 예측은 위와 같이 선정된 데이타 항목의 길이가 알려져 있으므로 각 데이타 항목을 방송하는데 소요되는 시간으로부터 결정이 가능하다. 한편 방송 데이타를 구성하는데 있어 데이타가 방송되는 부분에 다음 인덱스가 방송될 시점(NIP : Next Index Pointer)을 포함시키면 인덱스 정보가 방송된 이후에 데이타를 요구한 클라이언트가 다음 인덱스를 만날 때까지 계속적으로 방송 채널을 감시할 필요가 없다.

본 논문에서는 인덱스 정보 내에서 개별 데이타 항목의 방송 시간 정보를 포함하고 방송데이타의 구성 시에 NIP의 추가 여부가 인덱스의 구성 효과에 미치는 영향을 분석한다.

3. 주문형 방송을 위한 인덱스의 생성

3.1 인덱스 집합 예측

인덱스 집합 예측의 목표는 일정 시간 안에 방송되어질 데이타 항목의 집합을 예측하는 것이다. 예측에 사용된 기법으로는 큐 상태에 기반을 둔 예측 방법(Queue-based Estimation:EQ)과 데이타 요구율에 기반을 둔 예측 방법(Popularity-based Estimation:EP)이 있다 [12]. 주문형 방송 환경을 위해 제안된 여러 스케줄링 알고리즘 중에서 모든 데이타 요구의 대기 시간의 합이 가장 큰 데이타 항목을 방송하는 LWF(Longest Wait Time) 스케줄링 정책[7,13]이 평균 접근 시간의 관점에서 성능이 우수하고 개념적으로 간단하므로 본 논문에서는 LWF 스케줄링 알고리즘을 사용한 인덱스 생성에 대하여 기술한다. 본 논문에서 제안하고 있는 인덱스 구성 방법은 주문형 방송을 위해 제안된 다른 스케줄링 알고리즘[8,13]에도 쉽게 적용이 가능하다.

3.1.1 방송 서버의 큐 상태에 기반을 둔 방송 데이타의 예측(EQ)

방송할 데이타 항목은 현재 서버 큐 상태만으로 예측한다. 이러한 접근법은 서버의 현재 우선순위를 기반으로 상위 N개 데이타 항목을 선택함으로써 크기 N인 인덱스를 결정한다. 이러한 접근법은 매우 간단하고 단지 서버 큐를 조사하는 것으로서 예측을 위한 약간의 오버헤드를 야기한다.

3.1.2 데이타 요구율에 기반을 둔 방송 데이타의 예측(EP)

일정 시간 간격 동안에 현재 큐 안에 있는 데이타 항목 사이의 상대적인 순위의 변화 가능성을 고려하여 해당 시간 동안 각 데이타의 요구율에 따라 추가적으로 데이타 요청이 이루어졌을 때 기대되는 서버 큐의 우선순위 변경에 따라 상위 N개의 데이타 항목을 선택한다.

3.2 인덱스 구성

데이타의 방송은 버켓 단위로 이루어진다고 가정한다. 버켓은 방송을 위한 최소의 논리적 단위로서 방송 채널에 보내지는 단위 정보를 의미한다. 버켓은 방송 데이타를 포함하고 있는 데이타 버켓과 인덱스 정보를 포함하고 있는 인덱스 버켓으로 구분한다. 각 버켓은 독립적으로 식별이 가능하도록 다음의 헤더 정보를 갖는다.

- 버켓 식별자 : 방송 시작으로부터 버켓의 상대위치(offset)
- 버켓 유형 : 데이타 버켓 또는 인덱스 버켓의 구분

인덱스 버켓은 [인덱스 집합, 인덱스 구간]으로 구성된다. 인덱스 집합은 <키, 시간>의 쌍으로 구성된다. 인덱스 집합에서 키는 데이타 항목의 키 부분을 의미한다. 인덱스 집합에서 시간은 키를 포함하고 있는 데이타 항목이 방송되는 시점을 나타낸다. 인덱스 구간은 인덱스 집합에 있는 모든 데이타 항목이 방송되는 동안의 시간 간격이다. 모든 시간은 단위 방송시간(Broadcast tick)을 사용하는 것으로 가정한다.

데이타 버켓은 키 부분과 데이타 부분으로 이루어진다. 데이타 버켓의 마지막에는 다음 인덱스 버켓의 위치를 지시하는 포인터(Next Index Pointer : NIP)를 가질 수 있다. NIP는 다음 인덱스의 방송 시점을 나타내며, 인덱스 집합에서의 '시간'과 동일하게 표현된다. NIP는 클라이언트가 데이타를 요청하고 방송 채널을 감시하기 시작하는 시점에 어떻게 반응할 것인지를 결정하는 역할을 한다. 만일 클라이언트가 처음 수신한 버켓이 인덱스 버켓이었다면 당연히 클라이언트는 인덱스 버켓을 보고 해당 데이타의 수신 가능성을 확인할 수 있다. 그러나 처음 수신한 버켓이 데이타 버켓인 경우에는 NIP

의 여부에 따라 다음 두 가지로 반응한다. 만일 NIP가 없는 경우(이하 NIPX라 함) 클라이언트는 다음 인덱스 버켓을 만날때까지 계속해서 방송 채널을 감시해야 한다. 반면에 NIP가 있는 경우에는(이하 NIPO라 함) 다음 인덱스 버켓의 방송시점을 알 수 있으므로 다음 인덱스 버켓의 방송시점까지 클라이언트가 수면 모드로 들어가게 된다.

그림 4는 인덱스 구성의 예를 보여준다. 여기서 K_i는 데이타 d_i의 키이다. 또한 δ는 $t_4 - t_0$이다. 이 예에서 이동 클라이언트가 t_0 이전에 d_2를 요청한 경우에 활동 모드에서 방송 채널을 감시하다가 인덱스 버켓 I_0를 참조한 후 수면 모드에 들어갔다가 t_2 이전에 다시 깨어나서 d_2의 방송여부를 다시 확인하게 된다. 그러므로 이 경우 클라이언트의 적응 시간이 감소하게 되어 클라이언트의 전력 소모를 줄이게 된다. 만일 클라이언트가 t_2 시점에 d_4를 요구할 경우 NIP가 존재하면 다음 인덱스 정보(I_1)가 방송되는 t_4까지 수면 모드에 들어간다.

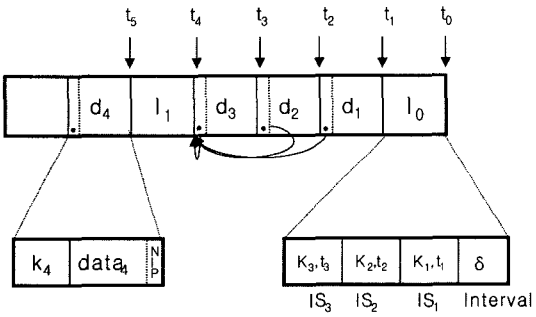

그림 4 인덱스 구성의 예

4. 방송 데이타 구성 및 관리

제안된 인덱스 구조를 사용하여 데이타를 방송하기 위해 방송 데이타를 구성하는 방법과 그리고 방송 서버에서 이동 클라이언트들의 데이타 요구를 서버 큐에 효과적으로 관리하는 방안에 대하여 기술한다.

4.1 방송 데이타 구성

서버는 동적으로 인덱스를 생성하고 방송 채널을 통하여 클라이언트에게 인덱스와 데이타를 보낸다. 즉, 인덱스 집합(IS:Index Set)과 데이타 집합(BS:Broadcast Set)를 보낸다. IS는 예측 기법에 따라 N개 데이타 항목을 선택함으로써 결정된다. IS는 데이타를 구분하는 키 정보와 그 키를 가진 데이타가 방송되어질 시간 정보를 포함한다. BS에 포함된 데이타가 전부 방송되는 시간은 인덱스에 포함된 데이타 항목의 개수, 즉 인덱스의 크기와 단위 방송시간의 곱이다.

그림 5는 클라이언트 C가 데이타 항목을 요청하고 방송 채널을 수신하는 두 시점을 나타낸다. 여기서 IS와 BS는 각각 인덱스 집합과 데이타 집합을 의미하며, IS와 BS에서 흑색 부분은 IS에 포함된 키에 해당하는 데이타가 BS에 포함되어있는 상황을 나타낸다. 한 클라이언트 C는 실선으로 표현된 시간 구간동안 방송채널을 감시하여 활동 모드에 있음을 나타낸다. 그리고 점선으로 표시된 구간동안 수면 모드에 들어감을 의미한다. 이 예에서 (a)의 경우(Index hit)는 클라이언트가 방송 채널 수신을 개시한 시점에 필요한 인덱스 정보를 수신한 경우이다. (b)의 경우(Index miss)는 클라이언트가 방송 채널 수신을 개시한 시점에 이미 인덱스 정보가 방송되어버린 경우를 의미한다.

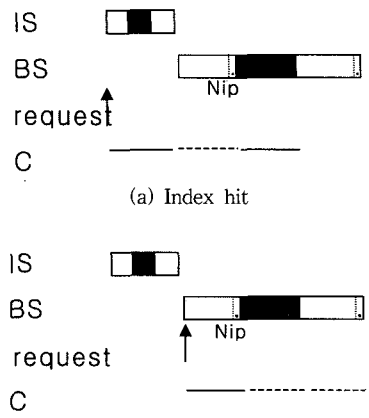

그림 5 방송 데이타 구성의 예

그림 6은 데이타 항목 a를 요청한 클라이언트 C1, C2, C3가 NIP의 존재 여부에 따라 IS와 BS에 대해서 어떻게 동작하는지를 보여주는 예이다. T_0 시점에 방송 수신을 개시한 클라이언트 C1은 인덱스 정보를 수신한 Index hit 경우로서 인덱스 정보내에서 방송시간 정보를 인지한 후 수면 모드로 들어간다. 이후 그 방송 시점에 깨어나서 데이타를 수신한다. T_1 시점에 방송채널을 개시한 클라이언트 C2, C3는 Index miss가 발생한 경우

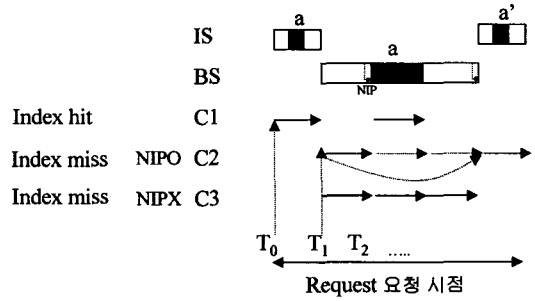

그림 6 NIP의 여부에 따른 방송 데이타 수신

로서 데이타 집합(BS)이 모두 방송된 후 다음 인덱스가 방송되는 시점에서야 다시 인덱스 정보를 수신할 수 있다. 다음 인덱스 정보를 나타내는 포인터인 NIP가 없는 경우(즉 NIPX의 경우)에는 언제 다음 인덱스 정보를 수신할 지를 미리 알 수 없으므로 C3와 같이 다음 인덱스 정보를 수신할 때까지 계속해서 활동 모드를 유지하여 방송채널을 감시해야만 한다. BS의 크기는 IS의 크기에 비교하여 일반적으로 매우 클 것이므로 이로 인한 전력소모가 커질 수도 있다. 만일 NIP 정보를 가진 경우(즉 NIPO의 경우) 클라이언트는 다음 인덱스 방송 시점을 알 수 있으므로 NIP를 참조하여 C2에서와 같이 수면 모드에 들어갔다가 다음 인덱스 정보가 방송될 때 깨어나서 방송 채널을 다시 감시한다.

4.2 서버 큐 관리

방송 서버에서 클라이언트의 데이타 요청은 큐를 사용하여 관리된다. 본 논문에서 가정하는 바와 같이 방송 스케줄링 알고리즘으로 LWF를 사용하는 경우 각 데이타 항목을 위한 큐 엔트리는 해당 항목을 요청한 시점 정보를 포함한다. 예를 들어 KT 데이타를 요청한 클라이언트 C1, C2, C3의 데이타 요청 시점이 각각 10, 20, 25라 하면 KT를 위한 큐 엔트리는 다음과 같이 구성된다.

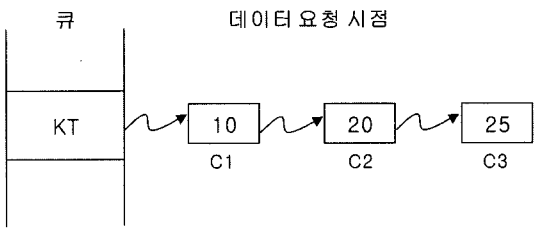

그림 7 큐 엔트리 구성 예

이 경우 현재 시점이 40이라 하면 KT를 요청한 클라이언트의 총 대기 시간은 (40-10) + (40-20) + (40-25) = 65가 되어 큐 내에서 상대적인 우선 순위가 결정된다. 만일 KT가 방송되면 KT를 요청하였던 모든 클라이언트들 C1, C2, C3는 동시에 데이타를 수신하게 될 것이다. 이후 큐에서 KT를 위한 엔트리는 제거된다. 그러나 이때 만일 C3가 수면 모드에 있다면 KT가 방송되더라도 C3는 KT 데이타를 받을 수 없다. 이 경우 C3는 다른 클라이언트가 또다시 KT 데이타를 요청하여 우선순위에 따라 다시 방송되는 경우에만 데이타의 수신이 가능하다.

이러한 상황은 한 데이타 항목이 방송될 때 해당 데이타 항목을 요청한 클라이언트가 수면 모드에 있는 경우에 발생한다. 즉 그림 6에서 C2가 데이타 a를 요청하는 경우 데이타 a가 방송되는 시점에서 C2가 수면 모드에 있게 된다. 한편 a가 방송되면 서버 큐에서 a의 큐 엔트리를 제거하므로 결국 C2의 데이타 요구 정보가 상실되는 경우가 발생한다. 그러나 C3의 경우 a가 방송되는 시점에 C3가 활동 모드에 있으므로 C3는 성공적으로 a를 수신할 수 있다. 그러므로 위에서 언급한 사항은 NIP를 사용하는 경우에만 발생할 수 있음을 알 수 있다.

이의 해결 방법으로 서버에서는 새로운 데이타 요구를 BQ(Backup Queue)에 별도로 관리한다. 이후 4.4절에서 기술하듯이 나중에 새로운 인덱스를 결정할 때 기존 큐와 BQ는 통합되어 올바른 방송 우선순위를 결정할 수 있도록 한다.

4.3 클라이언트의 방송 데이타 수신

클라이언트는 상향 채널을 통하여 서버에 접근 요구를 보낸다. 서버에서는 인덱스 정보 또는 데이타 정보를 방송한다. 클라이언트는 방송 채널을 감시한다. 클라이언트가 인덱스 버켓을 수신하면 인덱스 내에 클라이언트가 요청한 데이타가 포함되어 있는지 확인한다. 인덱스에 포함되어 있다면 해당 데이타 버켓을 수신할 때까지 방송 채널을 계속적으로 감시한다. 만일 인덱스 내에 없으면 클라이언트는 다음 인덱스 버켓이 방송될 때까지 수면 모드에 들어간다. 반면에 클라이언트가 데이타 버켓을 수신하면 방송 데이타 구성 방법에 따라 다음의 두 가지로 동작한다. NIPO의 경우 데이타 버켓 내에 다음 인덱스 버켓의 방송 위치를 가지고 있으므로 클라이언트는 다음 인덱스 버켓이 방송될 때까지 수면 모드에 들어간다. 그러나 NIPX의 경우에는 어느 시점에 다음 인덱스 정보가 방송될 지를 알 수 없으므로 계속적으로 방송 채널을 감시해야만 한다.

4.4 방송 스케줄링

4.4.1 NIPX를 위한 스케줄링

현재 시점을 clock, i번째 인덱스를 생성한 시점을 $clock_i$로 나타낸다. 또한 i번째 인덱스의 인덱스 구간을 II_i로 나타낸다. i번째 인덱스를 생성할 시점은 이전 인덱스 시점($clock_{i-1}$)과 i-1번째 인덱스의 인덱스 구간의 합이 된다. 즉 clock = $clock_{i-1}$ + II_{i-1} 이다. 인덱스를 생성해야 하는 시점에서 방송 서버는 i번째 인덱스 I_i를 생성하고 그것을 방송한다. 만일 인덱스 방송 시점이 아니면(즉 데이타의 방송이 필요함) 방송 서버내 데이타 집합으로부터 우선 순위가 가장 높은 데이타 항목을 방송한다. 데이타 항목을 방송한 후 서버 큐에서 그 항목을 제거한다. 다음은 데이타의 방송을 위한 기본적인 NIPX 스케줄링 알고리즘이다.

<Algorithm 1 NIPX Scheduling>

Input : an estimation policy(ε), the scheduling policy(S), the index size(N), the queue for pending requests(Q), the previous index broadcast time($clock_{i-1}$) and the previous index interval(II_{i-1})

Output : what to broadcast next(*next_broadcast*)
Procedure :
while Q is not empty do
  if *clock* = $clock_{i-1}$ + Π_{i-1} then // time to broadcast an index
    Estimate an Index Set IS_i *of Index size N* from Q using ε
    Generation an index I_i
    next_broadcast := I_i
  else // time to broadcast a data item
    Select a data item, d_i, with the highest priority from IS_i using S
    next_broadcast := d_i
    Remove a queue entry for d_i from Q
  end if
end while
return *next_broadcast*

4.4.2 NIPO를 위한 스케줄링

NIPO의 경우 Index miss가 발생할 때 클라이언트가 다음 인덱스 정보를 수신할 때까지 수면 모드에 있게 된다. 만일 이 동안에 그 클라이언트가 요청했던 데이타가 실제로 방송이 되어버리면 위의 스케줄 정책에 따라 해당 데이타가 서버 큐에서 제거되어버린다. 그렇게되면 수면 모드에 있던 클라이언트는 다시 깨어나더라도 해당 데이타를 수신할 수 없게 되거나 또는 다른 클라이언트들이 그 데이타에 대해서 다시 요청을 하는 시점까지 수신이 지연되는 결과를 맞게 된다. 그러므로 NIPO의 경우에는 이러한 클라이언트에 대한 새로운 요청을 백업 큐에 별도로 저장한다. 이후 새로운 인덱스가 생성되기 전에 백업 큐에 있는 데이타 요청을 현재 큐에 병합시킴으로써 데이타 요구의 손실을 예방한다. 다음은 NIPO 스케줄링이다.

<Algorithm 2 NIPO Scheduling>
Input : an estimation policy(ε), the scheduling policy(S), the index size(N), the queue for pending requests(Q), a back-up queue(BQ), the previous index broadcast time($clock_{i-1}$) and the previous index interval(Π_{i-1})
Output : what to broadcast next(next_broadcast)
Procedure :
while Q is not empty do
  if *clock* = $clock_{i-1}$ + Π_{i-1} then // time to broadcast an index
    Merge pending requests in BQ to Q
    Estimate a Index Set IS_i of Index size N *from* Q using ε
    Generation an index I_i
    next_broadcast := I_i
    Move pending requests for all data items which are not in IS_i from Q to BQ
  else // time to broadcast a data item
    Select a data item, d_i, with the highest priority from IS_i using S
    next_broadcast := d_i
    Remove a queue entry for d_i from Q
  end if
end while
return *next_broadcast*

5. 실험 및 결과

5.1 실험 환경

제안 기법의 성능 평가를 위하여 CSIM++을 사용하여 실험하였다. 실험 모델은 1개의 방송 서버와 많은 클라이언트로 구성된다. 클라이언트는 두 부류로 구분되며 제안 기법의 성능을 관찰하기 위한 하나의 측정 클라이언트(MC :Measured Client)와 MC 이외의 여러 클라이언트를 대표하는 가상 클라이언트(VC : Virtual Client)로 구성된다. MC와 VC는 모두 방송 서버(Broadcast Server)에 데이타 접근을 요구한다. 각 클라이언트는 서버에 페이지를 요청하며 각 요청의 간격은 지수 분포[14]를 따른다고 가정하였다. 요청되는 데이타가 편중되는 경우를 위하여 각 페이지에 대한 클라이언트들의 요청은 Zipf 분포[15]를 따르도록 하였다. 클라이언트들은 한 데이타 항목을 요청한 후 방송채널을 감시한다. 방송 서버는 데이타를 우선 순위 큐에 보관하였다가 스케줄링 우선 순위 기준에 따라 해당 데이타를 방송한다.

표 1은 실험에서 사용된 인자들을 기술한다.

IndexSize가 클수록 한 인덱스 구간에 많은 데이타

표 1 실험 인자

기호	설명	기본값	범위
DbSize	데이타베이스 크기	100	100
IndexSize	한 인덱스 내에 포함되는 데이타 항목의 개수	9	2-30
KeyToDataRatio	데이타 항목의 키와 데이타의 크기 비율	0.1	-
VCSkew	가상 클라이언트의 접근 편중도	0.9	0.0-1.5
MCSkew	측정 클라이언트의 접근 편중도	0.9	0.0-1.5
VCMeanReqTime	가상 클라이언트에서 데이타 접근 요구 사이의 평균 시간	0.05	0.005-0.1
MCMeanReqTime	측정 클라이언트에서 데이타 접근 요구 사이의 평균 시간	5.0	-
DataToBucketRatio	데이타 항목과 버켓의 크기 비율	1.0	-

항목의 키가 인덱스에 포함된다. KeyToDataRatio가 크면 데이타 항목에 비해 키의 크기가 작다. 그러므로 인덱스의 전체적인 크기가 감소하는 효과가 있다. VCSkew와 MCSkew는 VC와 MC에서 데이타 접근 편중도를 나타낸다. 한편 VCMeanReqTime와 MCMeanReqTime은 작을수록 단위 시간 내에 생성되는 데이타 접근 요구 수가 증가하여 시스템의 작업 부하가 커지게 된다. 이들의 상대적인 비율은 가상 클라이언트와 측정 클라이언트의 작업 부하에 미치는 비율을 나타낸다. 즉 VCMeanReqTime/MCMeanReqTime = 100인 경우, 측정 클라이언트가 전체 작업 부하에 미치는 비율이 1%임을 의미한다. 데이타베이스 내 총 데이타 항목 수는 100으로 고정하였으며 실험에서 VCMeanReqTime는 0.005-0.1, MCMeanReqTime는 5.0을 사용하였다.

5.2 성능 기준

데이타 방송의 효율성 측정을 위해 사용되는 기준은 접근 시간(access time)과 적응 시간(tuning time)이다. 접근 시간은 클라이언트가 질의를 요청한 순간에서부터 모든 요청한 데이타 항목을 얻을 때까지 소요되는 시간을 말한다. 적응 시간은 클라이언트가 요청한 데이타를 얻을 때까지 실제로 방송채널을 수신하면서 소모하는 시간을 말한다. 효율적인 데이타 전달을 위해서는 이러한 두 가지 기준을 최소화시키는 것이 요구된다[2,4,5].

접근 시간은 방송채널에서 데이타 접근의 효율성을 의미하며 적응 시간은 클라이언트의 전력소모 정도를 의미한다. 그러므로 인덱스 적용의 목표는 접근 시간을 최소한 증가시키면서 적응 시간을 최대한으로 줄이는 것이다.

5.3 실험 결과

본 논문에서 성능 분석은 세가지 기법, 즉, 인덱스 정보에 개별 데이타의 방송 시간을 포함하지 않는 경우(참고문헌 [12] 중에서 HintsOnly 기법임-이하 HINTSONLY로 표기), 그리고 본 논문에서 제안하는 인덱스 및 방송 데이타 구성 기법 두 가지인 NIPX와 NIPO를 비교하였다. 예측기법으로서 앞서 기술한 방송 서버의 큐 상태에 기반한 예측 기법(EQ로 표기)과 데이타 요구율에 기반한 방송 데이타의 예측 기법(EP로 표기)을 적용한 방송 스케줄링 기법의 성능을 비교 분석한다.

표 2는 2가지 예측 접근법과 본 논문에서 제안하는 인덱스 및 데이타 구성 방안 두 가지인 NIPX와 NIPO를 적용한 4가지 스케줄 생성 접근법을 서술하였다. 예를 들면 EQ-NIPO는 큐기반 예측과 NIPO를 갖는 스케줄 생성 접근법이다.

표 2 실험 스케줄링 기법

	큐 상태에 기반한 예측 기법 (EQ)	데이타 요구율에 기반한 예측 기법 (EP)
NIPO 스케줄 생성 (NIPO)	EQ-NIPO	EP-NIPO
NIPX 스케줄 생성 (NIPX)	EQ-NIPX	EP-NIPX

5.3.1 평균 적응 시간 분석

그림 8은 제안된 기법들의 평균 적응 시간을 보여준다. 인덱스의 크기가 증가할 때 적응 시간 또한 증가한다. 왜냐하면 더욱 많은 데이타 항목을 인덱스 집합에 포함하면 인덱스 구간이 커지게 되고, 클라이언트는 그 구간 동안에 활동 모드로 존재하기 때문이다. 각 스케줄 생성 기법에서 EP 접근법은 EQ 접근법보다 평균 적응 시간이 더 작다. 이 결과는 더욱 올바른 예측에 의한 인덱스 집합은 평균 적응 시간을 줄인다는 것을 나타낸다.

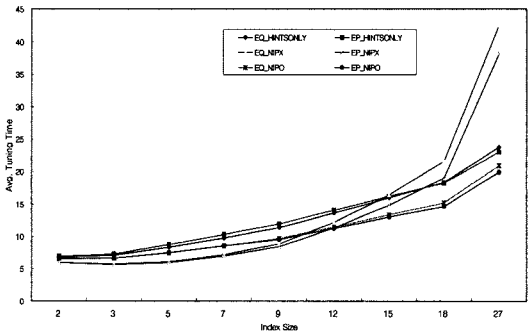

그림 8 평균 적응 시간

인덱스의 크기가 전체 데이타베이스의 10% 이하인 경우 기존 연구결과인 HINTSONLY에 비해서 본 논문에서 제안한 두 기법의 적응 시간이 상대적으로 작았다. 즉 인덱스 정보내에 방송 시간 정보를 포함함으로써 요구한 데이타 항목의 방송을 기다리는 클라이언트들이 수면 모드에 있는 시간을 증가시켜 효과적으로 전력 소모를 줄일 수 있음을 의미한다. 한편 인덱스 크기가 10% 이하 구간에서는 NIPO 보다 NIPX 기법이 성능이 우수함을 알 수 있다. NIPX는 Index miss의 경우라도 인덱스의 크기가 10% 이하로 크지 않아 데이타 집합(BS)이 방송되는 동안 클라이언트가 활동 모드로 있는 시간이 크게 증가하지 않기 때문이다. 인덱스 크기가 전체 데이타베이스의 10% 이상 커지면 NIPX 기법은 NIPO 보다 오히려 성능이 저하되었고 특히 20% 이상 크기가 되면 HINTSONLY 보다도 성능이 감소하였다. 이는 NIPX의 경우 데이타 집합이 방송되는 동안 클라이언트가 데이타를 수신하기 위하여 계속적으로 활동 모드로 지내게 되며, 데이타 집합이 많아지는 경우 즉 인덱스의 크기가 커지는 경우 그 효과가 더욱 증가하기 때문이다.

NIPX 접근법은 기존 HINTSONLY와 비교하여 인덱스 크기 2에서 12%~14% 적응 시간 감소를, 인덱스 크기 9에서 22%~30%, 인덱스 크기 18에서 -18%~-5%, 인덱스 크기 27에서 -79%~-65%의 감소 효과를 보인다. NIPO 접근법은 기존 HINTSONLY와 비교하여 인덱스 크기 2에서 3%~5% 적응 시간 감소를, 인덱스 크기 9에서 15%~20%, 인덱스 크기 18에서 17%~20%, 인덱스 크기 27에서 12%~14%의 감소 효과를 보인다. 기존 기법에 대한 제안 기법의 적응 시간 감소 효과를 정리하면 표 3과 같다.

이 실험 결과는 주문형 데이타 방송을 위한 인덱스를 적용할 때 인덱스 정보내에 방송 시간 정보를 포함하고, 다음 인덱스 시점(NIP)를 포함함으로써 클라이언트의 적응 시간을 상당히 크게 줄일 수 있음을 보여 준다.

표 3 평균 적응 시간 감소 비율 (IS : Index Size)

단위 : 백분율(%)

제안기법	IS / 예측기법	2	6	9	18	27
NIPX	EQ	12.2	27.1	22.1	-17.9	-78.8
	EP	14.1	32.3	29.6	-4.3	-65.3
NIPO	EQ	2.3	11.6	14.6	17.0	12.0
	EP	5.2	16.6	20.1	19.5	13.6

5.3.2 평균 접근 시간 분석

제안 기법은 인덱스 구성과 데이타 구성에 따라 방송 대역폭 이용에 있어서 약간의 오버헤드를 갖는다. 왜냐하면 인덱싱 정보가 부가적인 대역폭 소비를 필요로 하기 때문이다. 또한 데이타 구성요소 NIP 정보가 부가적인 대역폭 소비를 필요로 하기 때문이다. 그러므로 제안 기법은 HINTSONLY 기법보다 평균 접근 시간이 길어지게 된다. 그림 9는 각 기법들의 평균 접근 시간을 비교하는 결과이다. 인덱스의 크기가 10% 이하인 경우 각 기법들의 평균 접근시간은 유사하나, 10% 보다 커지면 NIPO가 가장 큰 접근 시간을 가지며, NIPX는 HINTSONLY와 큰 차이를 보이지 않았다. 각 스케줄 생성 기법에서 EP 접근법은 EQ 접근법보다 평균 접근 시간이 더 작다. 그러나 표 4에서 정리하였듯이 접근 시간의 증가는 4% 이내로서, 앞에서 살펴보았던 30% 가까운 적응시간의 감소 효과에 비하면 매우 작은 수치라 할 수 있다.

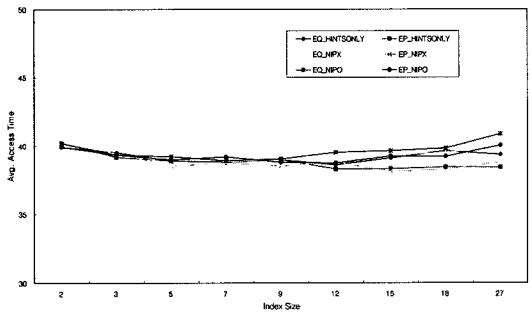

그림 9 평균 접근 시간

표 4 평균 접근 시간 증가 비율 (IS : Index Size)

단위 : 백분율(%)

제안기법	IS / 예측기법	2	6	9	18	27
NIPX	EQ	0.0	0.3	-0.0	-0.3	1.2
	EP	0.1	-0.4	-1.1	-0.5	1.0
NIPO	EQ	0.7	0.3	0.2	0.5	3.9
	EP	-0.7	-0.6	-0.4	2.0	4.2

5.3.3 접근 편중도 분석

이번 절에서는 제안된 접근법들에서 접근 편중도에 따른 평균 적응 시간과 평균 접근 시간의 성능 차이를 비교한다. 실험에서 인덱스의 크기는 9이며 접근 편중도(Θ)는 0.0에서 1.5이다.

그림 10의 결과에서 알 수 있듯이 제안된 기법들은 접근 편중도가 증가할 때 평균 적응 시간은 감소함을 보인다. 인덱스의 크기는 9일 때 NIPX가 NIPO보다 성능이 우수하고 두 제안 기법은 HINTSONLY 기법보다 성능이 우수하다. 이전 절의 관찰로부터 EP에 기반한 방송 예측이 EQ에 기반한 방송 예측보다 더 정확하며 좋은 성능을 얻는다는 것을 알 수 있다. NIPX와 HINTSONLY를 비교할 때 적응 시간의 감소비율은 30%~16%, NIPO와 HINTSONLY를 비교할 때 적응 시간의 감소비율은 21%~5%를 보인다.

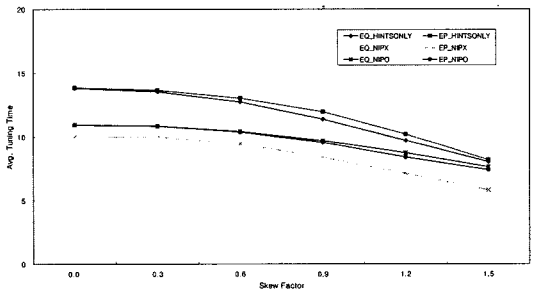

그림 10 접근 편중도에 따른 평균 적응 시간 변화

접근 편중도가 증가할 때 인기있는 데이타 항목에 대한 클라이언트의 요청은 증가한다. 같은 데이타 항목을 요청한 모든 활동 모드인 클라이언트는 방송에 의하여 데이타 항목이 동시에 만족되고 모든 스케줄링 접근법

에서 접근 시간은 감소한다.

그림 11의 결과를 보면, 제안된 기법들은 접근 편중도가 증가할 때 평균 접근 시간은 감소한다. NIPX와 HINTSONLY를 비교할 때 접근 시간의 증가 비율은 -1.9%~2.2%, NIPO와 HINTSONLY를 비교할 때 접근 시간의 증가 비율은 2.6%~0.02%로서 매우 근소한 차이를 보인다.

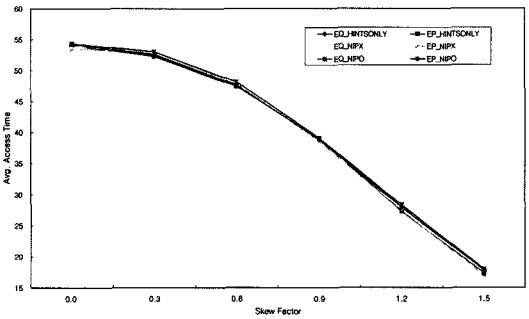

그림 11 접근 편중도에 따른 평균 접근 시간 변화

6. 결 론

본 논문에서는 이동 컴퓨팅 환경에서 주문형 데이타 방송을 위한 인덱스 구성과 이를 지원하기 위한 방송 데이타 구성 방안을 제안하였다. 주문형 방송에서 방송될 데이타의 결정은 방송 서버에서 스케줄링 알고리즘에 의하여 동적으로 결정된다.

방송 서버의 현재 큐 상태와 데이타 요구율을 관찰하여 데이타 집합을 선정하고 방송 예정시간 정보를 포함한 인덱스를 구성하는 방안을 제시하였다. 또한 방송 데이타를 구성함에 있어 다음 인덱스 시점(NIP)를 포함시킴으로써 인덱스 정보가 방송된 이후에 데이타를 요구한 클라이언트가 다음 인덱스를 만날때까지 계속적으로 방송 채널을 감시할 필요가 없도록 하는 방안을 제시하였다.

제안 기법은 예측된 방송 스케줄을 기반으로 구성된 인덱스를 미리 방송함으로써 클라이언트가 요구한 개별 데이타에 대한 방송 시간을 알 수 있고 다음 인덱스 정보의 방송 시점에 대한 정보를 방송 데이타에 포함시킴으로써 클라이언트가 보다 세밀하게 방송 채널의 수신 여부를 결정할 수 있게 하여 이를 통해 클라이언트의 적응 시간을 효과적으로 감소시킴으로써 전력소모를 줄이게 된다.

실험을 통하여 제안 기법의 성능을 평가하였으며 실험 결과는 제안 기법이 추가적인 작은 접근시간 부담을 갖지만 클라이언트의 적응 시간을 효과적으로 감소시킬 수 있음을 검증하였다. 인덱스 크기가 데이타베이스의 10% 보다 작은 경우 NIPX 기법의 적응 시간이 작았으며, 인덱스 크기가 데이타베이스의 10% 보다 큰 경우 NIPO 기법의 적응 시간이 작음을 보았다. 대부분의 데이타베이스에서 인덱스의 크기가 10% 보다 작을 것으로 여겨지므로 NIPX 기법이 효과적으로 사용될 것으로 생각된다.

본 논문에서는 방송 데이타의 크기가 고정된 것으로 가정하였다. 앞으로 가변적인 크기를 갖는 데이타의 주문형 방송을 위한 인덱스의 구성에 대한 연구를 진행할 예정이다.

참 고 문 헌

[1] T. Imielinski, B. R. Badrinath. Mobile Wireless Computing : Challenge in Data Management. *Communications of the ACM*, Vol37, No. 10, October 1994.

[2] S. Acharya, R. Alonso, M. Franklin, S. Zdonik. Broadcast Disks : Data Management for Asymmetric Communication Environments, *In Proc. of ACM SIGMOD Conf. on Management of Data*, pages 199-210, 1995.

[3] S. Acharya, M. J. Franklin, and S. Zdonik. Balancing push and pull for data broadcast. In *Proc. of ACM SIGMOD Conf. on Management of Data*, pages 183-194, 1997.

[4] T. Imielinski, S. Viswanathan, B. R. Badrinath. Data on the Air : Organization and access. *IEEE Transactions on Data and Knowledge Engineering*, 9(3):353-372, 1997.

[5] T. Imielinski, S. Viswanathan, B. R. Badrinath. Energy efficient indexing on air. *In Proc. of ACM SIGMOD Intl. conference on Management of Data*, 1994.

[6] N. Shivakumar and S. Venkatasubramanian. Energy-efficient indexing for information dissemination in wireless systems. *ACM-Baltzer Journal of Mobile Networks and Nomadic Applications (MONET)*, 1996.

[7] D. Aksoy and M. Franklin. Scheduling for large-scale on-demand data broadcasting. In *Proc. of IEEE INFOCOM Conference*, pages 651-659, March 1998.

[8] S. Acharya and S. Muthukrishnan. Scheduling on-demand broadcasts: New metrics and algorithms. In *Proc. of ACM/IEEE Internal Conference on Mobile Computing and Networking*, October 1998.

[9] M.-S. Chen, P. S. Yu, K.-L. Wu. Indexed Sequential Data Broadcasting in Wireless Mobile Computing. *In Proceedings of the 17th International Conference on Distributed Computing Systems(ICDCS97)*, Baltimore, maryland, USA, 1997.

[10] T. Imielinski, S. Viswanathan, B. R. Badrinath. Power efficient filtering of data on air. *In Proc. of Intl. Conference on Extending Database Technology*, pages 245-258, 1994.
[11] W.-C. Lee and D. L. Lee. Using Signature and Caching Techniques for Information Filtering in Wireless and Mobile Environments. *Journal on Distributed and Parallel Databases*, 4(3):205-227, 1996.
[12] Sangdon Lee, Donald Carney, Stanley B. Zdonik. Index Hint for On-Demand Broadcasting. *In Proc. of intl. Conference Data Engineering*, 2003.
[13] H. Dykeman, M. Ammar, and J. Wong. scheduling algorithms for videotex systems under broadcast delivery. In *Proc. of IEEE Int'l Conference on Communications*, 1 1986.
[14] Dennis D. Wackerly, William Mendelhall, and Richard L. Scheaffer. *Mathematical Statistics With Applications*. PWS Publishing, 5th ed., 1997.
[15] G. Zipf. *Human behavior and the Principle of Least Effort*. Addison-Wesley, Reading, MA, 1949.

강 선 희

1994년 한국방송통신대학 전자계산과(학사). 1997년 전북대학교 정보과학대학원 정보과학과(석사). 1999년~현재 목포대학교 멀티미디어공학과 박사과정. 1998년~현재 목포과학대학 컴퓨터정보과 조교수. 관심분야는 이동 데이타베이스

이 상 돈

1980년~1984년 서울대학교 컴퓨터공학과(학사). 1984년~1986년 서울대학교 컴퓨터공학과(석사). 1987년~1997년 한국통신 연구개발원 선임연구원. 1996년 서울대학교 컴퓨터공학과(박사). 1997년~현재 목포대학교 정보공학부 부교수. 2001년~2002년 브라운대학교 객원교수. 관심분야는 데이타 스트림 관리, 이동 데이타베이스, 웹 데이타베이스