데이타 방송 시스템에서 클라이언트의 요구정보를 이용한 캐싱 전략들의 성능

(Performance of Caching Strategies using Clients' Request Information in Data Broadcast Systems)

신 동 천 †

(Dong Cheon Shin)

요 약 무선 컴퓨팅 환경에서 데이타 방송 기법은 다수의 클라이언트에게 데이타를 전송하는 유용한 기법이다. 일반적으로 낮은 대역폭을 갖는 데이타 방송 시스템에서 캐싱은 대역폭에 대한 클라이언트들의 경쟁을 줄임으로써 응답 시간을 향상시키기 위해 도입된다. 본 논문에서는, 클라이언트가 유지하는 정보를 이용하는 기존 연구와 달리 서버가 유지하는 클라이언트의 데이타 요구에 관한 정보를 이용하는 캐싱 전략들을 제시하고 제안한 전략들의 성능을 시뮬레이션을 통하여 평가한다. 성능 평가에 따르면, 서버에서 유지하는 인기도와 대기 시간 정보를 함께 고려하는 전략이 다른 전략들보다 전반적으로 좋은 성능을 보여 주고 있다.

키워드 : 무선 컴퓨팅, 데이타 방송, 캐싱 전략

Abstract A data broadcast is an efficient technique to deliver data to a number of clients in wireless computing environments. In data broadcast systems, which generally have a narrow bandwidth, a caching is introduced to reduce contention for the bandwidth so that the response time can be decreased. In this paper, unlike previous works that use information maintained by clients, we propose several caching strategies using information on data requests of clients maintained by a server. Then, we evaluate the performance of proposed strategies by using simulation approach. According to the results, the strategy that considers both popularity and waiting time maintained by a server generally shows better performance than other strategies.

Key words : wireless computing, data broadcast, caching strategy

1. 서 론

무선 통신 기술의 급속한 발전은 새로운 컴퓨팅 환경의 등장을 촉진시키고 있다. 새로운 컴퓨팅 환경의 특징으로 무선과 이동성을 생각할 수 있다[1]. 무선망은 기존의 유선망과 달리 상대적으로 낮은 대역폭(bandwidth), 잦은 단절 등으로 특징 지울 수 있다. 한편, 이동성은 컴퓨팅 능력의 제한성, 낮은 배터리 수명 등으로 특징 지울 수 있다. 이러한 환경적인 제약으로 인하여 기존에 개발된 여러 가지 알고리즘이나 기법들이 이러한 새로운 컴퓨팅 환경에서는 좋은 성능을 보이기 어렵거나 그대로 도입하기 어렵게 된다. 예를 들어, 캐싱 전략, 질의 처리, 트랜잭션 처리 등을 위한 전략들이 효율적이기 위해서는 무선과 이동성이라는 새로운 환경을 반영하여야 함은 당연하다[2].

그림 1은 무선 컴퓨팅 환경의 일반적인 구조를 보여주고 있다[1]. 무선 컴퓨팅 환경은 크게 고정(fixed) 호스트와 이동 호스트(mobile host: MU)로 구성된다. 이동 호스트는 무선과 이동성을 갖는 클라이언트이며 고정 호스트는 유선 네트워크 상의 전통적인 호스트이다. 이동 지원 시스템(mobile support system: MSS)은 본 논문에서 서버라고 지칭하는 특별한 종류의 고정 호스트로 이동 호스트와의 통신 인터페이스를 담당한다. 하나의 MSS가 이동 호스트에 서비스 할 수 있는 영역을 셀(cell) 이라고 한다. 따라서, 동일한 셀 안에 있는 이동 호스트들은 무선 채널을 통하여 자신들의 MSS와 통신하게 된다.

무선 환경에서 데이타 방송 시스템은 제한된 대역폭

· 이 논문은 2004학년도 중앙대학교 학술연구비 지원에 의한 것임
† 종신회원 : 중앙대학교 정보시스템학과 교수
dcshin@cau.ac.kr
†† 논문접수 : 2004년 12월 14일
심사완료 : 2005년 4월 18일

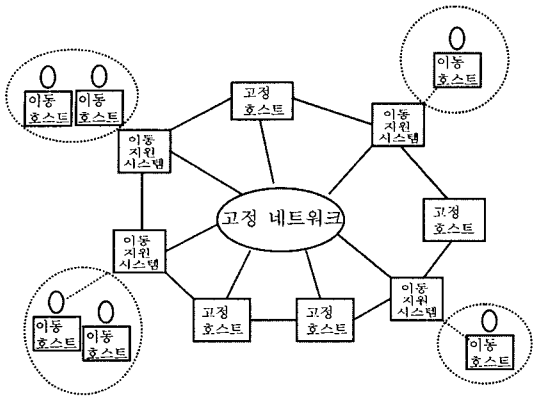

그림 1 무선 컴퓨팅 환경

에서 대량의 데이타를 다수의 클라이언트들에게 효율적으로 전달하는 방법중의 하나로 인식되어 왔다. 데이타 방송 시스템에서는 클라이언트에 캐시를 두어 필요한 데이타가 방송되기를 기다리지 않고 액세스하게 함으로써 낮은 대역폭으로 인한 제약을 극복하여 방송 시스템의 효율성을 높일 수 있다. 캐시를 도입하는 경우 해결해야 할 중요한 문제 중의 하나는 서버와 클라이언트 사이에 데이타의 일관성을 유지하는 문제이다[3, 4]. 전통적인 시스템 환경에서와 마찬가지로 해결해야 할 또 다른 중요한 문제는 캐시 내의 데이타를 대체하는 캐싱 전략이다.

전통적인 시스템 환경을 위한 여러 가지 캐싱 전략들이 제안되었다[5]. 이러한 대체 전략들은 이동성과 무선 환경의 특성들을 고려하지 않은 전략들이므로 데이타 방송 환경에서 좋은 성능을 보이기 어렵다. 따라서, 데이타 방송 환경을 위한 여러 가지 전략들이 제안되었다[6-9]. 데이타 방송 환경에서 제안된 대부분의 전략들은 히트율과 미스 비용을 고려하기 위한 정보들을 클라이언트 자신이 생성한다.

데이타 방송 시스템에서 캐싱 전략은 방송 알고리즘과 깊은 연관성을 갖게 됨은 명백하다. 예를 들어, 자주 방송되는 데이타는 그렇지 않은 데이타에 비해 상대적으로 대기 시간이 작으므로 캐시에 존재할 가치가 떨어진다. 뿐만 아니라, 데이타 방송 시스템에서는 하나의 셀 안에 있는 다수의 클라이언트들에게 데이타를 방송하게 되므로 다른 클라이언트들의 데이타 액세스에 대한 특성을 반영한 정보를 캐싱 전략에 사용하는 것이 바람직하다. 따라서, 데이타 방송을 담당하는 서버로부터 캐싱 전략에 필요한 정보를 받아 이용할 수 있는 캐싱 방법을 고려할 수 있다.

본 논문에서는 히트율을 높이고 미스 비용을 줄이기 위해 필요한 정보들을 서버로부터 받아 이용하는 캐싱 전략들을 제시한다. 이를 위해, 서버로부터 받아 캐싱 전략에 사용될 수 있는 정보들을 도출한다. 그리고 성능 평가를 위해 클라이언트 자신이 필요한 정보를 직접 유지하여 이용하는 전략들 및 전통적인 LRU(least recently used) 전략과 시뮬레이션을 통해 성능을 비교 평가 한다.

본 논문의 구성은 다음과 같다. 제안하는 캐싱 전략과 직접 관련된 연구들은 2장에서 간략히 기술한다. 3장에서는 캐싱 전략들이 필요로 하는 정보들을 소개하고 4장에서는 캐싱 전략들을 제시한다. 5장에서는 성능 평가에 대해 기술한다. 끝으로, 6장에서는 결론을 맺는다.

2. 관련 연구

이 장에서는 본 논문에서 제시하고자 하는 캐싱 전략과 직접 관련이 있는 데이타 방송 환경에서 제안된 전략들을 간략히 소개한다. 표 1에서 보는 바와 같이 관련 전략들은 캐싱에 필요한 정보를 클라이언트 자신이 유지하는 정보를 이용한다.

표 1 캐시 대체 전략

전략이름	클라이언트 정보 이용		서버 정보 이용
	히트율 향상 고려	미스 비용 감소 고려	
PIX[6]	O	O	X
PT[6]	O	O	X
CF[10]	X	O	X
Gray[8]	O	O	X
PWT[7]	O	O	X

- PIX(Probability of access and Frequency of broadcast): 각 페이지의 접근 확률과 방송 주기 내에 방송되는 페이지의 방송 빈도를 이용하여 대체할 페이지를 선택하는 알고리즘이다.각 페이지의 PIX 값은 접근확률/방송빈도로, PIX 값이 작은 페이지를 캐쉬에서 대체할 페이지로 선정된다. 따라서, 방송빈도가 상대적으로 매우 적거나 페이지의 접근 확률이 상대적으로 매우 큰 페이지가 캐쉬에 남게 되므로 히트율과 대기 시간을 함께 고려하는 전략이라 할 수 있다.
- PT(Probability and Time to broadcast): PIX가 대기시간 고려를 위해 사용하는 정적인 값인 방송 빈도수 대신에 다음 방송까지 걸리는 실질적인 시간을 사용하는 전략이다. 각 페이지의 PT값은 접근확률*(다음 방송 시간-현재 시간)으로, PT 값이 가장 작은 페이지가 대체할 페이지로 선정된다.
- CF(Closest First): 페이지 미스가 발생할 경우 방송 스케쥴에서 가장 먼저 방송될 페이지를 대체할 페이지로 선정하는 알고리즘이다. 따라서, CF는 LRU와

반대 개념의 알고리즘으로, LRU가 히트율을 높이기 위한 알고리즘인데 반하여 CF는 미스 비용 즉, 대기 시간을 고려하는 전략이다.

- Gray: 히트율과 대기시간 모두를 고려하기 위해CF와 LRU전략을 병행하여 대체할 페이지를 선택하는 알고리즘이다.모든 페이지는 {black, gray, white} 세 가지 상태를 부여 받는다.black은 현재 단계에서 요청된 페이지를 의미하고, gray는 이전 단계에서 요청된 페이지를, white는 요청되지 않은 페이지를 의미한다.현재 단계에서는 페이지가 요청되는 경우 black 표시를 하고, 페이지 미스가 발생할 경우 white로 표시된 페이지 중에서 CF를 이용하여 대체할 페이지를 선택한다.
- PWT(Priority and Waiting Time): 각 페이지에 클라이언트가 요청하는 횟수인 인기도(P)와 페이지 미스로 인한 데이타 요청 후 방송 때까지 걸리는 시간인 대기시간(waiting time)을 이용하는 전략이다. 이 값이 가장 작은 페이지가 대체할 페이지로 선정된다.

3. 캐싱 전략을 위한 정보들

3.1 클라이언트 정보들

데이타 방송 시스템의 캐싱 전략에서 히트율과 미스 비용을 고려하기 위해 [7]은 클라이언트가 유지하는 정보로 인기도와 대기시간을 도입하고 있다. 인기도는 클라이언트의 데이타 액세스 빈도에 대한 척도이며 대기시간은 클라이언트가 서버에게 데이타 요청을 한 후 방송되기까지의 시간이다. 따라서, 인기도는 히트율을 고려하기 위해 유지하는 정보이며 대기시간은 미스 비용을 고려하기 위한 정보가 된다. [7]은 인기도와 대기시간에 가중치 개념을 도입하고 있으나 본 논문의 주 목적인 서버가 유지하는 정보를 이용한 전략에 가중치 개념을 도입하지 않으므로 클라이언트가 유지하는 정보인 인기도와 대기 시간에 가중치를 도입하지 않고 사용하기로 한다.

인기도는 클라이언트의 데이타에 대한 관심 정도에 따라 데이타가 액세스 되는 빈도를 수치화한 값이 된다. 따라서, 특정 데이타의 인기도는 클라이언트가 액세스한 횟수로 정의할 수 있다. 본 논문에서는 이전에 인기가 있었던 데이타와 최근에 인기가 증가하고 있는 데이타를 차별화하여 최근에 액세스가 증가하고 있는 데이타가 캐시에 존재할 가능성을 높이기 위해 식 (1)과 같이 인기도(Pc)를 정의하여 사용한다. 따라서, 동일한 액세스 횟수를 갖고 있으면 최근에 액세스가 많은 데이타가 캐시에 존재할 가능성을 높게 할 수 있다. 식 (1)에서 A_n은 특정 데이타에 대한 액세스 횟수를 의미하고 CT와 RT는 각각 현재 시간과 특정 데이타에 대해 가장 최근에 액세스한 시간을 의미한다.

$$Pc = A_n / (CT-RT) \quad (1)$$

특정 데이타에 대한 대기시간(WTc)은 가장 최근의 대기시간(CWTr)을 사용한다. 이는 서버의 데이타 방송 환경에 대기 시간은 종속적이므로 가장 최근의 데이타 방송 환경을 반영하기 위함이다(식 2).

$$WTc = CWT_r \quad (2)$$

3.2 서버 정보들

동일 셀 안에서 서비스 받는 다수의 다른 클라이언트들의 액세스 요청 패턴을 반영하는 정보를 캐싱 전략에 이용하기 위해 서버가 유지하는 정보로 역시 인기도와 대기시간을 고려할 수 있다. 그러나 클라이언트가 유지하는 인기도와 대기시간과는 달리, 서버가 유지하는 인기도는 각 데이타에 대해 얼마나 많은 클라이언트들이 요청했는지를 나타내며 대기시간은 서버가 클라이언트로부터 데이타에 대한 요청을 접수해 방송 서비스되기까지 큐에서 대기한 시간을 나타낸다. 인기도는 히트율을 고려하기 위한 정보이며 대기시간은 미스비용을 고려하기 위한 정보이다. 서버는 이러한 정보를 방송하는 데이타와 함께 클라이언트에게 보내 캐싱 전략에 이용할 수 있도록 한다.

특정 데이타에 대한 인기도는 가장 최근 방송부터 현재 방송 당시에 그 데이타를 요청한 클라이언트의 수가 된다. 또한, 최근에 요청 수가 많은 데이타가 캐시에 존재할 가능성을 높이기 위해 식 (3)과 같이 인기도(Ps)를 정의하여 사용한다. 식 (3)에서 C_n은 가장 최근 방송부터 현재 방송 당시까지 특정 데이타에 대해 요청한 클라이언트 수를 의미하고 CT와 RBT는 각각 현재 시간과 특정 데이타가 가장 최근에 방송된 시간을 의미한다.

$$Ps = C_n / (CT-RBT) \quad (3)$$

특정 데이타가 여러 클라이언트로부터 시간차를 두고 요청되는 경우에 각 클라이언트 입장에서 보면 동일한 데이타일지라도 큐에서 대기한 시간이 다르게 된다. 따라서, 해당 데이타에 대한 대기 시간으로 각 클라이언트의 대기 시간들의 평균값을 사용할 수도 있다. 그러나, 대기 시간 정보를 유지하여야 하는 서버 입장에서 보면 특정 데이타의 대기 시간(WTs)은 처음 요청되었을 때부터 방송 될 때까지의 시간이므로 여기서는 이 시간(SWTf)을 대기 시간으로 사용한다(식 4).

$$WTs = SWT_f \quad (4)$$

4. 캐싱 전략

데이타 방송 시스템 환경에서 가장 이상적인 캐싱 전략은 히트율을 높이는 동시에 미스 비용을 최소화하는 것이라 할 수 있다. 따라서 히트율을 높이기 위해서는 인기도를 고려하고 미스 비용을 최소화하기 위해서는

대기 시간을 고려하여 다음과 같은 직관을 바탕으로 캐싱 전략을 수립한다.

• H1: 인기도가 높은 데이타는 캐시에 존재하여야 한다.

클라이언트가 필요로 하는 데이타는 캐시에 존재하거나, 방송 중, 혹은 서버에게 요청을 하게 된다. 방송 중인 경우는 현재 방송 주기 동안에 해당 데이타가 방송되기를 기다려야 하며 서버에게 요청을 하는 경우는 다음 방송 주기에 방송 되기를 기다려야 한다. 캐시에 존재하는 경우는 바로 액세스할 수 있으므로 인기도가 높은 데이타는 캐시에 존재하게 하는 것이 비용을 최소화하는 것이 된다.

• H2: 대기 시간이 큰 데이타는 캐시에 존재하여야 한다.

요청한 데이타의 대기 시간이 크게 되면 미스 비용이 크게 된다. 따라서, 대기 시간이 큰 데이타일수록 캐시에 존재하여야 미스 비용을 줄일 수 있다.

H1과 H2를 반영하는 전략들을 다음과 같이 수립할 수 있으며 해당 값이 가장 작은 데이타를 대체할 데이타로 선정한다.

1) 클라이언트 정보 이용 전략

*인기도 전략(CP): 클라이언트의 인기도 정보만을 이용하는 전략

$$CP = Pc$$

*대기 시간 전략(CWT): 클라이언트의 대기 시간 정보만을 이용하는 전략

$$CWT = WTc$$

*인기도/대기 시간 전략(CPWT): 클라이언트의 인기도 및 대기 시간 정보를 이용하는 전략

$$CPWT = Pc * WTc$$

2) 서버 정보 이용 전략

*인기도 전략(SP): 서버의 인기도 정보만을 이용하는 전략

$$SP = Ps$$

*대기 시간 전략(SWT): 서버의 대기 시간 정보만을 이용하는 전략

$$SWT = WTs$$

*인기도/대기 시간 전략(SPWT): 서버의 인기도 및 대기 시간 정보를 이용하는 전략

$$SPWT = Ps * WTs$$

인기도와 대기시간을 함께 고려하는 전략에서 인기도와 대기시간의 곱을 사용하고 있음은 이들 값의 단위가 다르며 아울러 인기도가 높을수록 그리고 대기시간이 클수록 캐시에 존재할 가치가 커짐을 반영하기 위함이다. 곱을 사용하고 있으므로 이들간의 가중치 반영은 현재 전략에서는 의미가 없게 된다.

위의 전략들에 대한 이해를 돕기 위해 표 2는 4개의 데이타 A, B, C, D 에 대한 인기도와 대기시간의 값을 임의로 가정할 때 각 전략에서 대체 데이타로 선정되는 것(o로 표시됨)을 예시하고 있다. 예를 들어, 서버가 유지하는 정보를 이용하는 SPWT 전략을 사용하는 경우 4개의 데이타 중 최소값인 18(6*3)을 갖는 데이타 C 가 대체 데이타로 선정됨을 보여 주고 있다.

표 2 캐싱 전략 예시

데이타	클라이언트 유지정보		서버 유지 정보		클라이언트 정보 이용 전략			서버 정보 이용 전략		
	Pc	WTc	Ps	WTs	CP	CWT	CPWT	SP	SWT	SPWT
A	2	2	4	6				O		
B	3	1	10	2		O			O	
C	1	2	6	3	O		O			O
D	5	3	7	5						

서버에서 인기도와 대기 시간을 구성하는 부분과 관련된 다음과 같은 내용을 알고리즘 형식으로 기술한다: (1) 클라이언트의 페이지 x에 대한 요청 메시지를 처리하는 *메시지관리자()*, (2) RxW기법에 의한 방송 프로그램 및 클라이언트 요청내용을 정해진 비율로 혼합하여 송출하는 *방송()*.

```
BEGIN
1. /* WaitingQueue 는 클라이언트의 요청 메시지 대기 큐 */
2. IF ∃ x such that x ∈ WaitingQueue THEN
3.    페이지 x의 인기도를 1 증가시키고, WaitingQueue 의 뒤에 추가
4. ELSE
5.    페이지 x 의 인기도를 1 로 설정하고,
      대기시작시간을 현재시간으로 설정한 후, WaitingQueue 의 뒤에 추가
6. END IF
END
```

알고리즘 1 메시지관리자(x)

```
BEGIN
1. WaitingQueue = Null;
2. NoofBroadcastedData = 0; /* 지금까지 방송된 페이지의 수 */
3. NoofRequestedData = 0;   /* 지금까지 클라이언트에 의해 요청된 페이지의 수 */
4. DO WHILE (TRUE)
5.    IF (NoofBroadcastedData – NoofRequestedData) / NoofBroadcastedData < 0.9 THEN
6.       RxW 방송 알고리즘에 따라 다음 차례로 방송되어야 할 페이지 x 를 방송
7.    ELSE
8.       WaitingQueue 의 맨 앞에 있는 페이지 x 와,
         그것의 인기도 및 대기시간을 방송
9.       NoofRequestedData++;
10.   END IF
11.   NoofBroadcastedData++;
12.END DO
END
```

알고리즘 2 방송( )

5. 성능 평가

5.1 성능 평가 환경

클라이언트와 서버의 정보를 이용하는 캐싱 전략들과 기존의 LRU 전략의 성능을 시뮬레이션을 통하여 평가하였다. 시뮬레이션 모델은 그림 2와 같으며 CSIM[11]

을 이용하여 구현하였다.

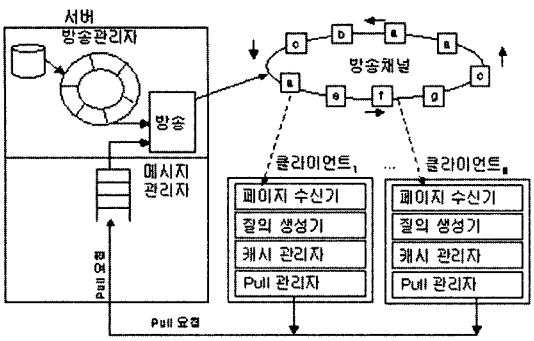

그림 2 시뮬레이션 모델

각 클라이언트는 페이지 수신기, 질의 생성기, 캐시 관리자, Pull 관리자의 4개의 프로세스로 구성된다. 먼저 질의 생성기에 의해 하나의 읽기 연산이 제기되면 캐시 관리자를 통해 해당 데이타가 캐시에 존재하는지 확인하게 된다. 만일 캐시에서 해당 데이타를 찾을 수 없을 경우에는 서버에 요청(pull)을 제기하게 된다. 서버는 RxW[12] 방송 기법에 따라 요청된 데이타 페이지를 방송한다. 뿐만 아니라, 클라이언트 캐싱을 위해서 서버측에서 수집된 각 페이지의 인기도와 큐에서의 대기 시간을 필요로 하기 때문에 이러한 정보가 추가적으로 클라이언트에 전달된다.

자신이 필요로 하는 데이타를 획득한 클라이언트는 또 새로운 데이타 읽기 연산을 제기할 준비를 하게 된다. 각 클라이언트는 하나의 페이지를 요청하여 받을 때까지 새로운 페이지에 대한 연산을 제기하지 않기 때문에 시뮬레이션 중 작업의 부하는 클라이언트의 수로 일정하게 유지될 수 있으므로 폐쇄된 큐잉 모형(closed queuing model)을 채택한 것으로 볼 수 있다. 표 3 (a)와 표 3 (b)는 각각 서버와 클라이언트에 관련된 주요 매개변수를 보여준다.

표 3 주요 매개변수

(a) 서버 주요 매개변수

매개변수	설명	값
db_size	데이타 페이지의 수	5000
down_time	서버가 한 페이지를 방송하면 클라이언트에 도달될 때까지 소요되는 시간	0.1초
acc_range	한 순간 한 클라이언트가 액세스하는 데이타 페이지의 범위	1000
hotarea_size	acc_range에 있는 데이타 페이지 중 핫영역(집중적으로 액세스되 부분)의 크기	100
hotacc_rate	핫영역 액세스 비율	0.95

(b) 클라이언트 주요 매개변수

매개변수	설명	값
num_clients	클라이언트의 수	100~300
Cache_size	클라이언트 캐시의 크기	10
Think_time	클라이언트의 각 요청 간 대기시간	3초
Upload_time	클라이언트가 페이지를 요청하는데 소요되는 시간	0.1초
cache_acc_time	캐시에 있는 한 페이지를 액세스하는데 소요되는 시간	0.001초

데이타베이스의 데이타 페이지의 수를 실제 이동 응용에서 이용되는 크기에 비해 적은 5000으로 설정하였는데 이는 상대적으로 적은 수의 클라이언트 간에 적정 수준의 방송 히트율을 확보하기 위한 것이다. 클라이언트는 자신이 필요로 하는 데이타가 캐시에 포함되어 있지 않을 경우 서버에 해당 데이타를 직접 요청하여 얻게 되는데 이러한 요청은 서버의 큐에 보관되어 처리된다. 클라이언트가 제기한 데이타의 요청에 대해 해당 데이타를 얻은 후에 일정한 시간이 경과하는 동안 새로운 연산을 제기하지 않도록 하기 위해서 think_time을 3초로 설정하였다.

일반적으로 데이타 방송 환경에서 통신채널의 대역폭은 상향과 하향에 차이가 있다. 그러나 서버와 클라이언트 사이에 하나의 메세지(데이타)를 주고 받는데 소요되는 시간의 관점에서 보면 거의 동일한 것으로 볼 수 있다. 따라서 서버가 요청된 데이타를 클라이언트로 방송하면 각 클라이언트는 적어도 down_time 만큼의 시간, 즉 0.1초가 지나야 해당 데이타를 읽을 수 있으며 또한 데이타 요청을 서버로 전송하기 위해서도 upload_time 만큼의 시간, 즉 0.1초가 필요한 것으로 설정하였다. 캐시로부터 데이타를 읽는데 소요되는 시간도 응답 시간에 영향을 줄 수 있으므로 캐시로부터 하나의 데이타 페이지를 액세스하는 시간, 즉 cache_acc_time을 0.001초로 설정하였다.

클라이언트마다 선호하는 데이타를 다르게 만들기 위해서 각 클라이언트가 한 순간 액세스하는 데이타의 범위(acc_range)를 1000으로 설정하여 전체 5000개의 데이타 페이지 중 1000개에 해당되는 부분만 액세스하도록 하였다. 이 1000개의 데이타 페이지 중에서 집중적으로 액세스하는 부분(hotarea_size)을 100으로 정하여 hotacc_rate의 값을 조정함으로써 각 클라이언트가 선호하는 데이타 페이지를 원하는 값으로 조절할 수 있도록 하였다. 본 실험에서는 캐시의 히트율을 높이기 위해서 hotacc_rate를 0.95로 설정함으로써 한 순간 각 클라이언트는 전체 5000개의 데이타 페이지 중 1000개의 페이지만 액세스하게 되고 그 중 95%의 요청이 100개의 부분에 집중되도록 하였다.

5.2 성능 평가 결과

시뮬레이션 결과의 정확성을 제고시키기 위해서 각

시뮬레이션의 초기 단계에 수집된 자료는 결과에 반영하지 않았으며, 또한 각 시뮬레이션에 대해 난수 생성기의 시드 값을 변경시키면서 여러 차례 반복 실험한 결과에 대한 평균을 이용하여 최종 결과를 산출하였다. 데이타 방송 시스템 환경에서는 전통적 시스템 환경과는 달리 히트율이 아니라 캐시 미스에 따른 대기 시간의 고려가 성능에 커다란 영향을 주므로 성능 척도로는 평균 응답 시간을 사용하였다. 하나의 서버가 담당하는 영역인 셀 안의 클라이언트 수가 이동 컴퓨팅 환경에서는 가장 가변적인 요소이므로 방송 시스템의 성능에 미치는 영향을 알아보기 위해 클라이언트 수를 변화시키면서 실험을 하였다. 또한, 캐시 크기에 따른 성능 변화를 실험하였다. 아울러, 데이타 방송 환경에서 캐시 전략은 히트율 뿐만 아니라 미스 비용까지 고려해야 함을 입증하기 위해 각 전략들의 히트율을 함께 평가하였다.

먼저 클라이언트 정보를 이용하는 전략들과 서버 정보를 이용하는 전략들의 성능을 각각 평가한 후 각 그룹에서 좋은 성능을 보이는 전략들과 전통적인 LRU 전략과 성능을 비교하였다.

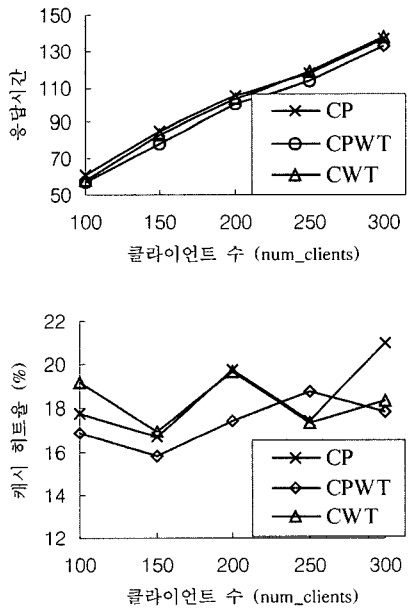

그림 3 클라이언트 정보 전략들의 성능

그림 3은 클라이언트의 정보를 이용하는 전략들의 성능을 보여 주고 있다. 클라이언트 수가 증가함에 따라 각 전략들의 응답 시간도 증가함을 알 수 있다. 이는 클라이언트의 수가 증가함에 따라 캐시 미스로 재기된 데이타 요청이 서버에서 방송될 때까지 대기하는 시간이 증가하기 때문이다. 또한, 전반적으로, 대기 시간과 인기도를 함께 고려한 CPWT가 성능에서 우위를 보여주고 있음을 알 수 있다. 이는 데이타 방송 시스템에서의 캐시 전략은 히트율 뿐만 아니라 미스에 따른 비용도 함께 고려해야 할 당위성을 입증해 주고 있다. 이 사실은 CPWT가 다른 전략들에 비해 상대적으로 낮은 히트율을 보이고 있다는 사실에서도 입증되고 있다. 3가지 전략들의 성능 차이가 크게 나타나지 않은 것은 인기도와 대기 시간을 함께 고려하는 방송 알고리즘인 RxW 의 근본적인 특성에 따라 인기도나 대기 시간만을 고려하는 전략들의 약점을 보완해 준 결과로 볼 수 있다.

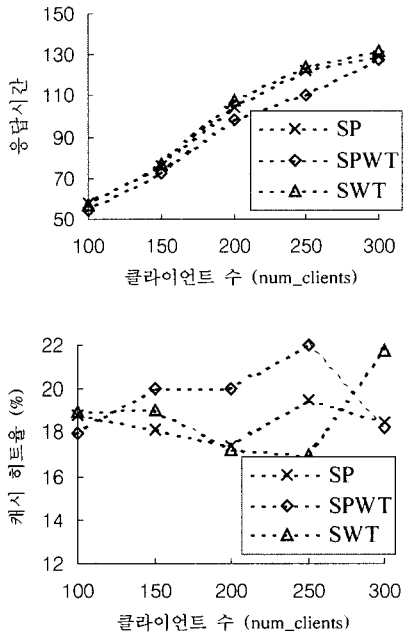

그림 4 서버 정보 전략들의 성능

그림 4는 서버 정보를 이용하는 전략들의 성능을 보여 주고 있다. 클라이언트 수가 증가함에 따라 각 전략들의 응답 시간도 증가함을 알 수 있다. 이는 클라이언트 수가 증가할수록 캐시 미스로 인한 대기 시간이 증가하기 때문이다. SPWT가 전반적으로 비교적 좋은 성능을 보여 주고 있지만 클라이언트 수가 증가할수록 성능의 차는 감소하고 있음을 알 수 있다. 이는 클라이언트의 수가 증가함에 따라 캐시 미스로 재기된 데이타 요청이 서버에서 방송될 때까지 대기하는 시간이 증가할 뿐만 아니라 이 대기시간이 캐싱 전략의 차이로 인해 얻어지는 시간에 비해 훨씬 커진 것에 기인하고 있다. 다시 말해서 캐싱 전략보다는 방송 알고리즘이 전체 시스템 성능에 결정적인 요인으로 작용하고 있음을 알 수 있다. 히트율을 보여 주는 결과에서도 응답 시간에

영향을 주는 요인이 히트율 뿐만 아니라 미스 비용임을 알 수 있다.

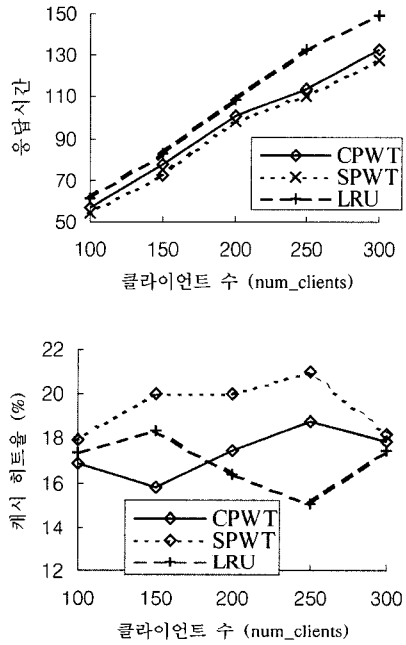

그림 5 LRU 전략과의 성능

전통적인 시스템 전략인 LRU와의 성능 평가 결과를 그림 5는 보여 주고 있다. LRU 전략도 클라이언트 수가 증가함에 따라 당연히 응답 시간도 증가하고 있다. 전반적으로, LRU 전략보다는 히트율과 미스 비용(대기시간)을 함께 고려하는 전략들이 좋은 성능을 보임을 알 수 있다. 또한, SPWT 가 CPWT 보다 다소 좋은 성능을 보이는 것은 캐싱 전략에서 다른 클라이언트들의 데이타 인기도를 고려함으로써 자신이 필요로 하는 데이타를 미리 캐싱하는 효과(precaching)에 기인한 것이라 할 수 있다(hotacc_rate는 95%로 설정함). hotacc_rate 변화에 따른 성능의 변화를 보여 주고 있는 그림 6이 이를 입증해 주고 있다(num_clients는 250으로 설정함). hotacc_rate 가 증가할수록 CPWT와 SPWT의 성능의 차도 다소 증가함을 볼 수 있다. 이는 클라이언트들의 요구 데이타가 집중될수록 미리 캐싱하는 효과가 커지기 때문이다.

클라이언트 수 변화에 따른 평가에서 우위를 보인 CPWT와 SPWT, 그리고 LRU 전략들이 캐시 크기 변화에 따른 성능을 그림 7은 보여주고 있다. 캐시 크기가 커질수록 모든 전략의 응답시간은 당연히 감소하고 있다. 전반적으로, 무선 데이타 방송 환경에서는 히트율만 고려한LRU 전략보다는 히트율과 미스비용을 함께 고려

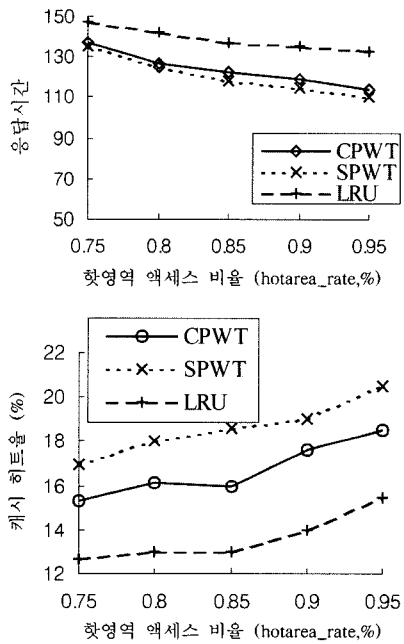

그림 6 hotacc_rate 변화에 따른 성능

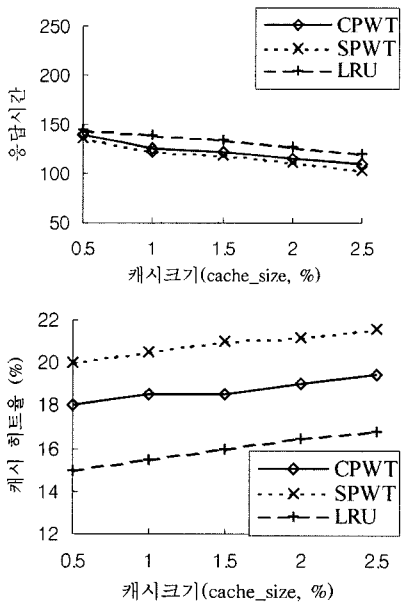

그림 7 캐시 크기 변화에 따른 성능

한 전략들이 좋은 성능을 보임을 알 수 있다. SPWT가 CPWT 보다 다소 좋은 성능을 보이는 것은 동일한 셀에 있는 다른 클라이언트들의 데이타 인기도를 고려하는 캐싱 전략이 자신이 필요로 할 데이타를 미리 캐싱하는 효과에 기인한 것이라 할 수 있다. 따라서, SPWT가 히트율에서도 우위를 보여 주고 있다. 한편, CPWT

와 SPWT의 성능 차이가 크지 않은 것은 캐싱 전략뿐만 아니라 방송 알고리즘 자체가 성능에 많은 영향을 끼치고 있음을 말해주고 있다.

6. 결 론

본 논문의 주된 목적은 데이타 방송 시스템에서 미스 비용을 고려한 전략이 필요함을 기존 전략인 LRU와의 성능 평가를 통해 입증하고, 아울러 캐싱 전략에 사용하는 정보를 기존의 연구처럼 클라이언트가 유지하는 정보를 사용하지 않고 서버가 유지하는 정보를 사용하는 전략들이 어떤 성능을 보여주는 가를 알아 보기 위한 것이다. 이를 위해, 클라이언트와 서버에서 유지 할 수 있는 정보로 인기도와 대기 시간만을 도입하여 여러 전략들을 도출한 후 성능을 비교하였다.

평가 결과에 따르면, 히트율 뿐만 아니라 미스 비용까지 함께 고려하는 전략이 LRU 보다 좋은 성능을 보여주고 있다. 또한, 서버에서 유지하는 정보인 인기도와 대기 시간을 함께 고려하는 전략이 다른 전략보다 다소 좋은 성능을 보여 주고 있다. 한편, 히트율을 평가한 결과는 데이타 방송 시스템에서의 캐싱 전략은 미스 비용을 고려해야 할 당위성을 입증해 주고 있다. 아울러, 본 연구의 결과로부터 데이타 방송 시스템의 성능에 더 큰 영향을 미치는 요소는 데이타 방송 알고리즘의 효율성이라고 판단할 수 있다. 따라서, 방송 알고리즘의 특성을 고려한 캐싱 전략 개발의 필요성을 강조할 수 있다.

참 고 문 헌

[1] D. Barbara, "Mobile Computing and Databases - A Survey," IEEE TransactionsKnowledge Engineering, Vol. 11, No. 1, pp. 108-117, January-/February 1999.

[2] S. K. Madria and B. K. Bhargava, "A Transaction Model to Improve Data Availability in Mobile Computing," Distributed and Parallel Databases, 10(2): pp. 127-160, 2001.

[3] K. Y. Lai, Z. Tari, and P. Bertok, "Cost Efficient Broadcast based Cache Invalidation for Mobile Environments,"Proc. Of the 2003 ACM symposium on Applied Computing, Melbourne, U.S.A., pp. 871-877, March 2003.

[4] X. Shao and Y. Lu, "Maintain Cache Consistency of Mobile Database Using Dynamical Periodical Broadcast Strategy," International Conference on Machine Learning and Cybernetics, pp. 2389-2393, Nov. 2003.

[5] S. Galvin and P. B. Galvin, Operation System Concepts, 4th Edition, Addison Wesley, 1994.

[6] S. Acharya, M. Franklin, and S. Zdonik, "Balancing Push and Pull for DataBroadcast," Proc. of ACM SIGMOD, Tuscon, Arizona, pp. 183-194, May 1997.

[7] Y. J. Lee and D. C. Shin, "A Cache Replacement Strategy for Pull-Based Broadcast in Mobile Computing Environments," Journal of KISS, Software and Application, Vol. 30, No. 8, pp. 780-791, Aug. 2003.

[8] V. Liberatore, "Caching and Scheduling for Broadcast Disk Systems," Technical Report 98-71, UMIACS, 1998.

[9] J. Xu, Q. Hu, W.-C. Lee, and D. L. Lee. "Performance Evaluation of an Optimal Cache Replacement Policy for Wireless Data Dissemination." IEEE Trans. on Knowledge and Data Engineering, 16(1): 125-139, Jan. 2004.

[10] S. Khanna, V. Liberatore, "On Broadcast Disk Paging," Proc. of the 30th ACM Symp. on the Theory of Computing, pp. 634 - 643, 1998.

[11] H. Schwetman, CSIM User's Guide for Use with CSIM Revision 16, Microelectronics and Computer Technology Corporation, 1992.

[12] D. Aksoy and M. Franklin, "RxW: A Scheduling Approach for Large-ScaleOn-Demand Data Broadcast," IEEE/ACM Transactions on Networking Vol. 7,No. 6, pp. 846-860, 1999.

신 동 천

1985년 2월 서울대학교 공과대학 컴퓨터공학과 졸업(학사). 1987년 2월 한국과학기술원 전산학과 졸업(석사). 1991년 2월 한국과학기술원 전산학과 졸업(박사) 1991년 1월~1993년 2월 한국전산원 선임연구원. 1993년 3월~현재 중앙대학교 산업과학대학 정보시스템학과 교수. 2004년 1월~2005년 1월 Indiana University Bloomington(IUB), School of Informatics 연구교수. 관심분야는 이동 컴퓨팅, 데이타베이스, 데이타 웨어하우스