모바일 컴퓨팅 환경에서 데이터의 중요도에 기반한 캐시 교체와 일관성 유지

정회원 김 삼 근*, 김 형 호*, 안 재 근*

Cache Replacement and Coherence Policies Depending on Data Significance in Mobile Computing Environments

Sam-geun Kim*, Hyung-ho Kim*, Jae-geun Ahn* ***Regular Members***

요 약

최근 모바일 컴퓨팅 환경은 사회 전반으로 급속도로 보편화가 진행되고 있다. 이러한 경향은 모바일 플랫폼 상에서 무선 네트워크를 통한 유선 네트워크 상의 데이터베이스 시스템의 접근 필요성을 부각시킨다. 그러나 모바일 컴퓨팅 환경은 전통적인 컴퓨팅 환경과는 본질적으로 다른 특성으로 인하여 데이터베이스 접근 방식을 그대로 적용하기 어렵다. 이 논문은 새로운 에이전트 기반 모바일 데이터베이스 접근 모델을 제시하고, 데이터 비축을 위한 캐시 교체와 일관성 유지 과정에서 대상 데이터를 선정하는 두 가지 데이터 중요도 함수를 제안한다. 이 함수들은 데이터의 접근 기간, 접근 빈도, 접근 경향, 갱신 빈도, 갱신 경향, 데이터 크기 분포 등을 종합적으로 반영한다. 모의실험에 의하면, 제안하는 함수를 사용한 정책은 LRU 정책, LIX 정책, SAIU 정책에 비하여 접근 지연 시간 감소, 캐시 바이트 적중률 향상, 캐시 바이트 오염률 감소 측면에서 경쟁력이 있음을 보인다.

Key Words : Cache Replacement, Coherence, Mobile, Database

ABSTRACT

Recently, mobile computing environments are becoming rapidly common. This trend emphasizes the necessity of accessing database systems on fixed networks from mobile platforms via wireless networks. However, it is not an appropriate way that applies the database access methods for traditional computing environments to mobile computing environments because of their essential restrictions. This paper suggests a new agent-based mobile database access model and also two functions calculating data significance scores to choose suitable data items for cache replacement and coherence policies. These functions synthetically reflect access term, access frequency and tendency, update frequency and tendency, and data item size distribution. As the result of simulation experiment, our policies outperform LRU, LIX, and SAIU policies in aspects of decrement of access latency, improvement of cache byte hit ratio, and decrease of cache byte pollution ratio.

Ⅰ. 서 론

모바일 컴퓨팅 패러다임은 관련 기술의 진보와 더불어 대중의 보편화가 급속하게 진행되고 있으며, 특히 최근의 유비쿼터스 패러다임과 맞물려 더욱 각광받고 있다. 이러한 경향은 모바일 플랫폼 상의 데이터 관리(data management) 특히, 모바일 클라이언트와 데이터베이스 시스템 간 상호작용의 중요성[1]을 더욱 부각시키고 있다. 그러나 모바일 컴퓨팅 환경은 이동성(mobility), 휴대성(portability), 배터리 수명, 연산

* 한경대학교 컴퓨터공학과(skim@hknu.ac.kr, protcol@gmail.com, ahnjg@hknu.ac.kr)
논문번호 : KICS2010-10-505, 접수일자 : 2010년 10월 21일, 최종논문접수일자 : 2011년 2월 11일

력, 메모리 가용량, 가변 대역폭(variable bandwidth) 등의 태생적 특성과 제약을 가지고 있다. 특히 의도적 또는 비의도적인 가변 대역폭 즉, 약한 연결(weak connectivity)과 네트워크 단절(disconnections)이 빈번하게 발생하는 특성은 정교한 모바일 데이터 관리의 실현을 저해하는 대표적인 요소로 지적된다[1,-4]. 그러므로 오늘날 모바일 데이터 관리의 핵심은 가변 대역폭 특성을 가지는 동적 네트워크(dynamic network) 환경하의 모바일 클라이언트에서 유선 네트워크 상의 데이터베이스 시스템 내의 데이터에 대한 데이터 접근성(data accessibility)과 데이터 가용성(data availability)을 향상시키기는 것이 관건이라고 볼 수 있다.

현재 모바일 컴퓨팅 환경에서 데이터 관리에 관한 연구는 크게 데이터 비축을 위한 모바일 클라이언트 캐싱(mobile client caching)과 데이터 전파를 위한 데이터 브로드캐스트(data broadcast)를 중심으로 진행되고 있다[5].

우선, 모바일 클라이언트 캐싱은 데이터의 일부를 모바일 클라이언트의 메모리 등의 임시 저장소에 한시적으로 복제함으로써, 반복적인 데이터 접근에 대한 성능향상은 물론 약한 연결환경이나 네트워크 단절시 비연결 동작(disconnected operation)을 가능하게 한다. 지금까지 연구된 모바일 클라이언트 캐싱은 주로 전통적인 파일 시스템이나 데이터베이스의 그것과 유사하게 데이터의 접근 최신성(access recency)에 기반[6]하거나, 접근 빈도(access frequency)[7,-9] 또는 갱신 빈도(update frequency)에 기반[10]한다. 그러나 기존 연구들은 데이터의 접근 기간, 접근율과 접근 경향, 갱신율과 갱신 경향, 데이터 크기 분포 등을 다각도로 고려하지 않았다. 그러므로 비교적 정형화된 파일이나 웹 콘텐츠 등의 전달에는 적합하지만 다양한 데이터베이스 질의 문장을 통한 결과 데이터 전달에 그대로 적용하기에는 무리가 있다. 또한, 모바일 클라이언트의 캐싱만을 고려하고 있으므로 보편적으로 3-계층 구조를 갖는 분산시스템 구조에 적합하지 않다. 즉, 클라이언트의 각 질의마다 응용 서버는 질의를 데이터베이스 서버에 요청하고 그 결과를 받은 다음 클라이언트에게 전달함으로써 전체적인 접근 시간의 증가를 초래한다.

한편, 데이터 브로드캐스트는 서버가 일단의 데이터 보고(Data Report, DR)나 캐시무효 보고(Invalidation Report, IR)를 특정 네트워크 세그먼트에 브로드캐스트함으로써 다수의 모바일 클라이언트에게 효과적으로 전달하는 방법이다. 전달방식에 따라 서버가 클라이언트의 관심 데이터의 프로파일(profile)을 이용하여 갱신된 데이터를 주기적 혹은 비주기적으로 일괄 전송하는 push-기반 전달(push- based delivery)[6,11,12], 클라이언트가 관심 데이터를 서버에 요청하면 서버가 브로드캐스트 스케줄링 정책에 따라 전송하는 pull-기반 전달(pull-based, on- demand delivery)[8, 13], 그리고 앞의 두 방식을 혼용한 하이브리드 전달(hybrid delivery)[14,15]로 세분화된다. 특히 하이브리드 전달방식은 모바일 클라이언트가 요청하는 데이터베이스 질의 문장을 동적으로 프로파일링(dynamic profiling)하므로 비교적 정형화된 파일이나 웹 콘텐츠 등의 전달에만 특화된 다른 방식들에 비하여 다양한 데이터베이스 질의 문장을 통한 결과 데이터 전달에 적용하기에 적합하다.

따라서 이 논문에서는 우선 모바일 컴퓨팅 환경에서 데이터베이스 질의 처리를 위한 접근 모델로서 [16]에서 소개된 MCA-FSA(Mobile Client with Agent - Fixed Server with Agent) 모델을 확장하여, 모바일 클라이언트가 요청한 질의 문장을 동적 프로파일링하고, 두 에이전트 모두 질의 결과 데이터를 더블 캐싱(double caching)하는 하이브리드 브로드캐스트 모델을 제안한다. 그리고 캐시 교체(cache replacement)와 일관성 유지(coherence) 정책에 결정적인 요소로서 작용하는 데이터의 접근 기간, 접근 빈도, 접근 경향, 갱신 빈도, 갱신 경향, 데이터 크기 분포 등을 종합적으로 고려한 데이터 중요도 산출함수를 제안한다.

이 논문의 나머지 구성은 다음과 같다. 2장에서는 모바일 컴퓨팅 환경을 위한 데이터 캐싱에 대한 관련 연구를 기술한다. 3장에서는 에이전트 기반 모바일 데이터베이스 접근 모델을 소개하고, 두 에이전트의 주요 기능들을 설명한다. 4장에서는 데이터 캐싱을 위한 두 가지 데이터 중요도 산출함수를 정의하고, 산출된 중요도에 기반한 캐시 교체와 일관성 유지 알고리즘을 기술한다. 5장에서는 실험을 통한 성능평가 결과를 보이고, 끝으로 6장에서는 결론을 맺는다.

Ⅱ. 관련 연구

Acharya 등은 모바일 클라이언트에서 데이터 캐시 교체를 위한 정책으로써 PIX 정책과 LIX 정책을 제안하였다[6]. PIX 정책은 데이터 i에 대한 중요도를 산출하기 위하여 식 (1)을 사용하여, 데이터 중요도가 낮은 데이터를 캐시 교체 과정에서 캐시 해제될 데이터로서 선택한다.

$$p_i/x_i \quad (1)$$

여기서 p_i는 데이터 i의 접근 확률(access probability)이며, x_i는 브로드캐스트 빈도(broadcast frequency)이다. 즉, PIX 정책에서 데이터 중요도가 낮은 데이터는 접근 확률이 낮거나, 데이터 수신이 빈번한 것이다. 모바일 클라이언트는 현실적으로 전체 데이터 집합에서 특정 데이터 i에 대한 접근 확률을 사전에 알 수 없으므로, PIX 정책은 구현 가능한 정책이 아니다. LIX 정책은 LRU (Least Recently Used) 정책을 변형한 것으로서 식 (1)의 접근 확률 p_i를 식 (2)로 변경한 것이다.

$$p_i = \Lambda/(CurrentTime - t_i) + (1-\Lambda)p_i \qquad (2)$$

여기서 t_i는 데이터 i에 대한 최종 접근일시이고, Λ는 0과 1사이의 값을 가지는 가중치 조정을 위한 제어 매개변수이다. [6]에 의하면, LIX 정책의 성능은 PIX 정책의 성능에 비교적 근접한다. 그러나 LIX 정책은 데이터의 접근 최신성은 고려하지만, 데이터의 갱신 특성 및 크기 특성은 고려하지 않는다.

Xu 등은 데이터 중요도 산출을 위하여 접근 확률(A_i), 갱신 빈도(U_i), 접근 지연 시간(L_i, retrieval delay), 데이터 크기(S_i) 등 네 가지 요소를 고려하여 식 (3)과 같이 SAIU 정책을 제안하였다[8].

$$gain(i) = \frac{L_i A_i}{S_i U_i} \qquad (3)$$

여기서 A_i, U_i, L_i는 각각 식 (4), 식 (5), 식 (6)과 같다.

$$A_i = a_a/(t^c - t_i^{la}) + (1-a_a)A_i \qquad (4)$$

$$U_i = a_u/(t^c - t_i^{lu}) + (1-a_u)U_i \qquad (5)$$

$$L_i = a_s/(t^c - t_i^{qt}) + (1-a_s)L_i \qquad (6)$$

여기서 t^c는 현재일시이고, t_i^{la}, t_i^{lu}, t_i^{qt}는 각각 데이터 i에 대한 최종 접근일시, 최종 갱신일시, 질의 요청일시(query time)이며, a_a, a_u, a_s는 각각 A_i, U_i, L_i에 대한 제어 매개변수이다. SAIU 정책은 종래의 고정 크기의 데이터만 고려하던 정책과 달리 가변 크기의 데이터와 데이터 크기 별 전송지연에 따른 서비스 시간(service time), 갱신 빈도, 접근 최신성 등의 특성을 고려한다. 그러나 데이터 크기를 단순하게 선형적으로 고려하므로, 접근하는 데이터의 크기 분포를 고려하여 적절한 크기를 가지는 데이터를 효과적으로 반영하지 못한다. 또한, 데이터의 최근의 접근 경향 및 갱신 경향을 고려하지 않는다.

Ⅲ. 모바일 데이터베이스 접근 모델

Venkatraman[16]은 보편적인 모바일 컴퓨팅 환경을 전통적인 클라이언트-서버 구조인 MC-FS(Mobile Client-Fixed Server) 모델, 유선 네트워크에만 에이전트가 위치하는 MC-FA-FS(Mobile Client-Fixed Agent-Fixed Server) 모델, 유무선 네트워크 모두 에이전트가 위치하는 MCA-FSA(Mobile Client with Agent-Fixed Server with Agent) 모델, 중앙 서버가 없이 모바일 클라이언트간 상호관계를 가지는 P-P (Peer-Peer) 모델, 그리고 무선 네트워크에만 모바일 클라이언트와 독립된 에이전트가 위치하는 MC-MA-FS (Mobile Client-Mobile Agent-Fixed Server) 모델 등 다섯 가지로 구분하였다. 이 모델들은 각각의 고유한 특징과 장단점을 가지고 있으며, 주요 항목을 비교하면 표 1과 같다.

MCA-FSA 모델은 서버 프록시(server proxy)인 클라이언트 측 에이전트와 클라이언트 프록시(client proxy)인 서버 측 에이전트를 통한 데이터 최적화, 가용성 향상 등을 꾀할 수 있다. 특히, MCA-FSA 모델은 네트워크 단절이나 약한 연결 등의 모바일 컴퓨팅 환경에서 데이터 관리를 위한 목적으로써 비교적 적합한 것을 알 수 있다.

MCA-MA-FS 모델은 일련의 수행 계획(itinerary)에 따라 각 모바일 클라이언트에 전송되어 필요한 자원에 접근하거나 서비스를 제공하는 전송 프로그램(transportable program)인 에이전트가 무선 네트워크에 위치하는 구조이다. 그러므로 MCA-MA-FS 모델은 갱신이 거의 없는 웹 콘텐츠의 캐싱이나 읽기전용의 네트워크 파일 캐싱 측면에서는 다른 모델에 비하

표 1. 모바일 컴퓨팅 모델 간 특성비교

특성 / 모델	네트워크 단절	약한 연결 적합성	계산량	소비 전력
MC-FS	미지원	낮음	중간	높음
MC-FA-FS	미지원	중간	적음	낮음
MCA-FSA	지원	높음	중간	중간
P-P	지원	중간	많음	높음
MC-MA-FS	지원	높음	적음	낮음

여 전반적으로 우수하지만, 갱신이 빈번하게 발생할 수 있는 데이터베이스 질의 결과의 캐싱에 적용하는 것은 부적합하다. 즉, MCA-MA-FS 모델은 보편적으로 유선 네트워크에 비하여 저속인 무선 네트워크 상에만 에이전트가 위치하는 한계로 인하여 캐시 일관성 유지를 위하여 중요한 캐시 데이터를 주기적으로 데이터베이스 서버에 질의하고 변경된 데이터를 단시간에 모바일 클라이언트에 반영하는 데이터 브로드캐스트 수행이 어렵다.

따라서 이 논문에서는 모바일 클라이언트에서 유선 네트워크 상의 데이터베이스 서버에 접근하는 모델로서 MCA-FSA 모델을 기반으로 하고, 기존 연구와 달리 두 에이전트 모두 캐시를 관리하는 더블 캐싱 메커니즘을 지원하도록 확장한 모델을 제안한다. 제안하는 구조를 개괄적으로 도시하면 그림 1과 같다. 이 구조에서 클라이언트 측 에이전트와 서버 측 에이전트는 각각 모바일 에이전트(mobile agent)와 데이터 에이전트(data agent)이다.

3.1 모바일 에이전트

모바일 에이전트는 데이터 에이전트 간 데이터 요청과 수신, 그리고 캐싱을 주기능으로 한다. 데이터 요청은 업링크 채널(uplink channel)을 통하여 비주기적으로(on-demand) 이루어진다. 업링크 채널은 인증, 질의 문장 등을 데이터 에이전트로 전송하기 위한 상향 송신 채널이며, 모바일 네트워크 환경에 따라 하향 수신 채널에 비하여 비대칭적으로 저속일 수 있다. 데이터 요청의 결과는 데이터 에이전트가 질의 처리를 수행한 후 브로드캐스트하는 푸시 이벤트(push event)에 의하여 비동기적으로 수신하게 된다. 이때 모바일 에이전트는 수신하는 데이터를 4.2절의 캐시 교체 정책에 따라 자체 캐시 블록(cache blocks)에 할당하여 데이터를 비축한다. 이 과정을 통하여 모바일 에이전트는 중요한 데이터를 최적으로 관리하면서, 약한 연결이나 네트워크 단절을 대비하고 데이터 가용성과 처리성능을 향상시킨다.

3.2 데이터 에이전트

데이터 에이전트는 모바일 에이전트의 데이터 요청에 대한 질의 처리와 브로드캐스트, 그리고 캐싱을 주기능으로 한다.

수신한 모바일 에이전트의 데이터 요청은 먼저 요청 대기행렬(on-demand queue)에 적재되고, FIFO (First In, First Out) 정책에 따라 질의 처리나 캐시검색을 통하여 처리된다. 이후, 브로드캐스트 스케줄링(broadcast scheduling)에 의하여 데이터를 전송한다. 처리했던 데이터는 모바일 에이전트와 같은 캐시 교체 정책에 따라 비축된다. 그리고 4.2절의 캐시 일관성 유지 정책에 따라 중요도가 높은 데이터를 중심으로 정기적으로 데이터베이스 시스템의 데이터와 비교한다. 갱신된 데이터는 캐시 블록의 데이터를 먼저 동기화하고, 전파 정책(dissemination policy)에 따라 갱신된 데이터 자체를 전송하는 데이터 보고(DR) 또는 데이터의 갱신정보만 전송하는 캐시무효 보고(IR) 중 택일해서 전파한다. 모바일 에이전트는 데이터 보고를 수신하면 캐시 블록의 데이터를 동기화하고, 캐시무효 보고를 수신하면 캐시 블록에서 데이터를 해제한다.

데이터 에이전트의 캐싱은 모바일 클라이언트의 데이터 요청에 대한 고속처리와 단기간의 대량 브로드캐스트에 주 목적이 있고, 모바일 클라이언트에 비하여 우수한 하드웨어 성능에 의하여 상대적으로 많은 캐시 블록을 관리하는 것이 특징이다.

Ⅳ. 데이터 중요도에 기반한 캐시 교체와 일관성 유지

데이터의 중요도는 특정 데이터에 대한 접근 기간,

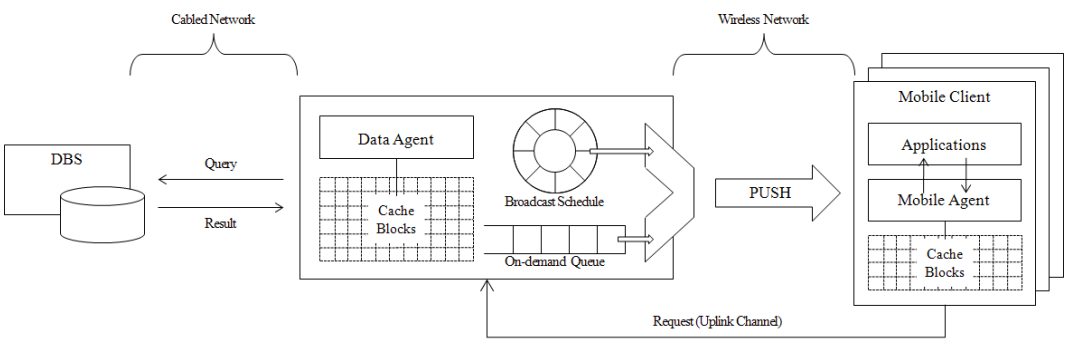


그림 1. 에이전트 기반 데이터베이스 접근 모델

접근 빈도와 경향, 갱신 빈도와 경향, 데이터의 크기 분포, 그리고 클라이언트 특성을 반영한 것으로서, 데이터 캐시 교체와 일관성 유지 처리과정에서 중요한 척도가 된다. 예를 들면, 갱신이 거의 없고 접근이 빈번하여 높은 데이터 중요도를 가지는 데이터는 그렇지 않은 데이터에 비하여 캐시 교체 과정에서 낮은 교체 우선순위를 가지고, 캐시 일관성 유지 처리과정에서 높은 우선순위를 가지게 된다[5-10].

그림 1에서 나타낸 모바일 에이전트는 캐시 교체를 위한 목적으로써 데이터 중요도를 이용하고, 데이터 에이전트는 캐시 교체와 일관성 유지를 위한 목적으로써 데이터 중요도를 이용한다. 즉 4.1절에서 기술된 데이터 중요도 산출함수에 의하여 산출된 데이터 중요도는 4.2절에서 기술하는 캐시 교체와 일관성 유지 알고리즘에서 데이터 우선순위의 판단기준이 된다. 데이터 중요도 산출함수는 기존 관련연구와 직관적 기준에 의하여 도출된 다음의 네 가지 기준에 근거하여 유도된 것이다.

첫째, 특정 데이터에 대한 마지막 접근 시간과 현재 시간의 차가 크다면 데이터 중요도는 낮다[6]. 그러나 접근 기간(access term)이 길다면 데이터 중요도는 높게 조정할 필요가 있다.

둘째, 특정 데이터에 대한 접근 빈도가 높다면 데이터 중요도는 높다[8,9]. 그리고 최근의 접근 빈도가 증가하고 있다면 데이터 중요도는 더욱 높게, 감소하고 있다면 데이터 중요도는 좀더 낮게 조정할 필요가 있다.

셋째, 특정 데이터에 대한 갱신 빈도가 높다면 데이터 중요도는 낮다[8-10]. 그리고 최근의 갱신 빈도가 증가하고 있다면 데이터 중요도는 더욱 낮게, 감소하고 있다면 데이터 중요도는 좀더 높게 조정할 필요가 있다.

넷째, 이전에 접근하였던 데이터들의 크기 분포를 고려하여 특정 데이터의 크기가 아주 작거나 크지 않으면 데이터 중요도는 높다. 데이터의 크기가 아주 작다면 재전송 시간이 짧으므로 접근 비용이 적기 때문이고, 데이터의 크기가 아주 크다면 캐싱 비용이 증가하고 단위 시간당 브로드캐스트되는 데이터 수가 감소하기 때문이다.

4.1 데이터 중요도 산출

이 절에서는 특정 데이터의 캐시 교체 과정에서 캐시 해제 대상 데이터를 선정하거나 캐시 일관성 유지를 시도하는 과정에서 대상 데이터를 선정하는 척도인 데이터 중요도를 산출하는 두 가지 함수를 정의한다. 이 절에서 사용하는 주요한 기호와 설명은 표 2와 같다.

앞서 기술한 첫째 기준에 따라서, 데이터 i에 대한 관심도 점수 T_i는 $n_i^a > 1$ 일 때 식 (7)과 같고, $n_i^a = 1$ 일 때 값은 1이다. 식 (7)은 [6]에서 제안한 식 (2)에서 데이터 i의 접근 기간을 고려하도록 수정한 것이다.

$$T_i = \frac{t_{i,last}^a - t_{i,first}^a}{t_c - {}_{i,last}^a} \tag{7}$$

둘째 기준에 따라서, 데이터 i에 대한 접근도 점수 A_i는 $n_i^a \geq 3$ 일 때 식 (8)과 같고, $n_i^a = 2$ 와 $n_i^a = 1$ 일 때 값은 각각 1.25와 1이다. 식 (8)은 데이터 i의 접근 빈도 이외 최근의 접근 경향도를 반영한 것이다.

$$A_i = n_i^a \omega_i^a \tag{8}$$

여기서 ω_i^a는 데이터 i에 대한 최근 m개의 단위 시간당 접근 빈도 변화율을 기하평균한 접근 경향도로서 식 (9)와 같다. 사용자는 1 이상의 정수 m을 정의함으로써 접근 경향도 산출에 관여할 수 있다.

$$\omega_i^a = \left[\prod_{j=t_{i,last-m}^a}^{t_{i,last}^a} \left(1 + \frac{\lambda_{i,j}^a - \lambda_{i,j-1}^a}{\lambda_{i,j}^a} \right) \right]^{\frac{1}{m}} \tag{9}$$

이때 $\lambda_{i,j}^a = \dfrac{n_{i,last-j}^a - n_{i,last-j-1}^a}{t_{i,last-j}^a - t_{i,last-j-1}^a}$ 이다.

셋째 기준에 따라서, 데이터 i에 대한 갱신도 점수 U_i는 $n_i^u \geq 3$ 일 때 식 (10)과 같고, $n_i^u = 2$, $n_i^u = 1$, $n_i^u = 0$ 일 때 값은 각각 1.5, 1.25, 1이다. 식 (10)은 데이터 i의 갱신 빈도 이외 최근의 갱신 경향도를 반영한 것이다.

$$U_i = n_i^u \omega_i^u \tag{10}$$

여기서 ω_i^u는 데이터 i에 대한 최근 n개의 단위 시간당 갱신 빈도 변화율을 기하평균한 갱신 경향도로서 식 (11)과 같다. 사용자는 1 이상의 정수 n을 정의함으로써 갱신 경향도 산출에 관여할 수 있다.

$$\omega_i^u = \left[\prod_{j=t_{i,last-n}^u}^{t_{i,last}^u} \left(1 + \frac{\lambda_{i,j}^u - \lambda_{i,j-1}^u}{\lambda_{i,j}^u} \right) \right]^{\frac{1}{n}} \tag{11}$$

이때 $\lambda_{i,j}^{u} = \frac{n_{i,last-j}^{u} - n_{i,last-j-1}^{u}}{t_{i,last-j}^{u} - t_{i,last-j-1}^{u}}$ 이다.

넷째 기준에 따라서, 접근하였던 모든 데이터 크기 분포에 따른 데이터 i의 크기위상 점수 Z_i는 식 (12)와 같다.

$$Z_i = \phi_i^e e^{-\phi_i} + 1 \tag{12}$$

이때 $10^e e^{-10} \approx 0$ 이다.

여기서 ϕ_i는 접근하였던 모든 데이터 크기 분포상 데이터 i의 크기의 위상이며, 0과 10 사이의 값을 가진다. 접근하였던 모든 데이터 중 최대 크기값과 최소 크기값을 각각 $z_{\max}$와 $z_{\min}$이라고 하면, ϕ_i는 식 (13)과 같다.

$$\phi = \begin{cases} 0 & , \text{if } z_{\min} = z_i \\ \dfrac{10(z_i - z_{\min})}{z_{\max} - z_i} & , \text{if } z_{\min} \leqq z_i \leqq z_{\max} \\ 10 & , \text{if } z_{\max} = z_i \end{cases} \tag{13}$$

식 (12)에서, $\phi_i = e$, $\phi_i = 0$ 일 때 Z_i는 각각 최대값 2와 최소값 1을 가지며, $\phi_i = 10$일 때 Z_i는 최소값 1에 근사한다. 즉, 식 (12)는 접근하였던 데이터 가운데 크기 위상이 e인 데이터를 중심으로 데이터 중요도를 높게 반영하도록 고안된 것이다. 이 논문에서 e는 자연상수를 의미하며, 사용자는 1과 10 사이의 실수를 e로 정의함으로써 크기위상 점수 산출에 관여할 수 있다.

표 2. 기호 표기법

기호	의미
D	데이터베이스의 모든 데이터 집합
i	데이터베이스의 특정 데이터 i, $i \in D$
t_c	현재일시
$t_{i,k}^{a}$	데이터 i를 k번째 접근한 일시, k = *first* ⇒ 접근한 최초일시, k = *last* ⇒ 접근한 최종일시
$t_{i,k}^{u}$	데이터 i를 k번째 갱신한 일시, k = *first* ⇒ 갱신한 최초일시, k = *last* ⇒ 갱신한 최종일시
n_i^a	데이터 i를 접근한 총 횟수
$n_{i,k}^{a}$	$t_{i,first}^{a}$부터 $t_{i,k}^{a}$까지의 기간 동안 데이터 i를 접근한 횟수
n_i^u	데이터 i를 갱신한 총 횟수
$n_{i,k}^{u}$	$t_{i,first}^{a}$부터 $t_{i,k}^{a}$까지의 기간 동안 데이터 i를 갱신한 횟수
z_i	데이터 i의 크기

위와 같을 때, 캐시 교체와 일관성 유지를 위한 데이터 i의 중요도 산출함수 $\zeta(i)$와 $\check{\zeta}(i)$는 각각 식 (14)와 식 (15)와 같이 정의한다.

$$\zeta(i) = \Lambda_1 \Phi + (1 - \Lambda_1) \frac{T_i A_i Z_i}{U_i} \tag{14}$$

$$\check{\zeta}(i) = \Lambda_2 \Phi + (1 - \Lambda_2)\, T_i A_i Z_i U_i \tag{15}$$

이때 $\Phi = \frac{n_i^a}{t_c - t_{i,last}^{a}}$ 이다.

여기서 Λ_1과 Λ_2는 제어 매개변수로서 0과 1사이의 값을 가진다. Φ는 에이징(ageing) 기법으로 널리 알려진 가중치 조정을 위한 변수[6]를 변형한 것이다. 사용자는 제어 매개변수를 조정함으로써 두 산출함수의 모멘텀(momentum)을 적용 환경에 보다 유연하게 반영할 수 있다.

4.2 캐시 교체와 일관성 유지 알고리즘

이 절에서는 데이터 캐싱을 위한 캐시 교체 및 캐시 일관성 유지 알고리즘을 기술한다. 캐시 교체는 두 에이전트에서 사용하는 가용 캐시 블록이 부족한 상황에서 캐시 할당을 하려는 임의의 데이터가 주어질 때 처리하는 일련의 절차이다. 캐시 일관성 유지는 데이터 에이전트에서 중요도가 높은 캐시 데이터를 중심으로 데이터베이스 서버 측 데이터와 비교하는 일련의 절차이다. 에이전트는 애플리케이션에서 접근하는 데이터를 캐시 블록에 관리하는 과정에서 4.1절의 식 (14)에 의하여 산출된 데이터 중요도에 따라서 캐시 교체를 수행한다.

그림 2는 캐시 할당을 위하여 임의의 데이터 i가 주어졌을 때의 캐시 교체 알고리즘을 보여준다. 현재 가용 캐시 블록의 여유가 없으면(11행), 중요도가 가장 낮은 캐시 데이터부터 여러 개를 캐시 해제하여 데이터 i의 크기이상의 가용 캐시 블록을 마련한다(14-19행). 이후, 데이터 i를 캐시 할당한다(22행). 데이터 에이전트는 모바일 에이전트에서 접근하는 데이터를 캐시 블록에 관리하는 과정에서 4.1절의 식 (15)에 의하여 산출된 데이터 중요도에 따라서 캐시 일관성 유지를 수행한다.

그림 2. 캐시 교체 알고리즘

```
1.  procedure CacheReplacement(d_i)
2.  { C is an array having n cached data items; c_1, ···, c_n.}
3.  { d_i is a data item to be stored in cache blocks. }
4.  Len := GetLength(d_i) : numeric
5.  begin
6.    if GetLength(TotalCacheBlocks) < Len then
7.      begin
8.        GetFailedMessage()
9.        Exit
10.     end {if}
11.   if GetLength(AvailCacheBlocks) < Len then
12.     begin
13.       AscendSortBySignificance(C)
14.       for k := 1 to n do
15.         begin
16.           CacheUnregistration(c_k)
17.           if GetLength(AvailCacheBlocks) ≥ Len then
18.             break
19.         end {for}
20.     end {if}
21.   CacheRegistration(d_i)
22. end
```

그림 3. 캐시 일관성 유지 알고리즘

```
1.  procedure CacheCoherence(Quantity, OpMode)
2.  { Quantity is the number of cached data items to be
processed. }
3.  { OpMode is the flag IR or DR }
4.  Queue : array, global
5.  begin
6.    if n < Quantity then
7.      Quantity := n
8.    DescendSortBySignificance(C)
9.    for k := 1 to Quantity do
10.     begin
11.       if ChangedData = Synchronize(c_k) then
12.         begin
13.           ReadjustSignificance(c_k)
14.           if IR = OpMode then
15.             Append(Queue, InvalidReport(c_k))
16.           else
17.             Append(Queue, DataReport(c_k))
18.         end {if}
19.     end {for}
20. end
```

그림 3은 캐시 일관성 유지 알고리즘을 보여준다. 캐시 일관성 유지는 중요도가 가장 높은 캐시 데이터부터 *Quantity*건을 대상으로 한다(9-19행). 해당 캐시 데이터는 먼저 데이터 동기화를 시도하여 데이터베이스 서버 측 데이터와 일치시킨다. 이때 캐시 데이터가 변경되어서 동기화된다면(11행), 데이터 중요도를 다시 계산한다(13행). 그리고 *OpMode*에 따라 모바일 클라이언트 집단에게 캐시무효 보고(IR)나 갱신된 데이터 보고(DR)를 하기 위하여 브로드캐스트 스케줄링 대기행렬에 관련 정보를 추가한다(각각 15행과 17행). 모바일 에이전트는 캐시무효 보고를 수신하면 대상 캐시 데이터를 캐시 블록에서 해제하고, 데이터 보고를 수신하면 캐시 데이터를 캐시 블록에 재할당하고 데이터 중요도를 다시 계산한다.

V. 실험 및 평가

이 장에서는 제안하는 데이터베이스 접근 모델과 데이터 중요도 산출함수에 의한 캐시 교체와 일관성 유지 성능을 평가하기 위한 모의실험 환경과 그 결과를 기술한다.

5.1 모의실험 환경

3장에서 제안한 데이터베이스 접근 모델의 구성요소인 데이터베이스 시스템, 데이터 에이전트, 모바일 에이전트의 기본환경은 각각 표 3, 표 4, 표 5와 같다.

데이터베이스 내의 개별 데이터의 크기는 *MinItemSize*와 *MaxItemSize* 사이의 랜덤 분포(random distri-

표 3. 데이터베이스 시스템 환경

매개변수	값	의미
NumItems	2000	데이터 개수(개)
MinItemSize	5	최소 데이터 크기(KB)
MaxItemSize	500	최대 데이터 크기(KB)
U-Cold	80	갱신패턴 U-Cold(%)
U-Hot	20	갱신패턴 U-Hot(%)
U-Interval	100	갱신간격(초)

표 4. 데이터 에이전트 환경

매개변수	값	의미
DaCachePcnt	10	캐시 블록 크기 요율(%)
DaCacheM	10	식 (9)의 m
DaCacheN	10	식 (11)의 n
DaC-OpMode	IR	캐시 일관성 유지 유형
C-Interval	50	캐시 일관성 유지 간격(초)
BandwidthDbs	100	DA-DBS 간 대역폭(Mbps)
BandwidthMa	10	DA-MA 간 대역폭(Mbps)

표 5. 모바일 에이전트 환경

매개변수	값	의미
NumClients	50	클라이언트 개수(노드)
MaCachePcnt	10	캐시 블록 크기 요율(%)
MaCacheM	5	식 (9)의 m
MaCacheN	5	식 (11)의 n
θ	0.95	Zipf 분포 변수
RgnSize	20	접근 데이터 개수(개)
A-Interval	5	데이터 접근간격(초)
NumCycles	10	총 사이클 횟수(회)
NumWCycles	1	Warm-up 사이클 횟수(회)
IdleTime	30	사이클 간 휴지 시간(초)

bution)를 따른다. 갱신패턴은 Xu 등이 소개한 U-Cold/U-Hot 갱신패턴[8]을 사용한다. 즉, 개별 클라이언트가 접근하는 데이터 중 접근도가 높은 U-Hot%의 데이터를 U-Cold% 확률로 갱신하고, 나머지 U-Cold%의 데이터를 U-Hot% 확률로 갱신한다.

데이터 에이전트는 *C-Interval*초 간격으로 캐시된 데이터 가운데 중요도가 높은 순으로 캐시 일관성 유지를 시도하며, 대상 데이터는 최대 250개로 제한하였다. 데이터 에이전트의 캐시 블록 크기는 (*MinItemSize* + *MaxItemSize* - 1) / 2 * *RgnSize* * *NumClients* * *DaCachePcnt* 로써 계산되며, 모바일 에이전트의 경우 이 식에서 *NumClients* = 1, *DaCachePcnt* = *MaCachePcnt*이다. 이 모의실험에서는 그림 1의 애플리케이션의 데이터 요청을 모바일 에이전트가 직접 처리하도록 구성하였다. 즉, 애플리케이션과 모바일 에이전트 간 데이터 접근 지연 시간(access latency time)은 고려하지 않는다. 그리고 하나의 호(cell)를 가지는 단일 네트워크 세그먼트를 고려한다. 데이터 에이전트는 모바일 에이전트가 요청한 데이터를 캐시 블록에서 검색하거나 데이터베이스 시스템에 질의한 후 FIFO 정책에 따라 네트워크 세그먼트에 브로드캐스트하고, 정기적으로 캐시 일관성 유지를 수행한 후 캐시무효 보고(IR)를 전파하는 브로드캐스트하는 두 개의 채널을 가진다. 캐시무효 보고를 위한 전송량은 데이터 당 바이트수로 대역폭에 비하여 무시할 수준이므로 전파에 따른 전송지연은 없는 것으로 간주한다.

각 모바일 에이전트는 개별적으로 전체 데이터베이스 *NumItems*개 데이터 중 *RgnSize*개 관심 데이터를 랜덤 분포로써 선정한다. 그리고 단위 사이클 당 관심 데이터에 대하여 θ 매개변수를 가지는 *Zipf* 분포를 따르는 접근패턴과 랜덤 분포를 따르는 순서에 따라 *A-Interval*초 간격으로 접근한다. 데이터 접근은 단위 사이클 당 100회로 이루어져 있으며, 총 *NumCycles* 회의 사이클을 수행한다. 여기서 처음 *NumWCycles*회의 사이클은 성능평가 과정에서 무시된다. 데이터 접근과정에서 발생하는 접근 지연 시간은 데이터 에이전트와 데이터베이스 시스템의 대기행렬에 의한 대기 시간(queuing time) 및 각 대역폭을 고려하여 산정된다.

한편, 데이터 중요도 산출함수 식 (14)와 식 (15)의 제어 매개변수의 값은 $\Lambda_1 = \Lambda_2 = 0.25$ 이다.

5.2 모의실험 결과

모의실험의 결과는 기본환경을 제외한 데이터베이스 내 데이터 크기 분포, 데이터 값, 모바일 에이전트의 접근패턴 등을 달리하여 5회씩 수행한 후 산술평균하였다. 그리고 동일한 환경과 조건을 전통적인 LRU 정책, Acharya 등이 제안한 LIX 정책[6], Xu 등이 제안한 SAIU 정책[8]에 적용하여 모바일 에이전트의 캐시 교체와 일관성 유지 성능을 비교하였다. 캐시 교체 성능평가의 비교척도는 접근 지연 시간과 캐시 바이트 적중률(Byte Hit Ratio)을 대상으로 한다. 캐시 일관성 유지 성능평가의 비교척도는 캐시 바이트 오염률(Byte Pollution Ratio)을 대상으로 한다.

그림 4와 그림 5는 각각 단일 클라이언트 환경에서 모바일 에이전트의 접근 지연 시간과 캐시 바이트 적중률을 비교한 결과이다. 이 실험은 5.1절에서 기술한 기본환경에서 *BandwidthDbs* = 10, *BandwidthMa* = 1, *DaCachePcnt* = 0, *NumClients* = 1, A-Interval = 1 로 조정하고 *MaCachePcnt*를 10에서 50으로 증가하면서 수행하였다. 접근 지연 시간은 *MaCachePcnt* = 0 의 결과를 기준으로 백분위 환산한 결과이다.

실험에 의하면, 제안하는 데이터 중요도 산출함수에 따른 접근 지연 시간은 평균적으로 LRU 정책 대

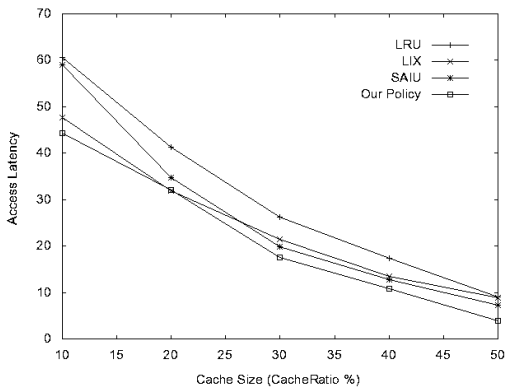


그림 4. 접근 지연 시간 비교 (단일 모바일 클라이언트 환경)

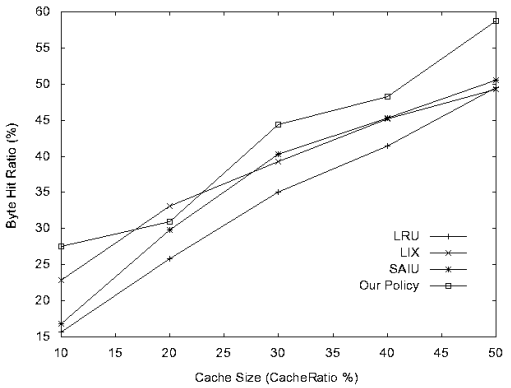


그림 5. 바이트 적중률 비교 (단일 모바일 클라이언트 환경)

비 9.16%, LIX 정책 대비 2.99%, SAIU 정책 대비 5.06% 감소함을 보이며, 다양한 캐시 크기 환경에서 고르게 개선하는 것으로 나타난다. 캐시 바이트 적중률은 평균적으로 LRU 정책 대비 8.49%, LIX 정책 대비 4.06%, SAIU 정책대비 5.45% 향상됨을 보인다. 특히 캐시 크기가 작은 환경에서 캐시 바이트 적중률이 상대적으로 높은 결과는 제한된 메모리와 저장소를 가지는 모바일 컴퓨팅 환경에서 다른 정책에 비하여 경쟁력이 있음을 보인다.

그림 6은 다수의 모바일 클라이언트 집단 환경에서 접근 지연 시간을 비교한 결과이다. 그림 7과 그림 8은 각각 오염된 캐시를 포함한 캐시 바이트 적중률과 오염된 캐시를 제외한 캐시 바이트 적중률을 비교한 것이다. 그림 9는 캐시 바이트 오염률을 비교한 결과이다. 이 실험은 5.1절에서 기술한 실험환경에서 *DaCachePcnt*와 *MaCachePcnt*를 모두 10에서 50으로 증가하면서 수행하였다. 접근 지연 시간은 *MaCachePcnt* = 0, *DaCachePcnt* = 0의 결과를 기준

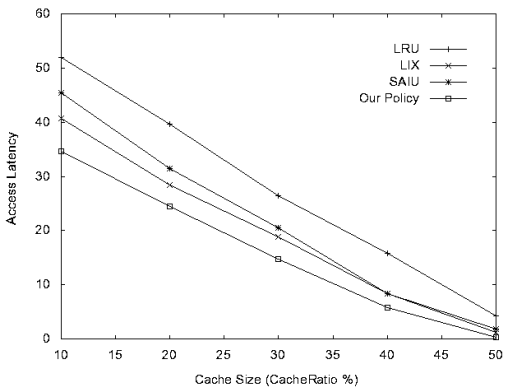


그림 6. 접근 지연 시간 비교 (다수 모바일 클라이언트 환경)

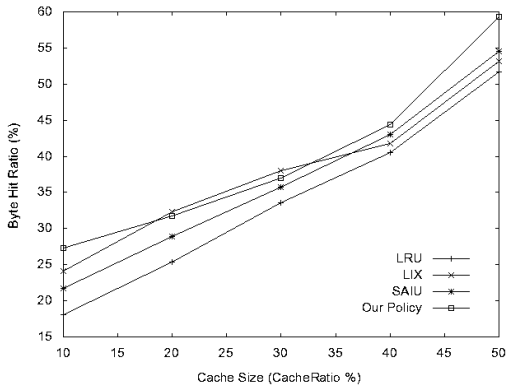


그림 7. 바이트 적중률 비교 (다수 모바일 클라이언트 환경, 오염된 캐시 포함)

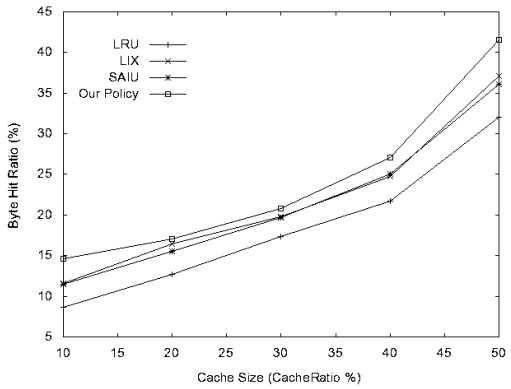


그림 8. 바이트 적중률 비교 (다수 모바일 클라이언트 환경, 오염된 캐시 제외)

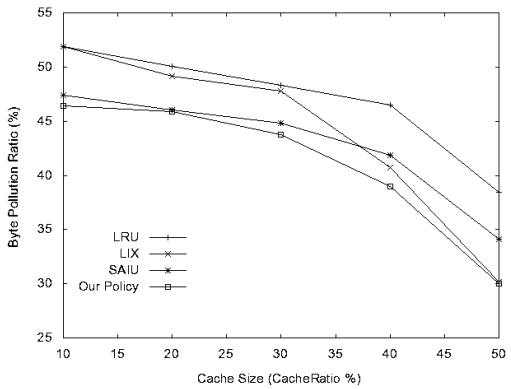


그림 9. 바이트 오염률 비교

으로 백분위 환산한 결과이다.

실험에 의하면, 제안하는 데이터 중요도 산출함수에 따른 접근 지연 시간은 평균적으로 LRU 정책 대비 11.63%, LIX 정책 대비 3.50%, SAIU 정책 대비

5.50% 감소함을 보인다. 오염된 캐시를 포함한 캐시 바이트 적중률은 평균적으로 LRU 정책 대비 6.10%, LIX 정책 대비 2.05%, SAIU 정책대비 3.16% 향상됨을 보이며, 오염된 캐시를 제외한 캐시 바이트 적중률은 평균적으로 LRU 정책 대비 5.75%, LIX 정책 대비 2.27%, SAIU 정책대비 2.66% 향상됨을 보인다. 캐시 바이트 오염률은 평균적으로 LRU 정책 대비 5.97%, LIX 정책 대비 2.88%, SAIU 정책대비 1.76% 감소됨을 보인다.

Ⅵ. 결 론

이 논문에서는 모바일 컴퓨팅 환경의 에이전트 기반 데이터베이스 접근 모델을 소개하고, 각 에이전트의 캐싱 정책 즉, 캐시 교체와 일관성 유지 과정에서 대상 데이터를 선정하기 위한 두 가지 데이터 중요도 산출함수를 제안하였다. 데이터 중요도 산출함수는 데이터의 접근 기간, 접근 빈도와 접근 경향, 갱신 빈도와 갱신 경향, 데이터 크기 분포 등을 종합적으로 고려한 것이다.

모의실험에 의하면, LRU 정책, LIX 정책, SAIU 정책 등 기존의 방식들과 비교하였을 때 높은 접근 지연 시간 감소와 낮은 캐시 바이트 오염도를 보인다. 특히 캐시 크기가 작은 환경에서 여타 정책에 비하여 높은 캐시 바이트 적중률을 보이는 특성은 제한된 하드웨어 사양을 가지는 모바일 컴퓨팅 환경에 보다 적합하다고 볼 수 있다.

한편, 데이터 브로드캐스트 과정은 네트워크 세그먼트 수만큼 수행하므로, 전체 클라이언트 수 대비 많은 클라이언트 수를 갖는 네트워크 세그먼트는 높은 우선순위를 부여받을 필요가 있다. 그러므로 이 논문에서 제안한 데이터 중요도 산출함수에 이러한 요소를 반영하여 좀더 보완한다면 다수의 네트워크 세그먼트를 가지는 모바일 네트워크 환경에서 전파 효율성을 더욱 향상시킬 수 있을 것이다. 이 부분은 향후 연구 주제로서 남긴다.

참 고 문 헌

〔1〕 D. Barbara, "Mobile Computing and Databases - A Survey," *IEEE Transactions on Knowledge and Data Engineering*, Vol.11, No.1, pp.108-117, 1999.

〔2〕 S. K. Madria, M. Mohania, S. S. Bhowmick, and B. Bhargava, "Mobile Data and Transaction Management," *Information Sciences*, Vol.141, No.3/4, pp.279-309, 2002.

〔3〕 M. H. Dunham and A. Helal, "Mobile Computing and Databases: Anything New?," *ACM SIGMOD Record*, Vol.24, No.4, pp.5-9, 1995.

〔4〕 R. Tewari and P. Grillo, "Data Management for Mobile Computing on the Internet," *ACM Annual Computer Science Conference*, pp.246-252, 1995.

〔5〕 K. C. Lee, W-C. Lee, and S. K. Madria, "Pervasive Data Access in Wireless and Mobile Computing Environments," *Wireless Communications and Mobile Computing*, Vol.8, No.1, 2008.

〔6〕 S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication Environments," *ACM SIGMOD Record*, Vol.24, No.2, pp.199-210, 1995.

〔7〕 S. Khanna and V. Liberatore, "On Broadcast Disk Paging," *SIAM Journal on Computing*, Vol.29, No.5, pp.1683-1702, 2000.

〔8〕 J. Xu, Q. Hu, D. L. Lee, and W-C. Lee, "SAIU: An Efficient Cache Replacement Policy for Wireless On-demand Broadcasts," *In Proceedings of the Ninth International Conference on Information and Knowledge Management*, pp.46-53, McLean, Virginia, USA, Nov. 06-11, 2000.

〔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 Transactions on Knowledge and Data Engineering*, Vol.16, No.1, pp.125-139, 2004.

〔10〕 H. Chen, Y. Xiao, and X. Shen, "Update-Based Cache Access and Replacement in Wireless Data Access," *IEEE Transactions on Mobile Computing*, Vol.5, No.12, pp.1734-1748, 2006.

〔11〕 W-C. Peng and M-S. Chen, "Efficient Channel Allocation Tree Generation for Data Broadcasting in a Mobile Computing Environment," *Wireless Networks*, Vol.9, No.2, pp.117-129, 2003.

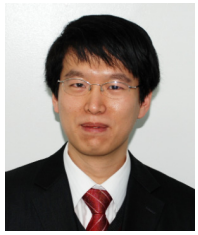
〔12〕 J-L. Huang and M-S. Chen, “Dependent Data Broadcasting for Unordered Queries in a Multiple Channel Mobile Environment,” *IEEE Transactions on Knowledge and Data Engineering*, Vol.16, No.9, pp.1143-1156, 2004.
〔13〕 S. Acharya and S. Muthukrishnan, “Scheduling On-demand Broadcasts: New Metrics and Algorithms,” *In Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp.43-54, Dallas, Texas, USA, Oct. 25-30, 1998.
〔14〕 S. Acharya, M. Franklin, and S. Zdonik, “Push and Pull for Data Broadcast,” *In Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pp.183-194, Tucson, Arizona, USA, May 11-15, 1997.
〔15〕 H-P. Hung and M-S Chen, “A General Model of Hybrid Data Dissemination,” *In Proceedings of the 6th International Conference on Mobile Data Management*, Ayia Napa, Cyprus, May 09-13, 2005.
〔16〕 S. Venkatraman, “Mobile Computing Models - Are They Meeting the Mobile Computing Challenges?,” *Association of Computing Machinery New Zealand*, Vol.1, No. , pp.3-12, 2005.

김 삼 근 (Sam-geun Kim) 정회원

1985년 부산대학교 계산통계학과
1988년 숭실대학교 전자계산학과(공학석사)
1998년 숭실대학교 전자계산학과(공학박사)
1992년~현재 한경대학교 컴퓨터공학과 교수

<관심분야> GIS, Spatial Data Mining, Web Service Computing, Mobile Computing, Business Intelligence

김 형 호 (Hyung-ho Kim) 정회원



2007년 한경대학교 컴퓨터공학과
2009년 한경대학교 컴퓨터공학과(공학석사)
2009년~현재 한경대학교 컴퓨터공학과 박사과정

<관심분야> GIS, Spatial Data Mining, Mobile Computing, Data Grid, Very Large Database

안 재 근 (Jae-geun Ahn) 정회원

1992년 서울대학교 산업공학과
1994년 서울대학교 산업공학과(공학석사)
1997년 서울대학교 산업공학과(공학박사)
1997년~현재 한경대학교 컴퓨터공학과 부교수

<관심분야> Optimization, MIS, SaaS, ASP, Mobile Computing