도심환경에서 위치의존 질의를 위한 방송과 캐싱 기법

(Broadcasting and Caching Schemes for Location-dependent Queries in Urban Areas)

정 일 동 † 유 영 호 †† 이 종 환 ††† 김 경 석 ††††
(Il-dong Jung) (Young-ho Yu) (Jong-hwan Lee) (Kyongsok Kim)

요 약 무선 통신 기술과 휴대형 정보 장치의 발달로 등장한 이동 컴퓨팅 환경(Mobile Computing Environment)은 사용자가 랩탑이나 PDA와 같은 휴대 가능한 장비를 이용해서 사용자의 물리적인 위치나 이동에 상관없이 무선 통신을 이용해서 서버 혹은 다른 컴퓨터의 자원과 함께 작업하는 것을 말한다. 최근 이동 컴퓨팅 환경에서 보편적인 형태가 되고 있는 위치 의존 질의(Location Dependent Query)는 위치에 의존하는 데이타를 처리하는 질의이다. 위치 의존 질의는 질의의 결과를 만들어 내는 중요한 척도가 위치이다. 위치 의존 질의를 효과적으로 지원하기 위해서는 이동 호스트의 캐싱 정책과 셀을 담당하는 지구국의 브로드캐스팅 정책이 중요하다. 적절한 캐싱 정책과 브로드캐스팅 정책을 정하기 위해서는 사용자의 이동과 데이타의 공간 속성을 고려해야 한다. 도심에서는 사용자가 도로를 따라서 이동하고 데이타가 도로에 인접해서 위치한다. 이런 특징을 가지는 도심에서 이동 호스트의 현재 위치에서 가장 가까운 곳은 직선 거리로 가장 가까운 곳이 아니라 이동 거리가 가장 짧은 곳이다. 따라서, 이전에 행해졌던 연구에서 사용한 직선 거리는 도심에 적합하지 않다. 직선 거리(Euclidean Distance)를 사용하면 이동 호스트의 이동 거리를 계산하기 위해서 피타고라스 정리를 이용해서 비슷하게 예상할 수 있지만, 실제 이동거리는 다양한 값이 나올 수 있기 때문에 적합하지 않다. 본 논문에서는 도심의 특성을 반영한 브로드캐스팅/캐싱 정책을 제안한다. 본 논문에서 제안하는 이동 호스트가 도심의 위치 정보를 효과적으로 캐싱할 수 있도록 인접한 데이타를 클러스터링해서 브로드캐스팅하여 이동 호스트의 구성 시간(setup time)을 최소화하였다. 그리고, 맨하탄 거리(Manhattan Distance)를 사용해서 위치 의존 질의에서 사용하는 데이타를 캐싱하고 질의를 처리하는 방법을 제안한다. 맨하탄 거리를 이용해서 캐싱하면 도로에 인접해서 위치한 데이타를 효과적으로 캐싱할 수 있다. 또한, 거리 계산 방법으로 맨하탄 거리를 사용하면 도심에서 실제 이동 거리와 비슷한 값을 알 수 있고, 직선 거리 계산식에 비해서 계산식도 간단하기 때문에 시스템 계산량도 줄일 수 있다.

키워드 : 이동 컴퓨팅, 자료 관리, 캐싱 정책, 방송 정책, 위치 의존 질의

Abstract The results of location-dependent queries(LDQ) generally depend on the current locations of query issuers. Many mechanisms, e.g. broadcast scheme, hoarding, or caching policy, have been developed to improve the system performance and provide better services, which are specialized for LDQs. Considering geographical adjacency of data and characteristics of target area, caching policy and broadcast scheme affect the overall performance in LDQ. For this reason, we propose both the caching policy and broadcast scheme, which these features are reflected in. Based on the adjacency of data in LDQ, our broadcast scheme use Hilbert curve to cluster data. Moreover, in order to develop the caching policy suitable for LDQ on urban area, we apply the moving distance of a MH(Mobile Host) to our caching policy. We evaluate the performance of the caching policy measuring the workload of MHs and the correctness of LDQ results,and the performance of the broadcast scheme measuring the average setup-time of MHs in our experiments. Finally, we expect that our caching

† 비 회 원 : LG전자 BAC연구소 주임 연구원
idjung96@daum.net
†† 비 회 원 : 부산대학교 컴퓨터 및 정보통신연구소 교수
yhyou@asadal.pnu.edu
††† 비 회 원 : 오토파워 부설연구소 연구원
jhwlee@asadal.pnu.edu
†††† 종신회원 부산대학교 정보컴퓨터 공학부 교수
gimgs@asadal.pnu.edu
논문접수 2003년 8월 1일
심사완료 2004년 10월 19일

policy provides more correct answers when executing LDQ in local cache and leads significant improvement of the performance of MHs. It also seems quite probable that our broadcast scheme leads improvement of battery life of the MH.

Key words : Mobile Computing, Contents management, caching, broadcasting, location-dependent query

1. 서 론

무선 통신 기술과 휴대형 정보 장치의 발달로 등장한 이동 컴퓨팅 환경(Mobile Computing Environment)은 사용자가 랩탑이나 PDA와 같은 휴대 가능한 장비를 이용해서 사용자의 물리적인 위치나 사용자의 이동에 상관없이 무선 통신을 이용해서 서버 혹은 다른 컴퓨터의 자원과 함께 작업하는 것을 말한다[1]. 이에 반해 전통적인 분산 컴퓨팅 환경은 클라이언트와 서버의 위치가 고정되어 있으며, 클라이언트가 연산하는 동안에 클라이언트와 통신하는 서버가 바뀌지 않는다고 가정한다. 이동 컴퓨팅 환경에서는 전통적인 분산 컴퓨팅 환경의 가정이 아니라 이동 호스트의 이동성, 네트워크 제약, 잦은 접속 단절 등을 고려해야 한다.

최근 이동 컴퓨팅 환경이 보급됨에 따라 이동 호스트의 현재 위치를 기반으로 하는 여러 가지 연구가 진행중인데, 가장 일반적인 연구는 위치 의존 질의(Location Dependent Query)에 관한 연구이다. 위치 의존 질의는 위치에 의존하는 데이타를 처리하는 질의이다. 위치 의존 질의는 질의의 결과를 만들어 내는 중요한 척도가 이동 호스트의 현재 위치이다. 위치 의존 질의를 효과적으로 처리하기 위해서는 이동 기지국 (Mobile Support Station)의 효과적인 데이타 브로드캐스팅 기법과 이동 호스트의 효율적인 데이타 캐시 기법이 필요하다.

건물의 배치가 단순한 시골보다는 건물의 배치가 복잡한 도심에서 위치 의존 질의를 이용하는 경우가 많을 것이다. 그러므로 도심에서는 많은 수의 이동 호스트가 위치 의존 질의를 요구할 것으로 예상된다. 도심의 특성을 무시한 일반적인 위치 의존 질의 처리 방법으로는 많은 질의를 효과적으로 처리할 수 없으며, 사용자에게 편리한 서비스를 제공할 수 없다.

본 논문에서는 도심의 특성을 고려하여 도심 환경에 적합한 이동 호스트의 캐시 기법과 이동 호스트를 효율적으로 지원할 수 있는 브로드캐스팅 기법을 제안한다. 그리고 제안한 이동 호스트의 캐시 기법이 사용자에게 적절한 결과를 반환하는가를 확인하고, 이동 호스트의 계산량을 확인하여 서비스의 품질을 높일 수 있는지를 알아본다. 또한, 이동 호스트의 구성 시간(setup time)을 측정해서 제안한 브로드캐스팅 기법이 이동 호스트를 효과적으로 지원하는지 알아본다. 본 논문의 구성은 다음과 같다. 2장에서는 이동 컴퓨팅 모델, 브로드캐스팅 기법, 캐시 기법, 위치 의존 질의에 대한 기존의 연구를 살펴본다. 3장에서는 본 논문을 서술하면서 사용한 가정과 정의를 정리하고, 도심의 특성을 고려하는 이동 호스트의 캐시 기법과 캐시 기법을 지원하는 브로드캐스팅 기법을 제안한다. 4장에서는 위치 의존 질의를 처리하는 이동 호스트와 데이타를 브로드캐스팅하는 이동 기지국을 모델링하고, 시뮬레이션을 통해서 제안한 브로드캐스팅 기법과 캐시 기법의 성능을 평가한다. 마지막으로 5장에서는 제안한 브로드캐스팅 기법과 캐시 기법을 정리하고 향후 연구 과제를 제시한다.

2. 관련 연구

2.1 이동 컴퓨팅 환경

이동 컴퓨팅 환경은 분산 시스템과 유사하지만, 다음과 같은 제약 사항이 있다[2].

- **네트워크 제약** : 이동 컴퓨팅 환경은 주로 이동 호스트에서 서버 방향(upstream)의 대역폭보다 서버에서 이동 호스트 방향(downstream)의 대역폭이 크다. 게다가 순수 방송 디스크(Pure Broadcast Disk) 환경은 서버에 질의를 요구할 수 없는 이동 호스트가 사용한다. 사용할 수 있는 대역폭이 10 Kbps 수 ~ Mbps 정도로 유선 망에 비해 작다.
- **잦은 접속 단절** : 이동 호스트는 정해진 곳에서 사용하는 것이 아니라 움직이기 때문에 접속이 끊어지는 경우가 많다. 뿐만 아니라 이동 호스트는 배터리를 절약하고 배터리의 사용 효율을 높이기 위해서 이동 호스트가 스스로 네트워크 연결을 끊는 경우가 많다.
- **배터리 용량 제약** : 이동 가능한 장비는 배터리를 충전하기 전까지 사용할 수 있는 에너지량이 정해져 있다.
- **작은 화면 크기** : PDA (Personal Digital Assistant)와 같이 이동 가능한 장비는 화면 크기가 작다. 그러므로 데이타를 입력받거나 결과를 출력할 때 개발자가 주의를 기울여야 한다.

이동 컴퓨팅의 위와 같은 제약 사항을 극복하기 위해서 브로드캐스팅 기법, 캐시 기법, 이동 호스트의 메모리 관리 기법과 같은 여러 가지 연구가 진행되고 있다.

2.2 브로드캐스팅 기법

이동 컴퓨팅 환경은 대역폭이 한정되고 전송 에러율

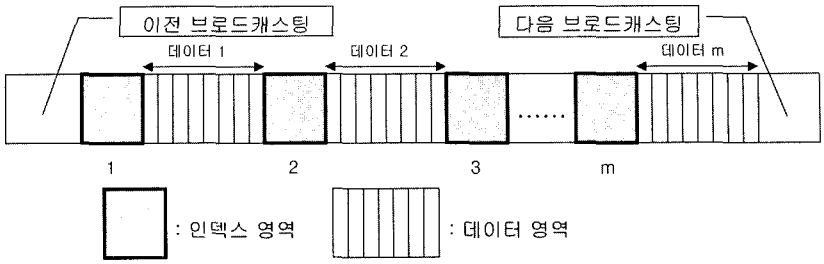


그림 1 (1, m) 인덱싱의 구조

이 높을 뿐 아니라 이동 컴퓨터의 전력이 제한되어 있는 등 많은 제약이 있다. 이동 컴퓨팅 환경은 주로 이동 호스트에서 서버 방향 (upstream)의 대역폭보다 서버에서 이동 호스트 방향 (downstream)의 대역폭이 상대적으로 훨씬 큰 비대칭적 통신 환경이다[3]. 이러한 통신 환경에서는 정보를 브로드캐스팅하여 전달하는 것이 효과적이다.

이동 호스트에서 자주 요구하는 데이타를 기지국에서 브로드캐스팅해서 전달함으로써 이동 호스트가 데이타를 요구하는데 필요한 배터리 사용을 줄이고 기지국이 담당하는 영역 안에 이동 호스트의 수가 증가하더라도 효과적으로 처리할 수 있다[4].

Imielinski와 Datta는 이동 호스트의 전력 소모량을 줄일 수 있는 연구를 하였다[5,6].

[6]은 이동 호스트의 전력 소모량을 줄일 수 있는 (1, m) 인덱싱 기법을 제안하였다. (1, m) 인덱싱 기법은 1번 브로드캐스팅하는 동안 인덱스를 m번 브로드캐스팅하는 방법이다. 또한, 인덱스를 부분적으로 중복시켜서 (1, m) 인덱싱 기법의 성능을 향상시키는 분산 인덱싱 기법을 제안하였다. 그림 1은 (1, m) 인덱싱을 사용한 브로드캐스팅 기법을 보여 준다.

(1, m) 인덱싱 기법은 각 인덱스 영역의 첫 번째 버킷(bucket)에 브로드캐스팅의 길이와 다음 브로드캐스팅 시작 시간을 표시하고 있는데, 이는 이동 호스트가 현재의 브로드캐스팅을 놓쳤을 때 이동 호스트가 다음 브로드캐스팅까지 채널에 접속하지 않고 기다릴 수 있도록 한다. 이 기법을 사용해서 이동 호스트의 조율 시간 (tuning time)을 줄일 수 있고, 결과적으로 이동 호스트의 배터리를 절약할 수 있음을 증명했다.

Datta는 인기도(popularity consciousness)와 무시 정도(ignore factor)를 이용해서 우선 순위를 정하고 우선 순위에 따라 브로드캐스팅할 데이타를 결정하는 적응형 브로드캐스팅 기법을 제안하였다. 여러 가지 상황을 가정하여 이동 호스트가 데이타를 받는 시나리오를 제시하고, 여러 가지 브로드캐스팅 시나리오를 확률을 이용해서 분석하였다. 또한, 이동 호스트와 서버를 모델링하고 실험을 통해 제안한 프로토콜이 이동 호스트의 접근 시간과 조율 시간(tuning time)을 줄이고 이동 호스트의 전력 소모량을 줄일 수 있음을 보였다[5].

2.3 캐시 기법

브로드캐스팅 기법의 도입과 함께 기지국에서 브로드캐스팅하는 데이타 중 앞으로 사용 가능성이 높은 데이타를 이동 호스트의 캐시에 저장하였다가 사용자의 질의에 응답할 때 캐시 내의 데이타를 이용하는 캐시 기법이 도입되었다. 캐시된 데이타의 사용은 질의의 응답 지연 시간을 줄이고 기지국과 이동 호스트와의 데이타 전송량을 줄인다. 이동 호스트가 활동 상태뿐만 아니라 접속 단절 상태에서도 사용자의 질의에 대해서 응답할 수 있는 장점을 갖는다[7].

[8]은 이동 컴퓨팅 환경에서 캐시 메커니즘에 대해서 정리하고, 무선 환경의 낮은 대역폭뿐만 아니라 이동 컴퓨팅 환경의 특성에 잘 적응할 수 있는 적응형 캐시 메커니즘을 제안하였다. 이는 캐시 아이템의 그래뉼래리티(granularity)를 속성 캐시와 객체 캐시로 나누어서 객체의 접근 확률에 따라 서버가 적절하게 캐시 기법을 결정하고, 이동 호스트의 객체 접근 패턴(pattern)에 따라 쉽게 적응할 수 있는 캐시 교체 정책을 제안했다.

2.4 위치 의존 질의

현재까지 이동 컴퓨팅 환경에서 위치 의존 질의에 관한 연구는 위치 의존 데이타에 관한 연구[9], 위치 의존 질의에 관한 연구[10], 위치 의존 질의를 지원하는 캐시 정책에 관한 연구[11,12] 등이 이루어졌다.

*위치 의존 데이타(Location Dependent Data)*는 이동 호스트의 위치에 따라 데이타의 가치가 결정되는 데이타이다. 보기를 들면, 전화 번호부, 교통량 정보, 날씨 정보, 지도와 같이 특정 지역에서만 의미를 가지는 것들이다.

*위치 의존 질의(Location Dependent Query)*는 위치 의존 데이타를 처리하는 질의이다. 질의의 결과는 명시적 혹은 묵시적으로 지정된 위치에 의존한다. 게다가 사

용자의 위치가 바뀜에 따라 질의의 결과가 바뀐다. 최근에 위치 의존 애플리케이션이 이동 컴퓨팅 환경에서 점점 더 일반화되고 있다[12].

[9]에서는 위치 의존 데이타의 도메인과 위치, 데이타의 시공간적인 중복, 위치 의존 데이타의 시공간적 일관성, 위치 의존 데이타의 질의 처리에 관한 연구와 같이 위치 의존 데이타의 여러 가지 측면을 다루었다. 여기서 다음의 특성을 정리하였다.

1) 데이타 집합의 의미와 그 가치는 특정 지역과 연관되어 있다.
2) 질의의 결과는 질의를 요구한 지리학적인 위치에 의존한다.
3) 질의는 특정 위치에서만 유효하다.

아래 보기 1) 에서 이러한 특성을 알 수 있다.

보기 1) 철수는 처음으로 춘천에 여행을 갔다. 여행 중에 시간이 늦어 숙박을 하기 위한 호텔을 찾으려고 자신의 이동 호스트에 "근처에서 가장 가까운 호텔을 조회하라." 와 같이 질의를 하였다.

보기 1)의 "근처에서 가장 가까운 호텔을 조회하라"라는 질의는 이동 호스트의 현재 위치가 춘천임을 고려하여 그에 맞는 질의 결과를 돌려준다.

[10]에서는 일반적인 위치와 관련한 여러 가지 관점을 제시하고, 위치 의존 질의의 형식을 제안하였다. 위치 관련 속성 (Location Related Attribute) 과 위치 무관 속성 (Non-Location Related Attribute) 을 정의하고, 위치 관련 릴레이션을 위치 관련 속성이 있는 릴레이션이라고 정의하였다. 또한 모든 공간 연산자와 "straight ahead"와 같은 연산자를 포함하는 위치 관련 연산자 (Location Related Operator)를 정의하였다. 정의한 연산자와 릴레이션으로 실제로 위치 의존 질의의 처리 형태에 대해서 서술하고 있다.

[11]은 이동 호스트의 위치가 시간에 따라 변화하는 속성이 있고 이에 따라 이동 호스트 내에 캐시된 데이타에 대한 미래의 사용 가능성이 변경되는 점을 고려한 캐시 교체 기법을 제안하였다. 이는 데이타의 위치와 데이타가 영향을 미치는 공간적 범위를 데이타의 공간 속성으로 정의하고 이러한 속성을 이용한 것이다.

[12]은 의미적 캐시(semantic cache)를 이용해서 위치 의존 질의를 효과적으로 처리할 수 있는 기법을 제안하였다.

1) 사용자의 현재 위치에서 혹은 사용자의 이동 방향에 가까운 위치 의존 데이타를 사용할 가능성이 크기 때문에 전통적인 캐시 관리 정책(LRU, MRU 등)은 위치 의존 질의를 처리하는데 적절하지 못하다.
2) 위치 의존 질의 사이의 의미적 지역성(semantic locality)은 사용자의 이동 경로와 연관성이 크다.

위의 1)과 2)의 성질을 이용해서 질의 패턴에 동적으로 적응할 수 있도록 사용자의 이전 질의 결과의 의미를 적절히 캐시하고, 이후의 질의에서 그 결과를 사용하는 방법을 제안하였다.

이동 호스트의 캐시 정책을 개선하여 위치 의존 질의 처리 효율을 높이는 기존의 연구는 위치 의존 질의의 처리 성능을 높이는데 한계가 있다. 그 이유는 위치 의존 질의가 사용되는 특수한 환경인 도심과 이동 호스트를 지원하는 기지국의 브로드캐스팅 기법을 고려하지 않고 있기 때문이다. 이동 단말기를 가진 사용자가 도심에 더 많을 것이므로 도심 환경에서 좋은 성능을 보이는 이동 호스트의 캐시 정책과 기지국의 브로드캐스팅 기법이 필요하다.

본 논문에서는 도심 환경을 고려한 위치 의존 질의의 결과를 반환하고, 이동 호스트의 배터리 소모량을 절약할 수 있는 이동 호스트의 캐시 기법과 기지국의 브로드캐스팅 기법을 제안한다. 그리고, 위치 의존 질의 환경을 모델링하고 제안한 기법들의 성능을 시뮬레이션을 통해서 확인한다.

3. 도심 환경을 고려한 위치 의존 질의 지원 방안

3.1 가정과 용어 정의

이 장에서는 위치 의존 질의를 효과적으로 처리하는 방안을 제시하기 전에 본 논문에서 가정하는 이동 컴퓨팅 환경과 도심 환경을 서술하고, 논문에서 사용한 용어를 정리한다.

3.1.1 이동 컴퓨팅 환경을 위한 가정

본 논문에서는 2.1 절의 이동 컴퓨팅 환경의 모델을 사용하며, 다음의 4 가지를 가정한다.

1) 한 기지국은 자신이 담당하는 셀 영역 내의 이동 호스트들에게 자신의 데이타베이스 내에 있는 데이타를 주기적으로 브로드캐스팅한다.
2) 이동 호스트는 기지국의 셀 내에서 항상 자신의 위치를 알 수 있다.
3) 이동 호스트의 캐시 용량은 서버의 데이타베이스의 용량보다 작고, 이동 호스트는 브로드캐스팅되는 내용 중 자신이 필요한 것을 선택적으로 캐시에 저장할 수 있다.
4) 이동 호스트는 이동 호스트의 캐시 영역에 포함되는 데이타를 캐시에 저장한다.

3.1.2 도심 환경을 위한 가정

공간 데이타나 멀티미디어 데이타와 같은 특정 데이타를 다루는 시스템의 성능을 향상시키기 위해서는 데이타의 특성을 이용해야 한다. 보기를 들어 멀티미디어 데이타베이스 시스템을 개발하기 위해서는 멀티미디어

데이타의 특징을 먼저 파악해야 한다. 멀티미디어 데이타의 표현 방법, 저장 방법, 질의 방법, 질의 결과의 프리젠테이션(presentation) 방법과 같은 여러 가지 사항을 고려해야 한다.

위치 의존 질의를 처리하는 시스템도 역시 위치 의존 질의의 대상이 되는 위치 의존 데이타의 특징을 이용한다면 표현, 저장, 질의, 결과 프리젠테이션을 효과적으로 할 수 있을 것이다. 위치 의존 데이타의 특징은 2.4절에서 밝힌 것과 같이 이미 [9]에서 연구되었다.

이동 호스트를 휴대한 사용자는 지구 위의 어느 지방이라도 여행할 수 있지만, 여행 혹은 업무 처리 관계로 도시를 방문할 일이 많을 것이다. 한적한 시골에서는 도시와 달리 건물의 배치가 단순하고 위치를 금방 알 수 있기 때문에 위치 의존 질의를 사용하는 경우가 적지만, 도시에서는 높은 건물이 많고 건물의 배치가 복잡하기 때문에 사용자가 특정 건물을 찾을 때 어려움이 많을 것이다.

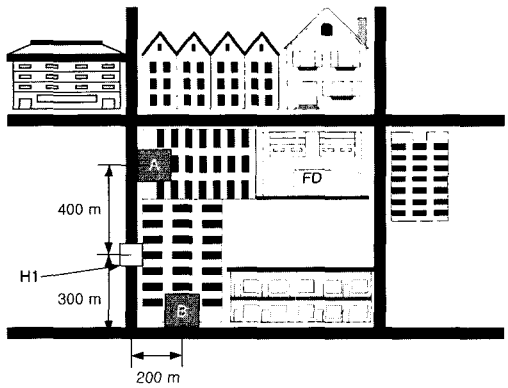


그림 2 도심 환경의 건물 배치

도심의 건물 배치 모습은 그림 2에서 볼 수 있다. 그림 2에서 알 수 있는 도심의 특징은 다음과 같다.

1) 도심 안의 모든 물체 (사용자) 는 도로를 따라 이동한다.
2) 도심의 도로는 직각으로 교차한다.
3) 도로를 따라 건물이 배치되어 있다.
4) 주요 건물이 많이 존재하기 때문에 사용할 데이타가 많다.

그림 3 도심의 특성 정리

이전 연구의 위치 의존 질의를 처리하는 시스템은 이동 호스트와 데이타 사이의 직선 거리를 사용해서 그 거리를 표현한다.

거리(距離) : 명) ① 두 곳 사이의 떨어져 있는 거리. ② 두 점 사이의 간격의 크기. 그 두 점을 연결하는 직선의 길이를 나타냄. ③ 서로 가까이 사귀지 못하고 새뜬 사이 [새 우리말 큰 사전, p. 129]

일반적으로 거리의 의미로 위 정의 ①의 의미를 사용하는데, 도심에서는 두 지점 사이의 떨어진 거리를 직선 거리로 말하기 곤란하다. 이동 호스트(사용자)들은 도로를 따라 이동하기 때문에 도시에서는 두 지점 사이의 의미적인 거리가 사람이 물리적으로 이동하는 거리를 거리로 보는 것이 적합하다. 그러므로 도심의 위치 의존 질의를 처리할 경우에도 직선 거리를 사용하여 질의 결과를 생성하는 것보다 이동 호스트가 도로를 따라 실제로 이동하는 거리를 사용하여 질의 결과를 생성하는 것이 적합하다.

이동 호스트가 도로를 따라 이동하는 거리를 알기 위해서는 도로의 특징을 알아야 한다. 도심에서 도로는 직교한다는 사실을 고려하여 두 지점 사이의 거리를 맨하탄 거리(Manhattan distance)로 계산하면 이동 거리를 정확히 예측할 수 있다.

그림 2에서 A와 B가 호텔이며, 이동 호스트 H1이 *"현재 위치에서 가장 가까운 호텔을 조회하라."* 라는 질의를 요구했다고 가정하자. 이동 호스트 H1은 직선 거리를 사용한다면 질의 결과로 직선 거리로 가장 가까운 B를 반환할 것이다. 이동 호스트의 현재 위치에서 B까지의 직선 거리는 360m이지만, 실제 이동 거리는 500m이므로 A까지의 거리인 400m보다 더 멀다. 이런 이유로 이동 호스트 H1의 현재 위치에서 B가 가장 가까운 호텔이라고 할 수 없다. 이동 호스트 H1이 맨하탄 거리를 사용한다면 질의 결과로 맨하탄 거리로 가장 가까운 A를 반환할 것이다. 현재 위치에서 400m만 이동하면 A에 도착할 수 있기 때문에 정확한 결과를 반환했다고 할 수 있다.

3.1.3 용어 정의

이 절에서는 본 논문에서 사용하는 용어와 개념에 대해서 정리한다.

기지국에서 관리하는 셀 내에 위치하는 이동 호스트와 위치 의존 데이타를 그림 4를 이용해서 설명하겠다.

정의 1. 데이타의 위치 DL_i(Data Location)은 데이타 D_i의 2차원 공간상의 위치로, 한 점의 좌표로 표시한다.

$$DL_i = (X_i , Y_i)$$

위치 의존 질의에서 사용하는 데이타는 건물 데이타이므로 건물이 보통 직사각형이라는 점을 고려해서 건물의 위치를 건물의 중점으로 나타낸다. 본 논문에서는 건물의 위치를 쉽게 다루기 위해서 점으로 표현하였다.

정의 2. 데이타 D_i의 속성은 공간적 위치, ID, 이름, 종류, 인덱스 ID, 특성으로 구성된다.

D_i = {id, location, name, class, index_id, property}

논문에서 사용하는 데이타를 표시하는 투플은 6개의 속성을 가진다. ID는 하나의 데이타베이스 내에서 유일

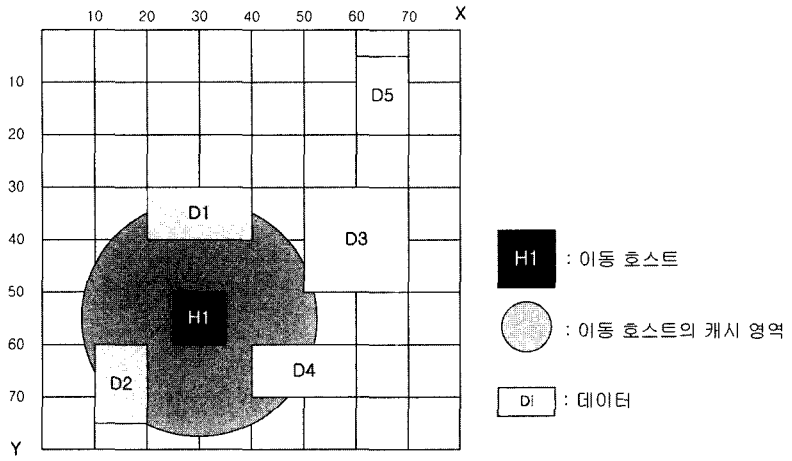


그림 4 셀 내의 이동 호스트와 데이타의 보기

한(unique) 정수 값으로 데이타를 구분하는 키 값이며, 종류는 건물의 용도, 즉 병원, 아파트 등의 용도에 따른 의미적 분류이다. 인덱스 ID는 데이타가 속하는 인덱스 ID를 의미한다. 인덱스 ID에 대한 내용은 5.2절에서 설명하겠다.

위의 정의에 따라 그림 4의 데이타의 속성을 나타낸 것은 표 1과 같다.

이동 호스트의 근처에 있는 데이타가 질의에 사용될 가능성이 크다. 이동 호스트의 근처에 있는 데이타인지 검사하기 위해서 이동 호스트는 데이타가 이동 호스트의 캐시 영역에 포함되는지를 검사한다. 따라서 캐시 영역은 이동 호스트에 가치가 있는 데이타를 판별하는 기준이 된다. 보기를 들면 그림 4의 이동 호스트 H1의 캐시 영역에는 D1과 D2가 포함되므로 이동 호스트 H1은 D1과 D2를 캐시에 저장한다.

표 1 [그림 5]의 데이타를 저장한 테이블 보기

id	location	name	class	index_id	property	
1001	(30, 35)	김 치과	병원	3000300	의원	D1
1002	(15, 67)	성 안과	병원	1000700	병원	D2
2001	(60, 40)	낙원 상가	상가	5000300	90평	D3
2002	(50, 65)	효원 상가	상가	5000700	70평	D4
3001	(65, 17)	효원 아파트	아파트	7000100	100세대	D5

정의 3. 캐시 영역(Cache Region)은 이동 호스트의 캐시의 대상이 되는 영역으로 캐시에 저장되는 영역이다. 캐시 영역의 크기는 이동 호스트에서 일정한 거리 안의 영역으로 표현한다.

정의 4. 활동 모드(active mode)는 이동 호스트가 연산을 하거나 데이타를 받기 위해서 채널에 접속 중이거나 데이타를 전송하는 상태를 말한다. 본 논문에서는 데이타를 보내거나 받는 상태로 정의한다.

정의 5. 대기 모드(doze mode)는 이동 호스트가 불필요한 배터리 사용을 방지하기 위해서 대부분의 기능을 멈추고 최소한의 전력을 사용하는 상태를 말한다.

유선망에서는 응답 시간이 중요한 성능 평가 척도가 되지만, 무선망에서는 응답 시간뿐만 아니라 배터리 사용량도 중요한 성능 평가 척도가 된다. 이동 호스트가 채널에 접속하거나 데이타를 전송하기 위해서는 활동 모드(active mode)에서 동작하며, 이 때는 전력을 많이 사용한다. 그리고 전원을 절약하고 배터리 사용 시간을 연장하기 위해서 이동 호스트가 채널에 접속하지 않는 대기 모드 (*doze mode*) 로 바꾼다[5].

본 논문에서는 이동 호스트의 구성 시간(setup time)과 위치 의존 질의의 결과가 최적 결과에 가까운 정도를 이용해서 브로드캐스팅 기법과 캐시 기법의 성능을 평가한다.

정의 6. 구성 시간(setup time)은 이동 호스트가 대기 모드(doze mode)에서 활동 모드(active mode)로 바꾸거나, 활동 모드에서 대기 모드로 바꾸는데 걸리는 시간이다[6].

정의 7. 활동 시간(active time)은 이동 호스트의 전력 소모 시간의 대부분을 차지하는 활동 모드에서 이동 호스트가 작동한 시간으로 정의한다.

보기를 들어 구성 시간과 활동 시간을 설명하겠다. 그림 5에서 이동 호스트가 (T_2 - T_1) 시간 동안 인덱스를 받고, (T_6 - T_5) 시간과 (T_{10} - T_9) 시간 동안 원하는 데이타를 받았다면, 조율 시간(tuning time) = (T_2 - T_1)이다. 이동 호스트가 대기 모드일 때 채널에서 데이타를 바로 받을 수 없으므로 활동 모드로 바꾸는 시간과 활동 모드에서 대기 모드로 바꾸는 시간이 필요하며, 구성 시간 = (T_{11} - T_{10}) + (T_9 - T_8) + (T_7 -

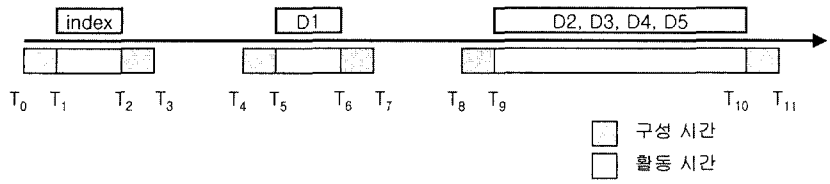


그림 5 구성 시간, 활동 시간

T_6) + (T_5 - T_4) + (T_3 - T_2) + (T_1 - T_0)이다.

3.2 도심 환경을 고려한 캐시 기법

3.2.1 위치 의존 질의 모델링

위치 의존 질의를 효과적으로 처리하기 위해서 이동 호스트의 움직임과 위치를 표현해야 하는데, 본 논문에서는 [12]의 위치 의존 질의 모델을 사용한다.

이동 호스트의 움직임과 위치를 표현하기 위해서 위치, 속도, 방향 측면에서 이동 호스트의 이동을 모델링한다. 일반적으로 물체는 3차원 공간에서 다양한 속도, 방향으로 이동하는데, 본 논문에서는 이동 호스트의 이동을 쉽게 모델링하기 위해서 이동 호스트가 2차원 공간에서 일정한 속도로 움직인다고 가정하도록 하겠다.

정의 8. 위치 L은 (L_x, L_y)로 표현하는데, L_x와 L_y는 정수이다.

위치를 나타낼 때는 보통 경도와 위도를 이용해서 많이 나타내고, 경도와 위도는 소수로 나타낸다. 하지만, 본 논문에서는 일반화가 쉽도록 정수로 표현한다. 지리적인 공간을 정해두고 실험을 하므로 정의 8은 기준점과의 상대적인 거리를 이용해서 위치를 표현하도록 한다.

정의 9. 속도 V는 <V_x , V_y>로 표현하는데, V_x와 V_y는 정수이다.

모델링을 쉽게 하기 위해서 X축의 양의 방향은 동쪽, X축의 음의 방향은 서쪽, Y축의 양의 방향은 북쪽, Y축의 음의 방향은 남쪽으로 정의한다. 2차원 공간에서 물체의 움직임의 속도와 방향을 정의하기 위해서 벡터를 이용한다. 보기를 들어 이동 호스트 A가 (30, 30) 위치에서 동쪽으로 3의 속도로 이동한 다음, 남쪽으로 4의 속도로 이동하였다. 다시 서쪽으로 5의 속도로 이동한 다음, 남쪽으로 4의 속도로 이동하였다고 하자. 그림 6은 위의 보기를 그림으로 표현한 것이다.

속도를 이용하면 이동 호스트의 위치를 예측할 수 있다. T 시간에 LL의 위치에 이동 호스트 H가 있다고 가정하자. 이동 호스트 H가 속도 V = <V_x, V_y>로 δ의 시간만큼 움직인다면 H는 (T+δ)의 시간에는 $L_{\delta t}$ = (L_x + V_x * δ, L_y + V_y * δ)의 위치에 도착해 있을 것이다. 어느 점에서 움직이는 물체의 상태는 위치와 속도로 표시할 수 있다. 게다가 시간이 지남에 따라 물체는 위치가 변하므로 이동하는 물체의 상태는 특정 시간과 연관

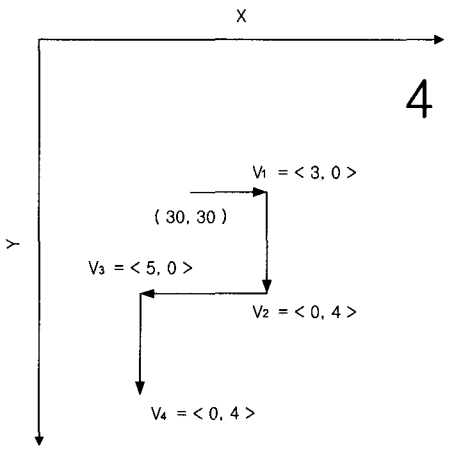


그림 6 이동 호스트의 이동 속도와 방향의 보기

지어서 고려하여야 한다. 따라서 이동 호스트의 상태는 정의 10과 같이 정의할 수 있다.

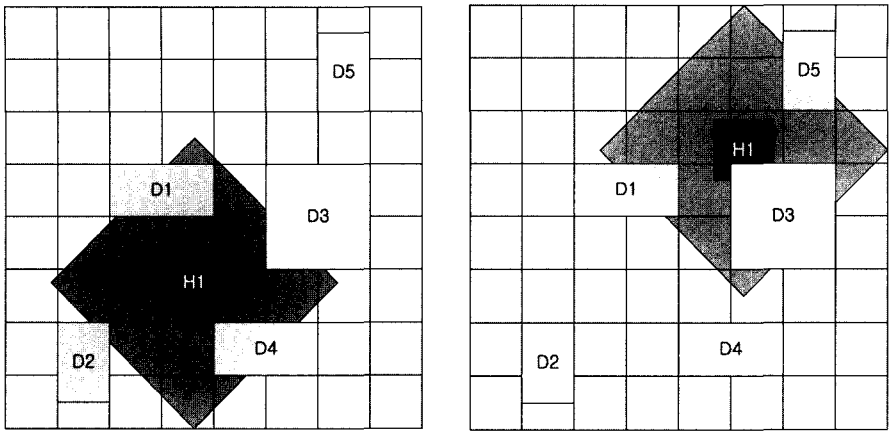
정의 10. 이동 호스트 H의 상태는 (H_T, H_L, H_V)로 정의하는데, H_T는 시간, H_L은 위치, H_V는 속도를 표시한다.

그림 7과 같이 이동 호스트 H가 T_1의 시간에 (30, 30)에서 출발한다고 가정하자. T_2에는 (33, 30)의 위치에, T_3에는 (33, 34)의 위치에 도착한다면 (T_1, L_1(30, 30), V_1<3, 0>), (T_2, L_2(33, 30), V_2<4, 0>)로 표현할 수 있다. 이동 호스트의 상태를 추적하는 것은 위치 의존 질의를 처리하는데 도움이 되며 캐시 관리를 효율적으로 할 수 있도록 한다. 이를 이용해서 앞으로 방문할 가능성이 높은 위치의 데이타를 미리 캐시함으로써 질의 응답에서 이용할 수 있다.

위치 의존 질의 모델링의 중요한 것은 위치와 관련한 조건의 표현 방법이다. 본 논문에서는 일반적인 조건으로 위치와 관련한 정보를 다루도록 하겠다. 위치 의존 질의를 처리하기 위해서는 정확한 위치를 지정해야 한다. 이 과정에서 위치 속성의 값을 질의에 포함시켜야 한다.

보기 2) ***위치 의존 질의 모델링***

호텔과 관련된 속성으로 ***name(이름), location(위치), price(가격), vacancy(빈 방 여부)****등을 가진다고*



(a) T_1의 이동 호스트 H_1 (b) T_2의 이동 호스트 H_2

그림 7 이동 호스트의 위치에 따른 캐시 영역 ($T_1 < T_2$)

가정하자. 이동 호스트가 질의 Q="반경 2Km 이내의 호텔 중에서 숙박비가 10만원 이하인 호텔을 찾아라."를 위치 L에서 요구했다. Q의 조건 Q_p = (price ≤ 100,000) ∧ (L_x − 2000 ≤ data.x ≤ L_x + 2000) ∧ (L_y − 2000 ≤ data.y ≤ L_y + 2000)으로 표현할 수 있다.

Q_p는 바로 찾을 수 있는 조건이 아니며, 최악의 경우에는 관련된 데이타 전부를 검색해야 하는 조건이다. 질의 Q는 반드시 위치인 L의 값이 설정되어야만 그 결과를 얻을 수 있다.

본 논문에서는 위치 의존 질의를 할 때 질의 대상이 되는 영역을 "보기 2)의 질의 반경을 지정하는 형태"가 아니라, 가장 일반적인 형태가 될 것으로 예상되는 "현재 위치에서 가장 가까운 호텔을 찾아라."와 같이 현재 위치에서 가장 가까운 데이타를 찾는 질의를 연구 대상으로 한다.처음 여행하는 곳에서는 여러 가지 데이타를 보려고 하기보다는 가장 가까운 데이타를 보려고 할 것으로 예상되기 때문에 본 논문에서는 가장 가까운 곳을 찾는 질의를 연구 대상으로 정한다.

3.2.2 이동 호스트의 위치 변화를 고려한 캐시 기법

이동 호스트가 이동하는 동안 어떤 정보를 이용했는지를 알아야 하기 때문에 최적의 캐시 교체 정책은 이동 호스트의 정확한 정보를 이용해야 알 수 있다. 하지만, 전체 정보를 유지하는 비용이 너무 크기 때문에 사용자의 현재 상태에 대한 정보를 이용해서 현재 수준에서 가장 최적의 캐시 정책을 수행할 수 있다[12].

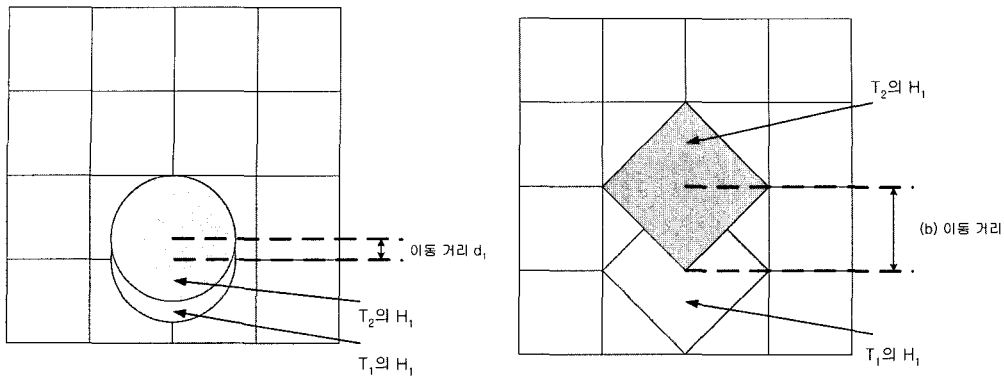
이동 호스트는 시간에 따라 위치가 바뀌는 속성을 갖는데, 질의가 요청된 위치에 따라 질의의 결과 값이 달라진다.이는 이동 호스트의 위치에 따라 이동 호스트 내에 캐시되는 데이타에 대한 의미와 가치를 다르게 평가해야 하는 필요성을 보여준다[11]. 이동 호스트가 서버에서 브로드캐스팅하는 데이타 중 일부를 자신의 캐시에 저장할 때, 이동 호스트의 캐시 영역 안에 데이타의 위치가 포함되는 데이타를 선택한다. 아래의 그림 8을 살펴보기로 하자.

T_i의 시간에 (30, 55)의 위치에 있던 이동 호스트 H_1이 T_2의 시간에는 (50, 25)의 위치로 이동했다고 가정하자. T_1의 시간에는 그림 7의 (a) 그림에서 알 수 있듯이 이동 호스트의 캐시 영역에 포함되는 D1과 D4이다(D2와 D3은 데이타의 중점인 위치 L_2와 L_3의 값이 캐시 영역 밖에 있다). 그러므로 이동 호스트 H_1의 캐시에는 D1과 D4가 저장되어 있다. T_2의 시간에는 그림 7의 (b) 그림에서 알 수 있듯이 이동 호스트 H_1의 캐시 영역에 포함되는 데이타는 D1과 D3, D5이다.

위에서 살펴본 이동 호스트의 위치에 따른 캐시 내용 변화의 필요성을 이동 호스트의 캐시 교체 알고리즘에 적용시켜 보자. 이동 호스트가 이동함에 따라 새로운 데이타가 이동 호스트의 캐시 영역에 포함되면 이동 호스트는 자신의 캐시에 데이타를 저장한다. 이 때 이동 호스트의 캐시 크기는 한정되어 있으므로 적절한 캐시 교체 알고리즘을 사용해야 한다. 캐시 교체 알고리즘은 캐시 내의 데이타 각각에 대한 캐시 교체 점수(Cache Replacement Score)를 계산하여 이용하는데 데이타의 위치를 기준으로 캐시 교체를 할 때는 캐시 교체 점수를 이동 호스트의 현재 위치와 데이타의 위치 사이의 거리를 사용한다.

정의 11. 이동 호스트 H_i의 캐시 크기($Cmax_i$)는 캐시 내에 저장 가능한 데이타의 개수이다.

정의 12. 이동 호스트 H_i의 캐시에서 데이타의 위치를 기준으로 맨하탄 거리를 이용한 캐시 교체 점수(CRS_MD : Cache Replacement Score based on Man-



(a) 이전 연구에서의 캐시 교체 점수 계산하는 경우 (b) 본 논문에서 캐시 교체 점수 계산하는 경우

그림 8 이전 연구와 본 논문의 캐시 교체 점수 계산하는 경우 ($T_1 < T_2$)

hattan Distance)는 이동 호스트 H_i의 위치 (XH_i, YH_i)와 데이타 D_j의 위치 (X_j, Y_j) 사이의 맨하탄 거리이다. 직선 거리를 계산하는 식에 비해서 맨하탄 거리를 계산하는 식이 간단하기 때문에 식을 연산하는 계산량이 적다. 그러므로, 계산 시간이 짧다.

$$MD = |XH_1 - X_2| + |YH_1 - Y_2|$$

정의 12의 식을 이용해서 캐시 내의 각 데이타의 캐시 교체 점수를 계산하여 새로 캐시에 저장할 데이타의 교체 점수보다 큰 (이동 호스트와 거리가 먼) 기존의 데이타를 캐시에서 삭제하고 캐시 교체 점수가 낮은 새 데이타를 삽입한다.

이동 호스트와 데이타의 거리를 캐시 교체 점수로 사용하는 것은 다른 방법(LRU, MRU 등)을 사용하는 것보다 간단하다. 또한, 위치 의존 데이타의 공간적인 의미를 이용하기 때문에 위치 의존 질의 처리 효율을 높일 수 있다. 하지만, 이동 호스트의 위치가 바뀌는 동안 계속해서 캐시 내의 데이타의 캐시 교체 점수를 계산해야 하기 때문에 시스템의 계산량이 크다. 특히 건물 데이타가 많은 도심에서는 이동 호스트의 위치가 조금만 바뀌어도 캐시 내 데이타의 캐시 교체 점수를 전부 계산해야 한다. 이는 시스템의 성능을 저하시킨다.

그림 8은 이동 호스트 H_1이 T_1의 시간과 T_2의 시간까지 이동하는 경우 이동 호스트의 캐시 내의 각 데이타의 캐시 교체 점수를 계산하는 시기를 보여 준다. 그림 8(a)는 이동 호스트의 위치가 조금만 바뀌어도 캐시 교체 점수를 다시 계산하는 것을 보여 주는데, 이는 이동 호스트가 계속해서 움직이는 경우에 캐시 교체 점수 계산을 계속해야 하기 때문에 연산 장치의 계산량이 많아진다. 캐시 교체 점수를 계산하기 위해서 연산 장치가 계속 연산을 한다면 다른 동작을 하기 위해서 연산 시간을 할당할 수 없기 때문에 시스템의 성능이 떨어진다.

본 논문에서는 그림 8(b)와 같이 이동 호스트의 위치가 격자만큼 움직였을 때 이동 호스트의 위치가 바뀐 것으로 인식하여 격자의 캐시 교체 점수를 계산하는 방법을 제안한다. 게다가 격자 1개를 데이타로 취급하면 캐시 교체 점수를 계산할 때 이동 호스트와 데이타의 거리를 캐시 교체 점수로 사용하지 않고, 이동 호스트와 인접 격자의 거리를 캐시 교체 점수로 사용하면 캐시 교체 점수를 계산할 대상의 수가 줄어든다.

이동 호스트가 격자를 벗어날 때 이동 호스트의 위치가 변한 것으로 인식하고 격자의 캐시 교체 점수를 계산한다면 데이타의 양이 많은 도심에서도 효과적으로 대처할 수 있다. 게다가 도심 속에 데이타의 양이 많아지더라도 격자의 수는 정해져 있기 때문에 이동 호스트의 계산량이 늘어나지 않는다. 캐시 관리를 위해서 많은 리소스를 사용하는 이전 시스템에 비해서 성능이 우수할 것으로 예상된다.

3.3 위치 의존 질의를 지원하는 브로드캐스팅 기법

이동 호스트가 위치 의존 질의를 처리하는 성능을 높이더라도 기지국의 데이타 전달하는 효율이 떨어지면 애플리케이션의 성능이 떨어진다. 이번 장에서는 애플리케이션의 성능을 높이기 위해서 이동 호스트가 위치 의존 질의를 효과적으로 지원할 수 있는 브로드캐스팅 기법을 제안한다.

이동 컴퓨팅 환경에서 기지국이 데이타를 브로드캐스팅할 때, 이동 호스트의 조율 시간(tuning time)을 줄이기 위해서 기지국은 브로드캐스팅되는 데이타의 순서를 알려주는 인덱스를 먼저 브로드캐스팅한다.

데이타는 지리 데이타이므로 2차원 공간에 분포하는 것으로 가정한다. 2차원 공간인 전체 영역에 존재하는 데이타를 브로드캐스팅하기 위해서 데이타의 브로드캐스팅 순서를 정해야 한다. 데이타의 좌표를 이용해서 브

로드캐스팅 순서를 결정하면 각 데이타의 순서에 일정한 규칙이 없고 데이타가 흩어져서 브로드캐스팅된다.

본 논문에서는 기지국이 담당하는 전체 영역을 그림 9와 같이 격자(grid)로 나누어서 격자 별로 데이타를 저장하고 격자 단위로 브로드캐스팅하도록 한다. 격자 단위로 인덱스를 구성하므로 전체 인덱스의 크기가 작아진다. 따라서 인덱스의 브로드캐스팅 시간이 짧아지고, 이동 호스트의 조율 시간이 짧아진다.

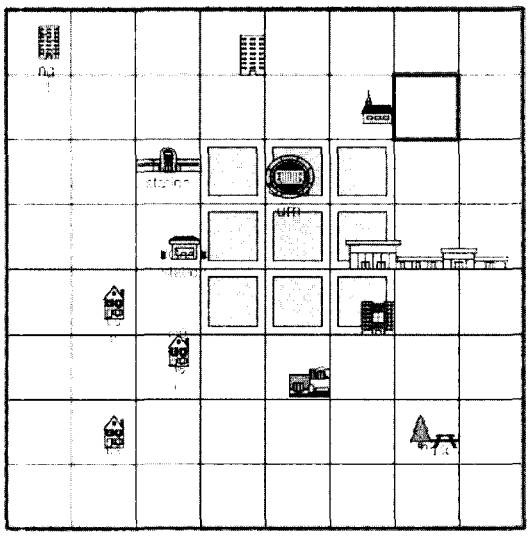


그림 9 전체 영역을 격자로 나눔, 캐시 영역이 300m인 이동 호스트가 (5,4) 격자에 위치함

정의 13. 격자(grid)는 기지국이 담당하는 셀(cell)을 일정한 크기로 나눈 것이다.

전체 영역을 일정한 크기의 격자로 나누어서 인덱스를 구성하기 때문에 인덱스를 구성하기 쉽고, 데이타를 저장할 때 같은 격자에 속하는 것을 인접하게 저장할 수 있다. 게다가 데이타를 격자 별로 관리하기 때문에 관리비용이 줄어든다. 이동 호스트도 격자 단위로 위치의존 데이타를 관리할 수 있으므로 효율이 증가할 것이다.

이동 호스트는 자신의 캐시 영역에 속하는 데이타를 저장하기 때문에 이동 호스트에서 필요한 데이타가 인접해서 브로드캐스팅되면 이동 호스트는 구성(setup) 횟수가 줄어들어 활동 시간이 줄어든다. 보기를 들어 캐시 영역의 크기가 300m(격자 1개의 크기 : 200m * 200m)인 이동 호스트가 그림 11의 (5, 4) 격자에 위치하고 인접한 격자가 그림 5의 (T6-T5) 시간과 (T10-T9) 시간 동안 브로드캐스팅된다면 이동 호스트는 데이타를 받기 위해서 데이타가 브로드캐스팅되기 전에 활동 모드로 바꾼 후 데이타를 기다려야 한다. 만약 두 데이타가 인접해서 브로드캐스팅되면 한 번만 구성해서 두 데이타를 받을 수 있으므로 그림 6의 (T7-T6) 시간과 (T9-T8) 시간이 필요하지 않다.

이와 같은 방법으로 그림 9의 인접한 격자를 클러스터링해서 데이타를 브로드캐스팅하면 이동 호스트의 구성 시간이 줄어든다. 하지만, 이동 호스트가 전체 영역에 분포하고 있다면 인접 격자가 인접해서 브로드캐스팅되는 특정 격자에 속한 이동 호스트를 제외한 나머지 이동 호스트의 구성 시간이 커진다. 그러므로 대부분의 이동 호스트가 좋은 성능을 얻을 수 있는 방법이 필요하다.

본 논문에서는 인접한 격자를 인접해서 브로드캐스팅하기 위해서 공간-채움 곡선을 사용한다. 공간-채움 곡선은 다차원 공간을 1차원 공간으로 매핑(mapping) 할 때 많이 사용하는데[13], 공간-채움 곡선을 사용하면 공간에서 인접한 데이타를 인접시켜서 선형으로 저장할 수 있다. 또한, 다차원 공간이 1차원 공간으로 매핑되면 1차원 접근 방법(access method)을 사용할 수 있기 때문에 데이타 관리 방법이 쉬워진다.

일정한 크기의 격자로 나눈 전체 영역에 공간-채움 곡선을 적용해서 브로드캐스팅 순서를 정한다. 전체 영역에 공간-채움 곡선을 적용하면 그림 10과 같이 되는데, 그림 10은 Hilbert 곡선을 적용한 보기이다.

본 논문에서는 인접한 공간 객체를 클러스터링해서 저장 장치에 1차원으로 저장할 때 사용하는 선형 클러스터링 방법을 사용해서 브로드캐스팅 순서를 결정할 것을 제안한다.

그림 10과 같이 공간-채움 곡선(Space-filling Curve)을 이용해서 전체 지역의 브로드캐스팅 순서를 정한다.

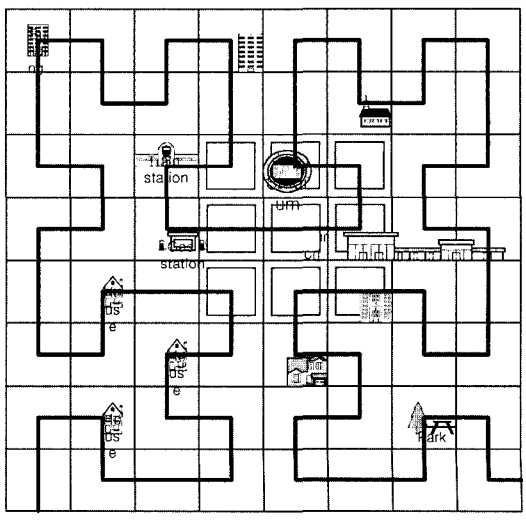


그림 10 전체 영역을 나누는 격자에 공간-채움 곡선 (Hilbert Curve) 적용

공간-채움 곡선을 사용하여 클러스터링했기 때문에 전체 영역의 격자가 인접해서 브로드캐스팅된다. 그러므로 데이타를 기다리는 이동 호스트의 구성(setup) 횟수가 줄어든다. 따라서, 이동 호스트의 전체 사용 시간 중에서 구성 시간이 줄어들기 때문에 전력을 효율적으로 사용할 수 있다.

본 논문에서 제안한 브로드캐스팅 기법의 성능은 4장에서 실험을 통해 알아본다.

4. 실 험

이번 장에서는 본 논문에서 제안한 브로드캐스팅 기법과 캐시 기법의 성능을 시뮬레이션을 통해서 평가한다. 시뮬레이션을 하기 위해서 리눅스에서 C 언어를 사용해서 프로그래밍하였다.

4.1 실험 데이타 모델(workload design)

현재 위치 의존 질의에 관한 벤치마크가 존재하지 않는다[12]. 본 논문에서는 이전 연구의 브로드캐스팅 기법/캐시 기법과 본 논문에서 제안하는 브로드캐스팅 기법/캐시 기법의 성능을 비교하기 위해서 위치 의존 질의를 처리하는 모델을 디자인하였다. 데이타를 저장하는 가장 기본적인 구성 요소인 데이타베이스, 위치 의존 질의를 처리하는 이동 호스트의 위치를 변화시키는 방법, 실험에서 사용할 위치 의존 질의를 모델링하였다.

• 데이타

데이타베이스를 구성할 때 사용하는 데이타를 말한다. 모든 데이타는 랜덤 함수를 이용해서 만들었으며, 파라미터는 아래의 표 2와 같다.

표 2 데이타 파라미터

파라미터	설명	값
ROAD_FREQ	도로의 간격	70~150
ROAD_WIDTH	도로의 폭	9
MAP_SIZE	데이타가 존재하는 영역의 크기. 전체 영역: 3200 * 3200	3200
DATA_TYPE	건물의 종류	1~10
DATA_NUM	각 타입에 따른 건물의 개수	9~131
DATA_SIZE	건물의 크기	20~150

전체 영역을 만들기 위해서 먼저 길을 일정 폭으로 생성한 후 길에 인접해서 건물이 존재할 수 있도록 건물을 생성했다. 건물의 종류는 우리 주변에서 흔히 볼 수 있는 것을 골라서 상점, 아파트, 우체국, 지하철 역, 경찰서, 학교, 병원, 공원, 소방서, 호텔(숙박 업소), 이렇게 10가지로 구분하였다. 건물을 생성할 때는 크게 특정 영역마다 있는 건물과 흔히 볼 수 있는 건물로 나누었다. 우체국, 지하철 역, 학교, 소방서, 경찰서와 같은 공공 건물은 보통 특정 영역에 한 개씩 존재하기 때문에 전체 영역을 나누어서 균등하게 분포할 수 있도록 하였다. 나머지 5가지 건물은 일정하게 분포할 필요가 없기 때문에 전체 영역에 랜덤하게 분포하도록 하였다.

• 데이타베이스

실험에서 사용한 데이타베이스는 생성한 데이타를 사용해서 한 개의 테이블 R에 저장하였다. R의 투플은 6개의 속성으로 구성되며, 투플이 가지는 속성은 표 3과 같다. 건물의 이름은 ID를 바탕으로 생성해 냈다. 전체 영역을 200m 단위로 나누었을 때 16*16개의 격자가 나오는데 각 격자마다 ID를 부여하여 ind_id에 저장하였다. 데이타베이스의 각 파라미터는 표 4에 있다.

표 3 투플의 속성

이름	설명
center	건물의 중심 좌표
id	건물의 데이타베이스 ID
name	건물의 이름
class	건물의 종류
ind_id	속하는 인덱스 영역
property	건물의 특징

표 4 데이타베이스 파라미터

파라미터	설명	값
Database_Size	데이타의 개수	520 (개)
Tuple_size	투플 1개의 크기	120 (Bytes)
index_size	격자 정보 투플의 크기	12 (Bytes)

• 이동 패턴

이동 호스트는 이동 중에 위치 의존 질의를 요구하는 것으로 가정했기 때문에 이동 호스트의 움직임을 모델링하는 것은 중요하다. 이동 패턴을 정의하기 위해서 사용한 파라미터는 표 5에 있다. 실험에서는 사용자의 이동 패턴이 3가지 경우인 것으로 가정하였다. 실험을 간단하게 하기 위해서 단방향 이동, 왕복 이동, 랜덤 이동으로 나누어서 생각하였다[12].

표 5 이동 패턴 파라미터

파라미터	설명	값
Speed	이동 호스트의 이동 속도	10~20
Scope_x, Scope_y	이동 한계 (영역 한계)	3200
Start_Loc	시작 위치	(0,0)~(0,3200) (0,0)~(3200,0)
Start_speed	이동 시작 속도	10~20
Dir	이동 방향	N, S, E, W

① 단방향 이동 : 가장 단순한 형태의 이동으로, 이동 호스트의 처음 정해진 이동 방향으로 영역을 벗어날 때까지 일정한 속도로 이동하는 형태이다.

② 왕복 이동 : 이 이동은 2단계로 나누어진다. 1번째 단계에서는 ①과 같이 단방향 이동을 하고, 2번째 단계에서 1단계에서 왔던 반대 방향 같은 이동 속도로 움직이는 운동이다.

③ 랜덤 이동 : 가장 일반적인 경우이며, 이동 호스트가 교차로에서 방향을 랜덤하게 선택하면서 이동하는 경우이다.

4.2 실험 모델(simulation model)

본 논문의 시뮬레이션 모델은 기지국 역할을 하는 서버, 이동 호스트, 호스트 간 연결을 담당하는 무선 연결로 구성된다. 서버는 데이타베이스를 유지하며, 데이타베이스 서버로 작동한다. 위치 의존 질의는 이동 호스트에서 요구되며, 캐시에 원하는 데이타가 없을 때는 서버에 질의를 전달한다.

• 서버

기지국이 담당하는 서버는 이동 호스트로부터 메시지를 받아서 처리한다. 메시지를 받아서 처리하는 것은 본 실험 모델에서 제외하였기 때문에 서버는 브로드캐스팅 스케줄을 결정하는 역할을 한다. 브로드캐스팅을 하기 위해서 그림 11과 같이 전체 영역을 격자로 나누어, 격자를 공간-채움 곡선에 따라 브로드캐스팅한다.

• 이동 호스트

직선 거리를 사용하는 캐시 기법과 맨하탄 거리를 사용하는 캐시 기법의 성능을 비교하기 위해서 이 2가지 기법 중 1가지를 사용하는 이동 호스트를 모델링하였다. 200대의 이동 호스트를 실험하였는데, 100대는 맨하탄 거리를 사용하고 나머지 100대는 직선 거리를 사용하였다.

이동 호스트는 "현재 위치에서 가장 가까운 식당을 조회하라."와 같은 이동 호스트의 현재 위치에서 가장 가까운 건물을 찾는 질의를 사용하는 것으로 가정하였다. 건물의 종류가 10가지이기 때문에 10가지 질의를 만들었다. 이동 호스트는 랜덤하게 질의를 선택해서 요구하도록 하였다.

본 논문에서 사용한 각종 파라미터는 표 6에 있다.

그림 11은 CRS_MD를 이용한 캐시 교체 알고리즘이다. 이 때 다음과 같은 변수가 사용된다.

D_i : 캐시 내에 새로 저장할 데이타

C_{cur} : 현재 캐시 내에 저장된 데이타의 총 개수

IS : 현재 캐시 내에 있는 격자 중에서 가장 큰 캐시 교체 점수를 갖는 격자

δt : 미리 정해진 작은 상수로 일정한 시간 단위

M_{fL} : 이동 호스트 M의 δt가 지난 후의 위치

M_L : 이동 호스트 M의 현재 위치

S_{CR} : 이동 호스트의 캐시 영역의 크기

4장에서 실험을 통해서 제안한 캐시 기법의 성능을 평가한다.

표 6 실험 파라미터

파라미터	설명	값
Clients	이동 호스트의 개수	200
Cache_Region_E	캐시 영역의 크기 (직선 거리)	200 800
Cache_Region_M	캐시 영역의 크기 (맨하탄 거리)	250 1002
Cache_Size	캐시의 크기	128(KBytes)
Setup_time	이동 호스트의 구성 시간	40 (ms)
Ordering	브로드캐스팅 순서 결정	Z-Order, Row-wise, Hilbert
Data_Bucket	데이타 버킷의 크기	128 (Bytes)
Index_Bucket	인덱스 버킷의 크기	16 (Bytes)
I_D_rate	인덱스와 데이타의 할당 비율	1:7
Bandwidth	무선망의 대역폭	28.8 (Kbps)

```
Cache_Management( )
{
  δt ← 미리 정해진 작은 수;
  MfL ← 이동 호스트 M의 ( ML + δt )시간에서의 위치;

  for ML ~ MfL {
    for 브로드캐스팅 중인 모든 격자 Ik {
      if( distance ( MfL, Ik) < SCR ) {
        if( Ccur < Cmax ) {
          Ik를 Cache에 삽입;
          Ik의 ID와 관련이 있는 데이터를 Cache에 삽입;
        } else {
          for Cache에 있는 격자 Iq {
            CRS_MD( Iq )를 계산;
          }
          Cache에서 Ihigh를 찾는다;
          Cache에서 Ihigh를 제거;
          Ihigh의 ID와 관련이 있는 데이터 Cache에서 제거
          Ik를 Cache에 삽입;
          Ik의 ID와 관련이 있는 데이터를 Cache에 삽입;
        } /* end of if( Ccur ) */
      } /* end of if( ML ) */
    } /* end of for( Ok ) */
  } /* end of for( ML ) */
}
```

그림 11 도심 환경을 고려한 캐시 교체 알고리즘

4.3 실험 결과

본 논문에서 제안한 브로드캐스팅 기법과 캐시 기법을 평가하기 위해서 3가지를 실험한다. 첫째, 위치 의존 질의를 처리하는 환경에서 여러 가지 브로드캐스팅 기법의 성능을 실험한다. 둘째, 도심에서의 위치 의존 질의를 처리하는 경우에 제안한 캐시 기법의 성능을 실험한다. 셋째, 제안한 캐시 기법을 지원하는 시스템의 계산량을 평가한다. 성능을 평가하기 위해서 공간-채움 곡선과 캐시 영역의 크기를 변화시키면서 실험하였다.

브로드캐스팅 기법의 성능을 평가하기 위해서 이동 호스트의 구성 시간을 사용하였다. 2차원 공간의 데이타를 이동 호스트의 전력 소모를 고려한 브로드캐스팅 기

법이므로 이동 호스트의 구성 시간을 사용해서 브로드캐스팅 기법의 성능을 평가하였다. 제안한 캐시 기법의 성능을 평가하기 위해서 캐시의 결과의 최적성과 거리 계산 시간을 이용해서 성능을 평가한다.

4.3.1 브로드캐스팅 기법

이동 호스트의 구성 시간이 클러스터링의 지표가 될 수 있기 때문에 구성 시간을 평가 지표로 사용한다. 그림 14는 Row-wise, Z-Order, Hilbert 곡선의 순서로 각각 브로드캐스팅했을 경우 캐시 영역의 크기에 따른 이동 호스트의 구성 시간을 보여준다.

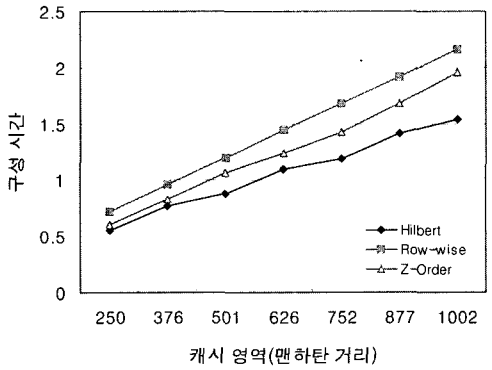


그림 12 브로드캐스팅 방법에 따른 이동 호스트의 평균 구성 시간

Row-wise 곡선은 캐시 영역의 가로 혹은 세로의 격자 1개가 증가할 때마다 이동 호스트의 구성 시간이 setup time 만큼 늘어나는데 비해서, 다른 곡선은 구성 시간이 선형으로 증가하지 않기 때문에 그림 13의 Hilbert와 Z-Order 곡선을 사용한 경우 이동 호스트의 구성 시간의 증가 비율이 일정하지 않다.

그림 12을 보면 Row-wise, Z-Order, Hilbert 곡선 중 Hilbert 곡선의 순서로 브로드캐스팅한 경우에 이동 호스트가 데이타를 받기 위해 사용하는 전력을 가장 적게 소비한다는 것을 알 수 있다. 그러므로 위치 의존 질의를 처리하는 이동 호스트를 효과적으로 지원하기 위해서는 Hilbert 곡선을 사용해야 한다.

본 논문에서 가정한 환경에서 필요한 데이타 버킷 사이에 불필요한 데이타 버킷 1개가 끼어있다면 이동 호스트의 구성 시간을 고려해 볼 때 불필요한 데이타 버킷까지 받아서 버리는 것이 전력을 적게 사용한다. Z-Order 곡선을 사용하여 브로드캐스팅하면 각 버킷의 브로드캐스팅 거리가 2를 넘는 경우가 많지만, Hilbert 곡선을 사용하여 브로드캐스팅하면 버킷의 브로드캐스팅 거리가 2이하인 경우가 많기 때문에 Z-Order 곡선을 사용한 경우보다 좋은 성능을 보인다.

4.3.2 캐시 기법

• 결과의 정확한 정도

그림 13은 본 논문에서 실험한 2가지 거리 계산 방법의 최적 결과 반환 비율을 보여 준다. 최적 결과 반환 비율을 계산하기 위해서 각 질의에 대해 캐시에서 반환된 값과 이동 호스트에서 가장 가까운 값을 비교해서 일치하는가를 확인하였다. 캐시 영역이 250일 때를 제외하고는 맨하탄 거리를 사용한 이동 호스트가 직선 거리를 사용한 이동 호스트보다 최적의 결과를 반환하였다. 맨하탄 거리를 사용하는 것이 도심에서는 더 적합하다는 것을 알 수 있다.

도심에서는 모든 도로가 서로 직교하고, 도로를 따라 이동 호스트가 이동하기 때문에 맨하탄 거리를 사용하면 이동 거리를 거의 정확하게 파악할 수 있는 것으로 본다.

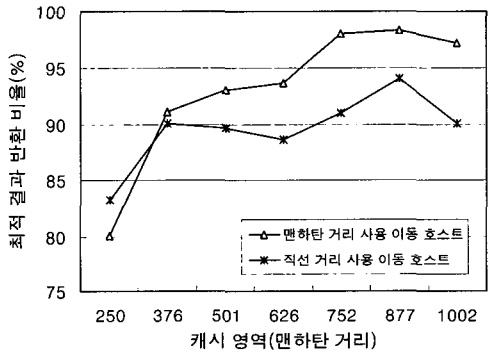


그림 13 맨하탄 거리를 사용하는 이동 호스트와 직선 거리를 사용하는 이동 호스트 최적 결과 반환율

• 거리 계산 부하량

직선 거리 계산식과 맨하탄 거리 계산식의 계산 량을 비교하기 위해서 다른 데이타를 가지고 같은 계산을 여러 번 실행하였다. 각 연산의 전체 실행 시간을 비교하

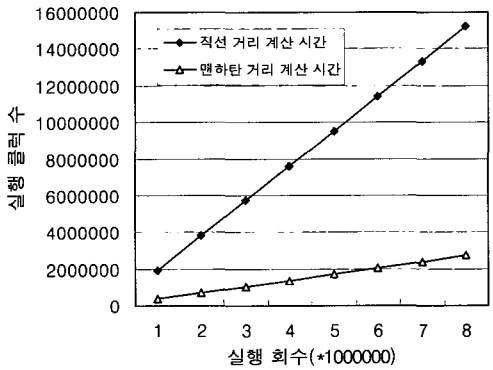


그림 14 직선 거리 계산식과 맨하탄 거리 계산식의 클럭 수 비교

기 위해서 연산의 시작 시간과 끝 시간을 이용해서 실행 시간을 계산하였다.

그림 14와 같이 맨하탄 거리를 계산하는 식이 약 78배정도 빠르다. 따라서 맨하탄 거리를 이용해서 거리를 계산하면 캐시 결과가 정확할 뿐만 아니라 시스템이 거리 계산을 위해 사용하는 리소스의 78배 정도를 아낄 수 있다.

5. 결론 및 향후 연구 과제

본 논문에서는 위치 의존 질의를 효과적으로 처리하기 위하여 브로드캐스팅 기법을 제안하고, 제안한 브로드캐스팅 기법과 함께 사용하여 도심에서 데이타를 효과적으로 캐시할 수 있는 캐시 기법을 제안하였다.

본 논문에서는 위치 의존 질의를 처리하는 이동 호스트를 효과적으로 지원하기 위해서 전체 영역을 격자로 나누고 격자를 공간-채움 곡선의 순서로 브로드캐스팅하는 기법을 제안하였다. 여러 가지 공간-채움 곡선 중에서 Hilbert 곡선의 순서로 브로드캐스팅하였을 때 이동 호스트의 전력 소모가 가장 적었다.

본 논문에서는 도심 환경의 특징을 연구하고 도심 환경에서 위치 의존 질의를 효과적으로 처리할 수 있는 캐시 기법을 제안하였다. 제안한 캐시 기법을 사용하면 이동 호스트에서 가장 가까운 데이타를 반환하므로 사용자가 거리를 예측하기 쉬울 것이다.

이동 단말기의 발달로 이동 컴퓨팅 환경은 우리 실생활에서 많이 응용되고 있다. 특히 사용자의 현재 위치를 고려한 서비스는 일반인들에게 가장 유용한 서비스 형태이다. 현재까지 연구된 이동 컴퓨팅 환경에서 위치 의존 질의 지원 시스템의 성능은 부분적인 시스템의 성능 향상을 가져왔는데, 본 논문에서 제안한 브로드캐스팅 기법과 캐시 기법을 사용하면 전체적인 성능 향상을 가져올 수 있을 것으로 본다.

본 논문에서는 2*2 격자를 기본으로 하는 공간-채움 곡선을 사용했는데, 각 캐시 영역의 크기에 잘 적응할 수 있는 변형 공간-채움 곡선에 대한 연구를 통해서 이동 호스트의 특징을 잘 지원할 수 있는 브로드캐스팅 기법의 연구가 필요하다. 본 논문에서는 도심 환경에서 도로가 항상 정동, 정서, 정남, 정북을 향한다고 가정하고 있는데, 도로의 방향이 바뀔 때 그에 대해 적응할 수 있는 방법을 연구할 필요가 있다. 그리고 한 개의 셀 뿐만 아니라 여러 셀을 움직이는 이동 호스트를 지원하는 브로드캐스팅 기법과 캐시 기법을 연구하면 위치 의존 질의를 효과적으로 처리할 수 있을 것으로 예상한다.

참 고 문 헌

[1] Jin Jing, Abdelsalam Helal, Ahmed Elmagarmid, "Client-Server Computing in Mobile Environments," ACM Computing Surveys, Vol. 31, No. 2, pp. 117~157, 1999.

[2] D. Barbara, "Mobile Computing and Databases - A Survey," IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 1, 1999.

[3] S. Acharya, "Broadcast Disks : Data Management for Asymmetric Communications Environments," ACM SIGMOD '95, pp. 199~210, 1995.

[4] T. Imielinski, S. Viswanthan, "Wireless Publishing : Issues and Solutions," Mobile Computing, Kluwer Academic Publishers, pp. 299~329, 1996.

[5] A. Datta, D.E. Vandermeer, A. Celik, V. Kumar, "Broadcast Protocols to Support Efficient Retrieval from Databases by Mobile Users," ACM Transactions on Database Systems, Vol. 24, No. 1, 1999.

[6] T. Imielinski, S. Viswanthan, B.R. Badrinath, "Data on Air : Organization and Access," IEEE Transactions on Knowledge and Data Engineering, Vol 9, No 3, pp. 353~372, 1997.

[7] Boris Y. Chan, Antonio Si, Hong V. Leong, "Cache Management for Mobile Databases : Design and Evaluation," IEEE CS Data Engineering, pp. 54~63, 1998.

[8] Hong V. Leong, Antonio Si, "On Adaptive Caching in Mobile Databases," proceeding of ACM *symposium on* Applied computing, pp. 302~309, 1997.

[9] Margaret H. Dunham, Vijay Kumar, "Location Dependent Data and its Management in Mobile Databases," Mobility in Database and Distributed System, 1998.

[10] S. Y. Seydim, M. H. Dunham, V. Kumar, "Location Dependent Query Processing," ACM MobiDE 2001, pp. 47~53, 2001.

[11] 김 호숙, 용 환승, "이동 데이타베이스 시스템에서 데이타의 위치와 영역 특성을 고려한 캐쉬 교체 기법", 정보과학회논문지: 데이타베이스 제27권 제1호, pp. 53~63, 2000.

[12] Qun Ren, Margaret H. Dunham, "Using Semantic Caching to Manage Location Dependent Data in Mobile Computing," ACM MobiCom '00, pp. 210~221, 2000.

[13] H. V. Jagadish, "Linear clustering of objects with multiple attributes," ACM SIGMOD '90, pp. 332~342. 1990.

정 일 동

2000년 부산대학교 전자계산학과(이학사). 2002년 부산대학교 대학원 전자계산학과(이학석사). 2002년~부산대학교 대학원 전자계산학과 박사과정. 2003년 11월~LG전자 DAC연구소 주임 연구원. 관심분야는 정보가전, 인터넷 컴퓨팅, 임베디드 시스템 등

유 영 호

1994년 부산대학교 전자계산학과(이학사). 1997년 부산대학교 전자계산학과(이학석사). 2003년 부산대학교 전자계산학과(이학박사). 2004년~현재 부산대학교 컴퓨터및정보통신연구소 기금교수. 관심분야는 XML, Mobile Computing, P2P

이 종 환

1996년 부산대학교 전자계산학과(이학사). 1998년 부산대학교 전자계산학과(이학석사). 2003년 부산대학교 전자계산학과(이학박사). 2004년 9월~현재 (주)오토파워 부설연구소 선임 연구원. 관심분야는 데이타베이스, 인터넷응용, 스마트 센서

김 경 석

1977년 서울대학교 무역학과(경제학사) 1979년 서울대학교 전자계산학과(이학석사). 1988년 일리노이 주립대(어바나-샴페인) 전자계산학 박사. 1988년~1992년 미국 노스다코타 주립대학교 전자계산학과 조교수. 1992년~현재 부산대학교 전자전기정보컴퓨터공학부 교수. 관심분야는 데이타베이스, 멀티미디어, 한글/한말 정보처리, 인터넷 컴퓨팅 등