연관 규칙을 이용한 이동 호스트의 선반입 알고리즘

The prefetch algorithm of a mobile host using association rule

김호숙 (Ho-Sook Kim), 용환승 (Hwan-Seung Yong)

이화여자대학교 컴퓨터학과
Dept. of Computer Science and Engineering, Ewha Womans University

요 약

최근 이동 컴퓨팅 환경에서 위치 기반 서비스를 지원하는 다양한 응용이 증가하고 있다. 본 논문은 이동 컴퓨팅 환경에서 대량의 공간 데이터베이스를 기반으로 효율적인 정보 서비스를 제공하기 위한 새로운 연관성 기반 선반입 알고리즘인 STAP을 제안한다. STAP은 이동 환경에서의 위치 기반 질의가 갖는 시-공간적 연관성을 이용한다. 또한 사용자의 이동 특성과 공간 데이터의 중요도를 함께 고려한다. STAP은 기존의 캐쉬 정책에서 고려하지 못했던 서비스 사이의 연관성이라는 새로운 측면을 고려함으로써 캐쉬 정책의 차원을 한 단계 높이는 계기가 되었다. 또한 성능 평가를 통하여 제안된 선반입 알고리즘의 효율성을 입증하였다.

키워드 – 이동 컴퓨팅, 위치 기반 서비스, 선반입 알고리즘, 연관 규칙, 추천 알고리즘

Abstract

Recently, location-based services are becoming very popular in mobile environments. In this paper, we propose a new association based prefetch algorithm (called by STAP) that efficiently supports information service based on the large quantity of spatial database in mobile environments. We apply the spatial-temporal relations that are meaningful for location-based queries in mobile environments. Moreover, STAP considers user's mobility and the weight of spatial data. The relation of services is a new aspect not considered in previous cache policies. So STAP is the first prefetch algorithm considering the spatial-temporal relations and thus the cache policy begins to gain a new dimension. We evaluate the performance of STAP and prove the efficiency of STAP.

Keyword – mobile computing, location-based services, prefetch algorithm, association rule, recommendation algorithm

1. 서론

무선 네트워크 기술과 이동 정보 기기의 발전은 컴퓨팅 환경에 있어서 이동 컴퓨팅이라는 새로운 패러다임을 낳았다. 이동 컴퓨팅 환경 (Mobile Computing Environments)이란 랩탑이나 노트북 등의 이동 단말기를 이용하는 사용자가 무선 통신을 통해서 물리적인 위치와 무관하게 서버 컴퓨터의 자원을 활용하여 작업하는 환경을 의미한다[1,2]. 최근 이동 장비의 소형화, 고성능화와 함께 무선 통신의 기반 구축에 힘입어, 작업 위치의 제약 없이 항상 최신의 정보를 얻고, 이를 통해 좀 더 높은 차원의 정보 서비스를 창출하고자 하는 사용자들의 요구가 높아지면서 이동 컴퓨팅의 시장은 점차 확대되고 있다[3]. 특히 PDA나 휴대전화 또는 소형 이동 장비를 이용하여 해당 지역의 지도 정보를 얻거나 웹 사이트를 접속하는 등의 위치 기반 서비스(LBS: Location-based Services)가 빠르게 확대되고 있다. 또한 초소형 GPS(Global Positioning System)에 대한 기술 개발은 차량용 항법 장치나 물류 추적 장치 등 다양한 응용 서비스의 개발을 가능하게 하였고 앞으로도 지역의 교통이나 뉴스, 사용자와 가장 가까운 특정 서비스를 찾는 응용이나, 사용자의 위치를 고려한 모바일 광고 등으로 응용 범위를 넓혀갈 것이다.

그러나 이동 컴퓨팅 환경은 좁은 대역폭과 잦은 단절 및 제한된 전력 소모 등과 같은 제약을 갖는다[2]. 본 연구는 이동 컴퓨팅 환경이 갖는 제약사항에 효율적으로 대처하면서 LBS를 제공하는 응용 시스템에서 이동 사용자의 질의에 효과적으로 대응하기 위하여, 서비스들 사이에 존재하는 관련성을 이용한 연관성 기반 선반입 알고리즘인 STAP (Spatial & Temporal Association based Prefetch algorithm)을 제안한다. 이때 기반이 되는 서비스의 관련성은 서비스를 요청하는 질의들의 시간적 연관 관계와 이러한 서비스를 제공해 주는 공간상의 오브젝트들의 실제 거리를 고려하여 연관 규칙 (association rule)과 클러스터링 (clustering) 등의 마이닝 기법을 이용하여 생성된다. 이동 컴퓨팅 환경에서 캐쉬 관리 알고리즘은 크게 서버로부터 연관성 있는 아이템을 선정하여 비용이 저렴한 다운 로드 시에 함께 가져오는 선반입에 대한 부분과 이동 호스트의 캐쉬 내에서 희생자를 선정하는 캐쉬 교체 부분으로 구성된다. 본 논문에서는 지금까지의 일반적인 캐쉬 정책과 달리 선반입과 캐쉬 교체 두 단계에 각각 다른 아이템 선정 기준을 적용하는 보다 효과적인 캐쉬 정책을 제안하고 성능평가를 통해 제안한 STAP의 효율성을 입증하였다.

본 논문의 구성은 다음과 같다. 1절의 서론에 이어 2절에서는 캐쉬와 위치 기반 서비스에 대한 관련 연구들을 소개하고, 3절에서는 시스템 구성과 마이닝 기법을 이용한 연관 클래스 집합 생성에 대하여 소개한다. 4절에서는 연관성 기반 선반입 알고리즘인 STAP에 대해 제안하고, 5절에서는 제안한 알고리즘에 대한 의미 있는 성능평가를 수행하기 위하여 이동 컴퓨팅 환경에 적합한 성능평가 환경을 구현하고, 성능평가를 통하여 STAP의 효율성을 입증한다. 마지막으로 6절에서는 결론을 맺는다.

2. 관련연구

이동 컴퓨팅 환경은 무선 통신의 좁은 대역폭에 의해 전송 데이터 양의 제약을 받고, 전송 매체의 외부 간섭에 의한 잦은 단절이 발생할 수 있다. 또한 이동 호스트는 시간에 따라 셀 간을 이동하면서 서로 다른 기지국 간의 연결에 의해 트랜잭션을 지속하며, 전력 소모의 제한을 갖는다[2]. 이러한 특징을 고려하여 기지국에서 데이터를 전송할 때 자주 요구되는 데이터를 방송(broadcast)을 통해 전달함으로써 이동 호스트가 데이터를 요구하는데 필요한 에너지 소모를 줄이고 사용자 수의 증가에 효율적으로 대처하는 방법이 제안되었다[4]. 이와 함께 기지국에서 방송되는 데이터 중에서 향후 사용 가능성이 높은 데이터를 이동 호스트의 캐쉬에 저장 하였다가 이후 사용자의 질의에 응답할 때 이용하는 캐쉬 기법이 도입되었다[5]. 이동 호스트의 운영에서 향후 필요로 하는 데이터에 대해 예측하는 것은 이동 컴퓨팅 환경의 특성상 중요한 문제가 된다. 즉 서버 측에서는 무선 네트워크를 통해 전송하기 위하여 미리 필요한 데이터를 무선용으로(wireless-friendly format) 변화시킬 시간을 확보할 수 있고, 이동 호스트 측에서는 업 로드(up-load)에 비해 낮은 비용이 드는 다운 로드(down-load) 시에 관련성 있는 데이터를 미리 선반입 해 두거나, 방송되는 데이터 중에서 연관된 데이터를 선반입 함으로써 시간과 비용을 줄일 수 있다. 또한 사용자들의 요구에 부합하는 데이터 뿐 아니라 서버에서 관련 데이터를 추천(recommendation)하는 방법을 통해서 사용자의 편리성과 효율성을 높일 수 있다.

이동 호스트의 캐쉬에 대한 연구는 그 중요도에 비해 활발한 연구가 진행되지 못하고 있는 것이 현실이다[7]. 최근까지 이루어진 연구를 정리하면 초기에는 데이터의 접근 시간이나 수정 빈도 등을 기준으로 하는 연구들이 수행되었다[1,2]. 가장 일반적인 캐쉬 교체 방법인 LRU는 접근 시간을 기준으로 가장 오래 전에 접근된 오브젝트를 맨 먼저 교체의 대상인 희생자로 선정하는 방법이다. 이후에 이동 호스트의 움직임의 특성을 고려한 방법들이 제안되었는데 그 중에서 주목할 만한 연구로 Ren 등은 이동 호스트의 위치와 속도, 방향 등을 고려한 의미적 캐쉬 구조 (semantic cache scheme)인 FAR 알고리즘[6]을 제안하였다. 또한 Zheng 등은 아이템의 영역, 접근 시간, 방향과 거리를 모두 고려한 캐쉬 교체 방법으로 PA와 PAID(Probability Area Inverse Distance) [7]를 제안하였다. PAID에서는 접근 시간이 오래되고, 영역의 크기가 작으며 거리가 먼 오브젝트가 먼저 희생자로 선정된다.

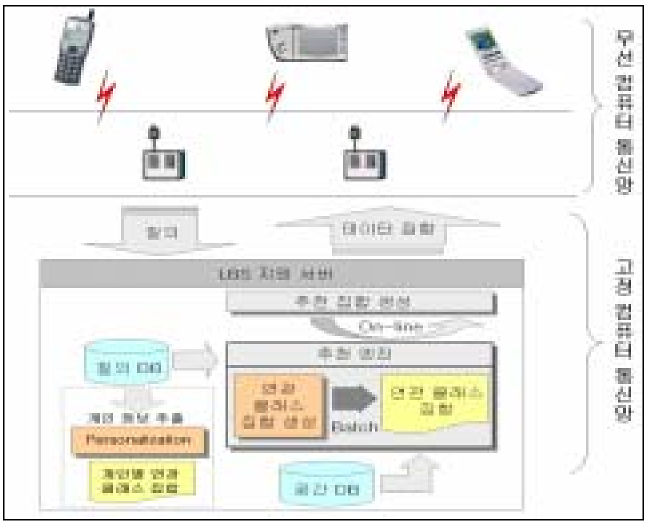
최근 이동 컴퓨팅 응용 프로그램이 다양해지면서 이를 지원하기 위하여 마이닝 기법들이 도입되기 시작하였다. Peng [8]등은 분산 환경에서 각 서버에 적당한 데이터를 할당하기 위하여 사용자의 이동에 대한 로그 파일을 이용하여 사용자의 움직임과 접근되는 오브젝트 속성을 고려한 패턴을 마이닝하고, 그 결과를 각 서버에 대한 데이터 할당에 이용하는 연구를 수행하였다. Phatak [9]등은 이동 사용자의 웹 페이지의 접근에 대한 성능 향상을 위하여 클러스터링 방법을 이용하여 관련 웹사이트 찾아 모바일 용으로 미리 바꾸어 놓았다. Morimoto [10]는 공간 데이터베이스에 존재하는 오브젝트들 중에서 자주 이웃하는 그룹을(frequent neighboring class set) 효과적으로 찾는 연구를 수행하였다.

위치 기반 서비스와 관련된 국내의 연구들로는 백형종[11] 등이 이동 컴퓨팅 환경에서 위치 기반 질의를 지원하는 이동 트랜잭션 모델을 제시하였고, 류근호[12]등은 협업 필터링 (CF : collaborative filtering), 개인화 기술 (personalization) 및 데이터 마이닝 기법을 바탕으로 하는 개인화 추천 시스템의 아키텍쳐를 제시하였다. 그밖에 서영덕[13]등은 실제 인터넷 GIS를 제공하고 있는 홈페이지의 질의 결과를 바탕으로 사용자의 요구를 분석하여 사용자가 원하는 지역과 검색어 등에 대한 통계 결과를 제시하였다.

그러나 지금까지 이동 환경 하에서 위치 기반 서비스를 수행하는 응용을 기반으로 사용자의 질의가 갖는 시간적, 공간적 특징을 함께 고려하여 선반입에 적용한 연구는 수행되지 못하였다. 본 연구는 사용자의 질의들 사이의 존재하는 시간적 연관 관계를 고려하면서, 동시에 질의의 결과가 되는 공간 오브젝트들 사이의 공간상의 거리가 갖는 특징을 마이닝 방법을 통해 발견하고 이를 이용하여 효율적으로 이동 컴퓨팅 환경에서 위치 기반 서비스를 지원하기 위한 새로운 선반입 알고리즘을 제안한다.

3. 시스템 구성

논문의 기술을 위하여 다음의 몇 가지를 가정한다. 첫째 이동 컴퓨팅 환경은 이동 기지국(Mobile Support Station)과 이동 호스트(Mobile Host 또는 Mobile Unit)로 구성된다. 이동 기지국은 고정 네트워크에 연결된 호스트들 중에서 이동 호스트에게 무선 통신 인터페이스를 지원하는 것으로서 데이터베이스 서버의 역할도 담당한다. 둘째 우리는 위치가 2차원 좌표로 표현되는 지리적 위치 모델(geometric location model)을 이용하며, 이동 호스트는 GPS와 같은 시스템을 이용하여 자신의 위치 정보를 알고 있다고 가정한다. 본 논문에서 사용하는 서버의 데이터베이스에 저장된 데이터의 레코드는 모두 동일한 고정 크기(fixed size)이며 수정(update)이 없는 읽기 중심(read-only) 질의만 수행된다고 가정한다.



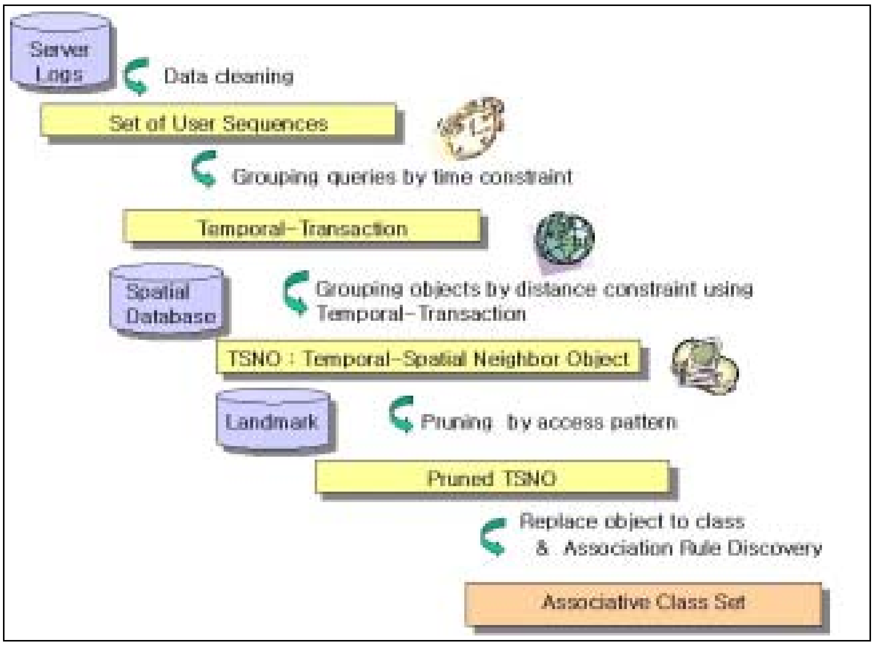
[그림 1] 시스템 구성도

본문에서 구상하는 시스템 구성도는 그림 1과 같다. 이동 사용자가 자신의 캐쉬에 없는 정보를 얻기 위해 무선 컴퓨터 통신망을 통하여 서버에 접속하면 해당 질의를 받은 서버는 사용자의 질의를 질의 데이터베이스에 저장하고, 공간 정보를 갖고 있는 공간 데이터베이스를 이용하여 질의에서 요청되는 데이터를 이동 호스트에게 전달한다. 이때 본 논문에서 제안하는 연관성을 이용한 선반입 알고리즘에 의해 관련성이 높은 데이터를 추천 집합으로 생성하여 함께 이동 호스트로 전달하기 위하여 LBS 지원 서버는 크게 3가지 구성 요소를 포함한다. 첫번째는 질의 데이터베이스에 적재되어 있는 로그 데이터에 마이닝 기법을 적용하여 연관 클래스 집합을 생성하는 단계이며 이는 일정 기간마다 일괄 작업(batch job)으로 처리된다. 둘째는 생성된 서비스 간의 연관 클래스 집합 정보를 이용하여 사용자의 질의에 따라 추천 집합을 생성하는 단계로 이는 사용자의 요구 사항이 발생하는 시점에 따라 실시간(on-line)으로 처리되어진다. 마지막으로 구성도에서 점선으로 표시된 개인 정보를 이용한 개인별 연관 클래스 집합 생성은 향후 추가되어야 하는 부분으로 개별화된 (personalized) 연관 관계를 이용하면 보다 정교한 추천 작업이 이루어 질 수 있을 것이다. 연관 클래스 집합을 생성하는 단계는 3.1절에서, 생성된 연관 클래스 집합을 이용하여 추천 집합을 생성하는 과정은 4절에서 상세히 설명하도록 한다.

3.1 연관 규칙을 이용한 서비스의 연관 클래스 집합 생성

이동 호스트의 캐쉬 선반입 정책에서 가장 중요한 고려점은 향후 사용자가 어떤 정보를 요청할 것인가를 예측하는 것이다. 이를 위하여 본 연구에서는 이동 호스트들의 질의들의 시간적, 공간적 특징을 분석하여 자주 함께 질의 되는 서비스들의 집합 (frequent service set)인 연관 클래스 집합(Associative class set)을 생성하고, 이를 캐쉬 선반입의 오브젝트 선발 기준으로 삼는다. 연관 규칙과 클러스터링 알고리즘을 이용하여 질의들의 연관 클래스 집합을 찾는 연구는 [14]에 의해 수행되었고 이를 간단히 정리하면 다음과 같다. 서버의 질의 데이터베이스에 저장되어 있는 질의 로그(server query log)를 대상으로 전처리(preparation), 변형(transformation), 패턴 발견 (pattern discovery)등의 마이닝 단계를 거쳐서 연관 클래스 집합을 생성한다. 먼저 시간별로 쌓여진 서버의 질의 로그를 사용자별로 정리하여 시간 순서로 정렬된 사용자별 시퀀스를 구성한다. 이때 시퀀스란 트랜잭션 시간을 기준으로 정렬된 트랜잭션의 집합이 된다. 다음, 사용자의 시퀀스를 대상으로 질의가 발생된 시간이 일정 범위 내에 있는 것을 하나의 시간-트랜잭션으로 분리한다. 즉 분석자에 의해 의미 있는 시간의 범위로 정해진 시간 간격(TW: time window)을 기준으로 TW 내에 요청 된 질의들을 시간적 연관성이 있다고 구분하여 분리한다. 다음으로는 생성된 시간-트랜잭션 그룹에서 질의 되는 오브젝트들이 분포하는 공간적 거리가 가까운 것들을 모아서 시-공간 이웃집합을(TSNO: Temporal-spatial neighbor object) 생성한다. 이 경우 일정 시간 간격 내에 요청 되는 질의들 중에서 질의의 결과를 나타내는 오브젝트들끼리 공간적으로 가깝게 위치한 질의들은 공간적으로 거리가 먼 오브젝트들을 요청한 질의에 비해서 연관도가 높다는 가정을 기반으로 질의의 결과가 되는 오브젝트의 거리가 일정 거리 한도

(threshold) 이내에 존재하는 것들을 묶어서 시-공간 이웃집합을 생성한다. 다음으로 생성된 시-공간 이웃집합 중에서 표지물(landmark)에 해당하는 오브젝트를 제거한다. 이는 보통 공간 질의를 요청할 때 원하는 목적 오브젝트를 조회하면서 해당 지역의 특징적인 건물 즉 교육기관이나 행정 기관 등의 표지물을 함께 조회하는 경향이 있다[13]는 연구 결과를 반영한 것이다. 즉 순수하게 서비스들 사이의 연관성을 갖는 오브젝트만을 기준으로 연관 클래스 집합을 찾기 위하여 시-공간 이웃 집합 중에서 서비스의 연관 관계가 없는 표지물 오브젝트를 제거한다. 한 지역 내에 존재하는 표지물 오브젝트를 결정하는 방법은 해당 지역의 지리 정보 전문가에 의해 선택되거나 로그 분석을 통해 자동으로 결정 될 수 있다. 정제된 시-공간 이웃 집합을 이용하여 연관 관계 생성 마이닝 알고리즘을 수행하여 연관 클래스 집합을 생성한다. 이렇게 생성된 연관 클래스 집합은 이동 환경 하에서 사용자에게 지리 정보를 서비스하는 응용이 갖는 특징인 서비스의 시간적 연관성과 공간적 연관성 그리고 질의 패턴의 특성을 고려한 연관성 있는 클래스의 집합이 된다. 연관 클래스 집합의 생성 단계를 그림으로 나타낸 것은 아래 그림 2와 같다.



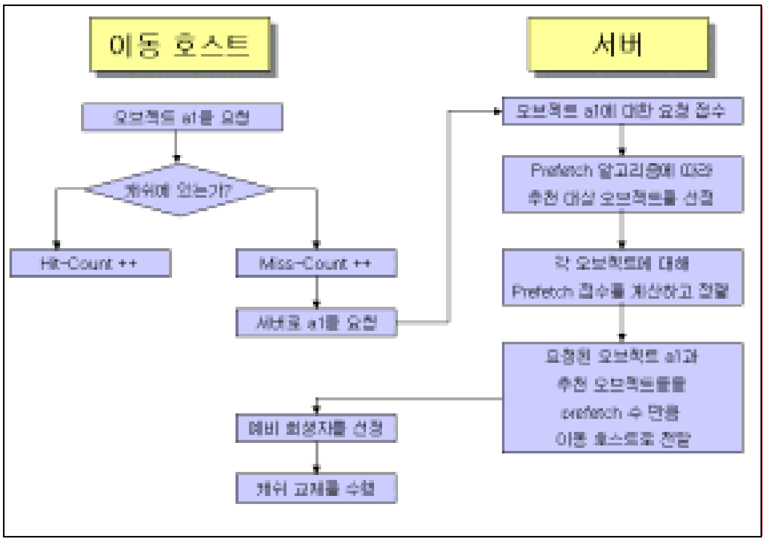
[그림 2] 연관 클래스 집합 생성 단계도

4. 연관성 기반 선반입 알고리즘

본 장에서는 3.1절에서 생성된 연관 클래스 집합을 기반으로 새로운 연관성 기반 선반입 알고리즘인 STAP (Spatial & Temporal Association based Prefetch algorithm)을 제안한다. STAP은 기존의 캐쉬 알고리즘에서 고려했던 사용자의 위치와 이동성, 대상 오브젝트의 가중치 뿐 만 아니라 질의되는 오브젝트들 사이의 시간적, 공간적 연관성을 고려한다.

4.1 선반입과 캐쉬 교체의 과정

이동 컴퓨팅 환경에서 사용자가 요청한 질의를 수행하는 캐쉬 처리 과정은 그림 3과 같다. 먼저 이동 호스트에서 발생된 질의가 오브젝트 a_1을 요청할 때 이동 호스트는 해당 오브젝트가 자신의 캐쉬에 있는가를 검사한다. 만약 있다면 hit-count를 높이고, 캐쉬 내의 오브젝트 a_1의 접근 시간을 현재의 시간으로 변경하면 된다. 만약 요청하는 오브젝트가 캐쉬 내에 없는 경우 miss-count를 증가하고 오브젝트의 정보를 얻기 위해 서버로 연결을 시도한다. 서버측에서 a_1에 대한 요청을 받으면 a_1과 함께 향후 접근 가능성이 높은 다른 오브젝트들을 선반입 알고리즘에 따라 추천 대상 오브젝트들로 선정한다. 선정된 각각의 추천 대상 오브젝트들에 대해 선반입 점수를 계산한 후 정렬한다. 마지막으로 요청된 오브젝트 a_1과 추천 대상 오브젝트들을 정해진 선반입 수 만큼 이동 호스트로 전달한다. 이동 호스트는 캐쉬 내에 선반입 될 오브젝트들을 저장하기 위한 공간을 확보하기 위해 캐쉬 교체 알고리즘에 따라 희생자 선정 처리를 수행한다.



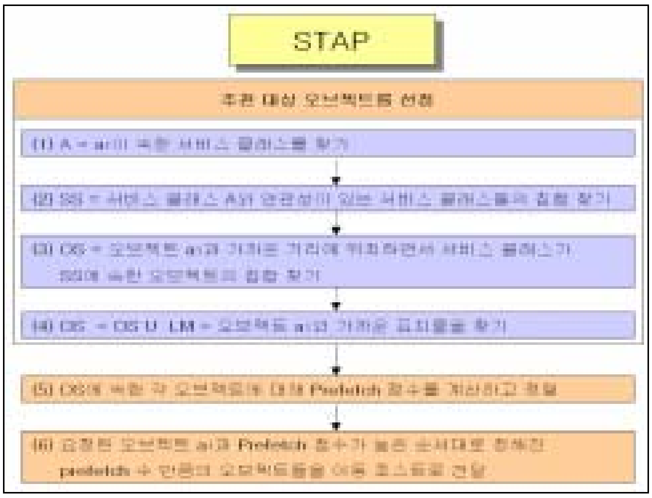
[그림 3] 캐쉬 교체 과정

이동 호스트에서 새로운 오브젝트를 캐쉬로 선반입 하기 위해서는 현재 보유하는 캐쉬의 오브젝트 중에서 가장 향후에 사용될 가능성이 가장 낮은 것을 찾아 희생자로 선정하는 캐쉬 교체 과정이 함께 수행 되어야 한다. 본 논문에서는 선반입 알고리즘의 성능을 평가하기 위하여 함께 사용되는 캐쉬 교체 알고리즘을 2가지로 정의하고 실험하였다.

첫번째, 가장 일반적인 캐쉬 교체 방법인 LRU 는 접근 시간을 기준으로 가장 오래 전에 접근된 오브젝트를 맨 먼저 교체의 대상인 희생자로 선정하는 방법이다. 둘째 방법은 공간 데이터에 대한 의미 기반 캐쉬 교체 방법인 FAR 로[6] 캐쉬 교체 시 사용자의 이동 방향과 속도 그리고 오브젝트와의 거리를 기준으로 캐쉬 교체를 수행한다. 즉 사용자의 이동 방향과 반대 방향에 있고 거리가 먼 오브젝트를 먼저 희생자로 선정한다.

4.2 선반입 알고리즘

지금까지 이동 환경에서 공간 데이터베이스를 이용하여 LBS를 수행하는 응용을 위한 선반입에 대한 연구는 거의 수행되지 못하였다. 본 논문에서는 성능 평가에서 사용할 선반입의 방법을 크게 2가지로 제안한다. 첫째는 FAR와 같은 기존의 위치 기반 캐쉬에서 고려한 사용자의 위치와 오브젝트의 거리, 이동 방향을 고려하여 선반입을 수행하는 방법이다. 즉 현재 사용자의 위치로부터 진행 방향에 위치하는 오브젝트 중에서 거리가 가까운 것을 우선적으로 선반입 하는 방법으로 이를 General-Prefetch로 명명한다. 이 방법은 사용자의 위치와 가까이 있는 것이 향후 접근 가능성이 높을 것이라는 일반적인 가정에 기반을 둔다. 둘째는 본 논문에서 제안한 STAP으로 연속되는 질의들의 시-공간적 연관 관계와 공간 데이터 질의에 대한 응답 시 고려되어야 하는 표지물 정보, 사용자와 오브젝트와의 거리와 방향, 오브젝트 자체의 중요도를 모두 고려한 선반입 방법이다. STAP의 각 상세 단계는 그림 4와 같다.



[그림 4] STAP의 수행 단계

오브젝트 a_1이 서버로 요청되었다고 가정할 때, STAP을 이용하여 a_1과 함께 추천할 대상 오브젝트들을 선정하는 과정은 다음과 같다. 첫째, a_1이 속한 서비스 클래스를 찾는다. 둘째, a_1이 속한 서비스 클래스가 A라 할 때 A와 연관성이 있는 다른 서비스 클래스들을 3.1절에서 생성한 연관 클래스 집합을 기준으로 찾아서 서비스 클래스 집합 SS(Set of Service class)을 생성한다. 셋째, 오브젝트 a_1과 가까운 거리에 위치하는 오브젝트 중에서 서비스 클래스가 SS에 속한 오브젝트들의 집합을 찾아서 추천 대상 오브젝트 집합인 OS(Object Set)로 정한다. 넷째, 오브젝트 a_1과 가까운 위치에 있는 표지물(LM: Landmark)을 찾아서 OS에 추가한다. 이렇게 생성된 추천 대상 오브젝트 집합에 대하여 각각의 선반입 점수(prefetch_score)를 계산하여 정렬한다. STAP에서 사용된 선반입 점수 계산 방법은 아래의 식1과 같다.

(식 1)

prefetch_score (a_1, o_1) =

$(c_1 * \text{association\_score}) + (c_2 * \text{weight\_score}) + (c_3 * \text{distance\_score})$;

즉 오브젝트 a_1이 요청되었을 때, 다른 오브젝트 o_1의 선반입 점수는 두 오브젝트 a_1과 o_1의 연관성 점수와 (association_score), 오브젝트 o_1의 중요도 점수 (weight_score), 그리고 두 오브젝트 a_1과 o_1의 거리 점수를 (distance_score) 이용하여 산출된다. 연관성 점수(association_score)란 추천 후보인 오브젝트 o_1이 속해 있는 서비스 클래스 O와 질의에서 요청된 오브젝트 a_1이 속한 서비스 클래스 A와의 연관성이 얼마나 있는지를 계산하는 것이다. 예를 들어 설명해 보자. 서비스 클래스 A가 속해 있는 연관 클래스 집합이 3개 있다고 가정하고 각각에 속한 서비스 클래스의 종류가 아래 표1과 같다고 하자.

[표 1] 연관 클래스 집합의 예

연관 클래스 집합의 번호	소속 서비스 클래스	지지도
1	A, B, C	0.1%
2	A, D, E, F	0.02%
3	A, X, Y	0.033%

이 경우 서비스 클래스 B와 C에 속한 오브젝트의 연관성이 다른 서비스 클래스 D,E,F,X,Y에 속한 오브젝트에 비해 높을 것으로 예상 될 수 있다. 즉 연관 클래스 집합의 지지도에 따라 연관성 점수를 부여하였다. 본 논문에서는 각 연관 클래스 집합의 빈발 항목(frequent item set)인 소속 서비스 클래스만을 이용하여 연관성 점수를 계산하였다. 즉 연관 규칙의 신뢰도(confidence)를 고려하지 않았지만 이를 고려하면 보다 정교한 추천이 가능할 것으로 기대된다. 이는 향후 연구 과제로 남겨둔다. 오브젝트 o_1의 중요도 점수(weight_score)란 해당하는 오브젝트가 갖는 중요도를 고려하기 위한 것으로 오브젝트의 weight 속성값에 따라 점수를 부여하였다. 오브젝트 o_1의 거리 점수(distance_score)란 해당하는 오브젝트와 요청된 오브젝트 a_1과의 거리에 따라 점수를 부여하였다. 또한 선반입 점수 계산을 위해 각 점수 앞에 곱해진 상수 값 c_1, c_2, c_3는 각각의 기준 점수에 대한 중요도를 반영하는 것으로 적용하는 응용 프로그램의 특성에 따라 서로 다른 값으로 설정 되어야 할 것이다. 본 논문에서는 1:1:1의 비율로 실험하였다.

5. 성능 평가

이동 환경에서 알고리즘의 성능 평가를 유효하게 수행하기 위해서는 데이터(data)와 질의(query), 사용자의 이동성(mobility)과 연결(connectivity)의 특징을 비롯한 실험 수행의 환

경(execution guideline)에 대한 설정이 중요한 의미를 갖는다. 본 절에서는 성능 평가 시 사용된 각종 용어를 정의하고, 성능평가 결과가 갖는 타당성을 입증하기 위해 설정된 실험 기준에 대하여 설명한다. 또한 파라메터를 변경하면서 성능평가를 수행하여 제안된 STAP의 효율성을 확인하였다. 논문에서 설정한 성능평가의 환경은 이동 환경에서의 벤치마크 기준을 설정한 [15]논문을 기반으로 데이터와 질의, 사용자의 행위 특성 그리고 수행 환경 4가지의 측면을 고려하였다.

- 데이터 : 인위적으로 생성되었으며 위치 정보를 포함하는 공간 데이터이다.
- 질의 : 이동 컴퓨팅 환경에서 수행되는 질의는 여러 가지 형태가 있지만 그 중에서 가장 일반적인 형태는 위치 기반 질의(LDQs :Location Dependent Queries)로 이는 사용자의 위치와 관련하여 질의 결과가 달라지는 형태이다[15]. 본 성능평가에서는 LDQ를 포함한 다양한 형태의 질의를 사용하였다.
- 사용자의 행위 특성 : 이동 사용자의 움직임 (방향과 속도 등)과 연결성의 특성을 나타낸다.
- 수행 환경 : 고정된 네트워크에 연결된 데이터 서버가 이동하는 사용자의 질의에 응답하는 일반적인 형태를 띈다.

5.1 공간 데이터와 질의 데이터

공간 데이터를 저장하는 테이블의 속성은 오브젝트 ID, 서비스 클래스, 오브젝트 이름, 오브젝트의 위치 좌표값, 표지물 정보, 중요도와 선택수로 구성된다. 테이블은 인위적으로 생성된 총 50,000 개의 오브젝트를 포함하며 각각의 오브젝트는 점(point) 형태로 (x, y) 좌표를 이용하여 위치를 표시하는 공간 데이터이다. 대상이 되는 영역(test area)은 오른쪽 위와 왼쪽 아래 점으로 표현되는 사각형의 영역이다. 왼쪽 아래 점의 위치를 (0,0)으로 하고, 오른쪽 위의 점의 위치를 (1000,1000)으로 정한다. 각각의 데이터는 10개의 중심점을 기준으로 가우시안 분포(Gaussian distribution)의 형태로 분포한다. 이때 가우시안 분포란 중심으로 갈수록 데이터의 분포가 더욱 밀집되는 형태를 말한다. 각각의 데이터는 500개의 서비스 클래스에 속해 있다. 서비스 클래스란 오브젝트가 제공하는 서비스의 종류를 기준으로 분류한 것으로 논문[10]을 기준으로 하여 500여 개의 서비스 클래스로 구분하였다. 하나의 서비스 클래스에 50 ~ 150개의 오브젝트가 속해 있다. 표지물이란 한 지역에 대한 대표성을 갖는 공간 오브젝트로 보통 지리 데이터를 질의할 때 가까이 위치한 표지물을 기준으로 질의하는 특성이 있으므로[13] 각 지역에 대한 표지물 여부를 속성으로 관리한다. 전 영역에 걸쳐서 200개의 표지물이 분포되어 있다. 중요도는 해당하는 오브젝트가 갖는 중요도를 수치로 표현한 것이다. 하나의 오브젝트가 갖는 중요도는 응용 프로그램에 따라 다르게 나타난다. [7]에서는 한 오브젝트가 갖는 중요도를 동일한 속성을 갖는 다른 오브젝트들의 위치를 기준으로 한 보로노이 영역으로 보았고, 모바일 광고와 같은 응용에서는 광고주가 제공하는 요금에 비례하여 중요도를 결정할 수 있을 것이다. 본 논문에서는 각 오브젝트의 중

요도를 1부터 최대 20까지 임의로 할당하였다. 이때 표지물의 중요도는 10과 20으로 다른 오브젝트에 비해 높은 중요도를 갖도록 하였다. 각 오브젝트의 선택수는 질의 발생시 마다 해당 오브젝트를 선택하는 경우에 1씩 증가하도록 하였다. 이는 일정 기간이 지난 후에 배치 작업을 통하여 해당 오브젝트의 중요도에 반영(feedback)될 수 있도록 접근 수를 기록하는데 이용된다. 즉 많이 이용되는 오브젝트의 중요도를 차츰 증가시키도록 접근 수에 근거한 캐쉬 정책 수립 시 기준 값이 되도록 한다.

성능평가에서 사용된 전체 질의 수는 90,735개이며 총 100명의 사용자를 대상으로 각 사용자별로 평균 900여 개의 질의를 수행하였다. 이동 컴퓨팅 환경에서의 성능 평가 시 사용되는 질의는 크게 3가지로 구분된다[15]. 첫째 Location Aware Query(LAQ)는 오브젝트의 주소나 우편번호 등 위치 관련 속성을 포함하는 질의지만 사용자의 위치와는 관련이 없는 질의이다. 둘째, Location Dependent Query(LDQ)는 질의의 결과가 사용자의 현재 위치와 관련 있는 질의이다. 마지막으로 Non-Location Related Query(NLR_Q)는 상호 등을 이용하여 찾는 질의로 위치와 관련 없는 질의이다. 본 성능 평가에서는 질의의 형태를 5가지로 구분하였다. 각각의 형태에 해당하는 질의의 수와 전체 질의에 대한 %는 표 2와 같다.

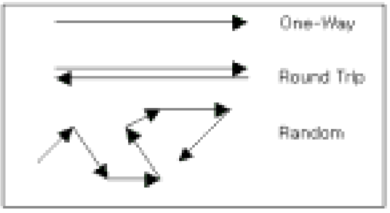
[표 2] 질의 형태 구분

형태	질의 수 (전체 질의 %)	종류
Type 1	34,099 (38 %)	NLR_Q : 오브젝트의 ID를 이용한 질의 사용자의 현재 위치로부터 일정 범위 내에 (distance) 위치하는 오브젝트를 조회함.
Type 2	8,400 (9 %)	LDQ : 사용자의 위치에서 가장 가까운(closest to) 특정 서비스 클래스에 속하는 오브젝트 찾기
Type 3	13,500 (15 %)	LAQ : 특정 오브젝트(type2에 속함)와 가까운 위치(distance)에 있는 특정 속성을 갖는 오브젝트 찾기
Type 4	5,732 (6 %)	LAQ : 특정 오브젝트(type2에 속함)와 가장 가까운 (closest to) 표지물 찾기
Type 5	29,004 (32 %)	NLR_Q : 오브젝트의 ID를 이용한 질의 사용자의 현재 위치와 무관함

일반적으로 LBS 응용에서는 현재 사용자의 위치와 가까운 오브젝트를 질의하는 경우가 많다고 가정한다[6]. 본 성능 평가에서 사용된 질의를 분석하면 현재 사용자의 위치에서 일정 범위 내에 위치한 오브젝트가 질의 되는 비율이 68% (type 1~ type4)를 차지한다. 각 질의 레코드는 현재 사용자의 위치를 나타내는 속성 UserX와 UserY 값을 갖고, 질의 요청 시간을 Time 속성에 저장한다. 또한 질의의 목표가 되는 대상 오브젝트의 공간 상의 위치를 나타내는 속성으로 ObjectX와 ObjectY를 갖는다.

5.2 사용자의 행위 특성과 성능평가 기준

사용자의 행위 특성은 크게 연결성(connectivity)과 움직임(model of movement)의 특성으로 나누어 설명할 수 있다. 연결성은 논문 [2]에서 정의한 것과 같이 주로 네트워크에 연결되어 작업을 수행하는 "workaholics"한 형태와 비연결 상태가 주로 발생하는 "sleepers"으로 나누어진다. 본 성능 평가에서 연결성은 고려하지 않는다. 둘째, 움직임에 대한 특성은 속도, 방향, 움직임의 형태로 구분될 수 있다[15]. 성능 평가에서는 각 사용자가 동일한 속도로 움직인다고 가정하였고, 방향은 8가지 방향 (0도부터 45도씩 더해가면서 315도까지 8가지 방향을 설정함)으로 구성되며, 움직임의 형태는 직선형(one-way), 회귀형(Round Trip), 임의형(Random) 3가지 형태로 이루어진다. 직선형은 공간 상의 한 점 S로부터 다른 한 점 T까지 움직이면서 질의를 발생하는 형태이다. 이 경우 한번 지나간 위치는 다시 돌아오지 않는다. 회귀형의 경우는 공간상의 한 점 S로부터 시작하여 다른 한 점 T까지 직선형으로 움직인 후 방향을 180도 바꾸어서 다시 시작점인 S로 돌아오는 형태이다. 임의형은 움직임의 방향을 임의로 결정하는 경우로 직선형 경로의 연결 모양으로 지나간 위치를 다시 방문할 수도 있다. 각각의 경우를 그림으로 나타내면 그림 5와 같다.



[그림 5] 움직임의 형태

본 성능평가에서는 캐쉬 적중률 (hit ratio), 선반입의 연관성(relevance), 선반입의 완전성(completeness)을 기준으로 캐쉬 교체 알고리즘과 선반입 알고리즘의 성능을 평가한다. 캐쉬 적중률은 캐쉬 내에 질의의 대상이 되는 오브젝트가 존재하는 질의 수를 전체 질의 수로 나눈 값의 %로 나타낸다. 연관성과 완전성은 선반입 알고리즘에 근거한 추천 집합이 얼마나 유용한지를 평가하기 위한 것[9]으로 아래 식에서 R은 추천된 아이템의 집합이고, V는 그 중에서 사용자가 질의에 사용한 아이템의 집합이다.

$$\text{Hit ratio} = \frac{Hit\_Query\_Count}{Total\_Query\_Count} * 100$$

$$\text{Relevance} = \frac{|\mathrm{R}| \cap |V|}{|\mathrm{R}|}$$

$$\text{Completeness} = \frac{|\mathrm{R}| \cap |V|}{|\mathrm{V}|}$$

[그림 6] 성능평가 기준

5.3 성능 평가 결과

본 절에서는 구성된 성능 평가 환경 하에서 기존의 연구에서 제안된 캐쉬 교체 알고리즘들을 이용하여 General-Prefetch와 STAP의 성능을 비교, 평가 하였다. 또한 선반입 자체의 성능을 평가하기 위한 연관성(relevance)과 완전성(completeness)에 대한 실험을 수행하였다. 본 성능평가에서 사용하는 알고리즘은 크게 선반입 부분과 캐쉬의 교체 부분으로 나누어진다. 선반입 방법은 General-Prefetch와 STAP 두 가지로 나누어 실험한다. 또한 캐쉬 교체 알고리즘은 LRU, FAR을 이용하여 실험하였다. STAP에서 사용하는 연관 클래스 집합은 표 1과 같은 형태의 테이블에 저장되어 있으며 실험에서 사용된 입력 파라메터의 종류와 설정 값은 표3과 같다.

[표 3] 입력 파라메터의 종류

입력 파라메터	설명 및 종류
캐쉬 크기	이동 호스트의 캐쉬 크기로 서버의 DB 크기를 기준으로 0.5%, 1%, 2%, 3%, 4%, 5% 등의 6가지 종류로 실험
선반입 크기	이동 호스트가 선반입하는 크기이며 캐쉬를 기준으로 0.5%, 1%, 2%, 3% 등의 4가지 종류에 대해 실험
선반입 방법	1: General-Prefetch, 2: STAP
캐쉬교체 방법	1: LRU, 2: FAR
사용자 ID	사용자의 ID (1~100 사이의 값)
경로 ID	경로의 ID (1~14 사이의 값)

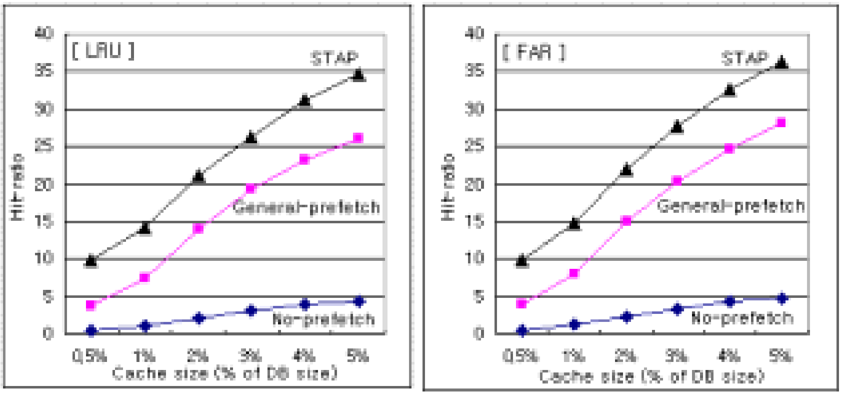
표 4에서 각 성능 평가의 수행 단위 수란 한 사용자가 하나의 경로를 지나면서 수행한 성능평가의 횟수이다. 1회의 수행 단위에서 발생하는 평균 질의 수는 65~70회이다.

[표 4] 입력 파라메터 설정 값

입력 파라메터	실험1	실험 2
캐쉬 크기	0.5%, 1%, 2%, 3%, 4%, 5%	3%
선반입 크기	2%	0.5%, 1%, 2%, 3%
선반입 방법	1: General-Prefetch, 2: STAP , 3: No-Prefetch	1: General-Prefetch, 2: STAP
캐쉬 교체 방법	1: LRU, 2: FAR	1: LRU, 2: FAR
사용자 ID	1 ~ 10	11 ~ 20
경로 ID	1 ~ 14	1 ~ 14
수행 단위 수	5,040	2,240

5.3.1 실험 1: 캐쉬 크기 증가에 따른 알고리즘별 적중률의 변화

실험 1은 이동 호스트의 캐쉬 크기를 서버 데이터베이스 크기의 0.5%부터 5%까지 변화시키면서 선반입 알고리즘의 적중률을 평가하였다. 그 결과 모든 경우에서 캐쉬 크기가 증가할수록 더 우수한 캐쉬 적중률을 나타내었다. 또한 제안한 선반입 알고리즘의 우수성을 보이기 위하여 선반입을 수행하지 않는 No-Prefetch와 비교하여 General-Prefetch과 STAP의 적중률을 실험한 결과, 선반입을 수행하지 않은 No-Prefetch에 비해 General-Prefetch과 STAP이 월등히 좋은 성능을 나타내었다. 이때 캐쉬 교체 알고리즘은 LRU와 FAR를 이용하였다. 동일한 선반입 알고리즘을 이용하는 경우에 두 캐쉬 교체 알고리즘의 적중률을 비교하면 FAR 알고리즘이 접근 시간을 기준으로 캐쉬를 교체하는 LRU 방법에 비해 조금 더 우수한 성능 (2%)을 나타내는 데 이러한 결과는 발생되는 질의가 시간을 기준으로 생성되지 않고 표 2에서와 같이 위치 기반 질의가 많은 데이터의 특징에 기반 한다.



(a) 캐쉬 교체 알고리즘 : LRU (b) 캐쉬 교체 알고리즘 : FAR

[그림 7] 캐쉬 크기 증가에 따른 알고리즘별 적중률의 변화

5.3.2 실험 2: 선반입 크기의 변화에 따른 알고리즘별 적중률, 연관성, 완전성의 변화

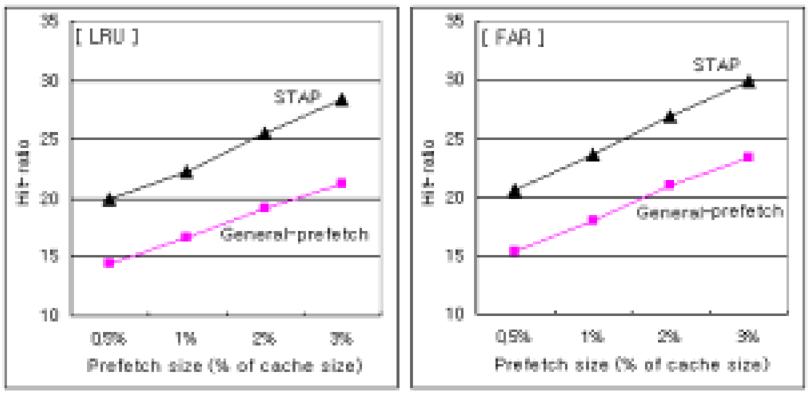
실험 2는 이동 호스트의 캐쉬 크기를 전체 서버의 데이터베이스 크기의 3%로 고정하고, 선반입 크기를 캐쉬 크기의 0.5%에서 3%로 증가하면서 알고리즘 별로 성능평가를 수행하였으며 각각의 결과는 그림 8, 9, 10과 같다.

그림 8의 결과와 같이 선반입 크기가 증가 할수록 캐쉬의 적중률은 비례하여 증가하며 실험 1과 같이 STAP을 사용한 경우가 General Prefetch에 비해 좋은 성능을 나타내었다. 동일한 선반입 알고리즘을 사용하는 경우에는 FAR 방법의 캐쉬 교체가 LRU 방법을 사용하는 것에 비해 우수한 결과를 보였다.

그림 9는 선반입 크기 증가에 따른 연관성의 변화를 보여준다. 실험 결과 선반입 크기가 증가함에 따라 보다 많은 수의 데이터가 선반입 되어 오고, 이에 따라 관련성이 적은 데이터에 대한 선반입이 증가하게 되어 결과적으로 연관성이 점차 떨어지는 결과를 보여준다. 또한 STAP의 경우 선반입 크기가 0.5%에서 1%로 변화할 때 급격한 변화를 보여주는데

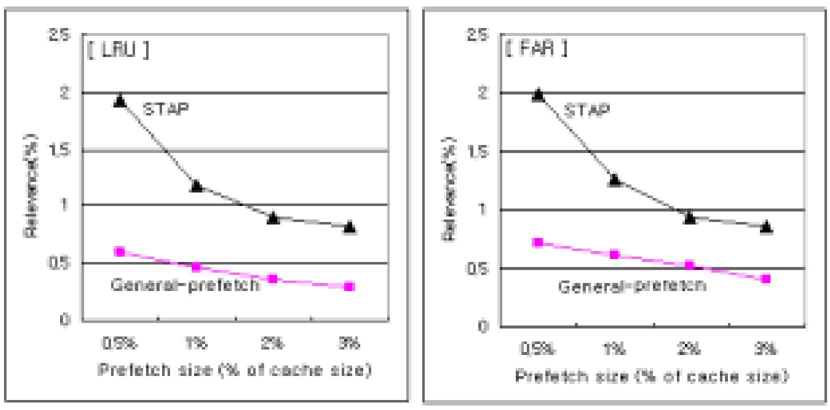
이는 실험에서 사용된 캐쉬 크기가 전체 서버의 데이터의 3%이므로 선반입 되는 데이터의 수가 7.5개(0.03*50,000*0.005=7.5)에서 15개로 변화하는 시점에 선반입에 의한 관련성이 적어진다는 것을 의미한다. 이에 반해 General Prefetch의 경우에는 연관성의 감소율이 일정한데 이는 위치를 기반으로 수행하는 선반입의 경우, 질의의 연관성과는 상관없이 사용자와 데이터와의 거리를 기반으로 선반입이 수행되기 때문이다.

그림 10의 결과는 선반입 크기 증가에 따르는 완전성의 변화를 보여주며 STAP을 사용한 경우가 General-Prefetch에 비해 5~7% 좋은 성능을 나타내었다.



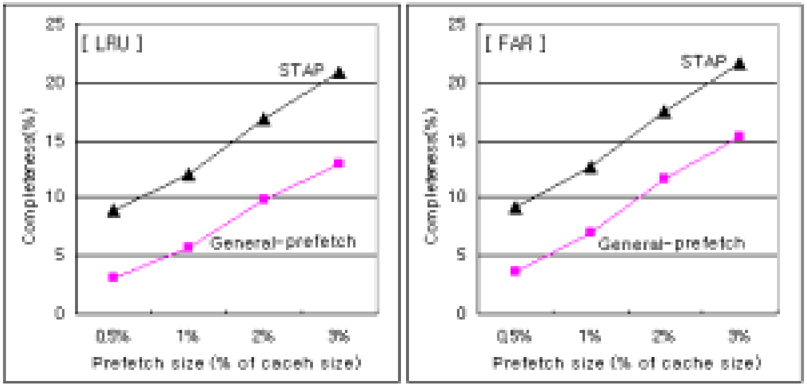
(a) 캐쉬 교체 알고리즘 : LRU (b) 캐쉬 교체 알고리즘 : FAR

[그림 8] 선반입 크기에 따른 알고리즘별 적중률 변화



(a) 캐쉬 교체 알고리즘 : LRU (b) 캐쉬 교체 알고리즘 : FAR

[그림 9] 선반입 크기에 따른 알고리즘별 연관성 변화



(a) 캐쉬 교체 알고리즘 : LRU (b) 캐쉬 교체 알고리즘 : FAR

[그림 10] 선반입 크기에 따른 알고리즘별 완전성 변화

6. 결론 및 향후 연구 과제

본 논문에서는 이동 컴퓨팅 환경에서 위치 기반 서비스를 효과적으로 지원하기 위하여 사용자의 질의 사이에 내포되어 있는 시-공간적 연관성을 이용한 새로운 연관성 기반 선반입 알고리즘인 STAP를 제안하였다. 저자가 아는 범위에서 STAP은 기존의 캐쉬 알고리즘이 고려하지 못한 서비스 간의 연관성을 고려한 첫번째 선반입 알고리즘으로 이를 통해 동일한 속성을 갖는 오브젝트들 만을 대상으로 향후 접근성을 고려하는 것이 아니라 다양한 속성을 갖는 오브젝트들 사이의 서비스 연관성을 선반입 정책에 이용할 수 있게 되었다. 또한 알고리즘의 성능 평가를 유효하게 수행하기 위한 실험 환경을 데이터와 질의, 사용자의 이동성의 측면에서 설정하였고, 설정된 환경에서 다양한 파라메터를 적용하여 선반입과 캐쉬 교체 알고리즘에 대한 성능 평가를 수행하였다. 그 결과 논문에서 제안한 STAP이 사용자의 위치와 데이터의 거리만을 고려하는 General-Prefetch에 비해 월등히 좋은 성능을 나타내는 결과를 보였다.

향후 연구 과제로는 본 논문에서 제안한 연관성 기반 선반입 알고리즘 확장하여 연관 규칙에 신뢰도를 이용하는 방법을 도입하여 추천의 정확도를 높이고, 사용자의 연결성과 속도 측면, 질의 형태에 변화에 따른 성능 평가를 수행할 것이다. 또한 개인화(personalization)와 연계하여 개인의 역사적 특징(history)을 고려하도록 확장하는 연구가 필요할 것이다.

참고 문헌

[1] J. Jing, A. Helal, A. Elmagarmid, "Client-Server Computing in Mobile Environments," ACM Computing Survey, Vol. 31, No. 2, pages 117-157, 1999.

[2] Daniel Babara, Tomasz Imielinski, "Sleepers and Workaholics: Caching Strategies in Mobile Environments," In Proceedings of ACM SIGMOD conference, pages 1-12,

1994.
[3] Tomasz Imielinski, Henry F. Korth, "Introduction to Mobile Computing," Mobile Computing, Kluwer Academic Publishers, pages 1-43, 1996.
[4] Tomasz Imielinski, S. Viswanthan, "Wireless Publishing: Issues and Solutions," Mobile Computing, Kluwer Academic Publishers, pages 299-329, 1996.
[5] Boris Y. Chan, Antonio Si, Hong V. Leong, "Cache Management for Mobile Databases: Design and Evaluation," In proceedings of the Fourteenth International Conference on Data Engineering, pages 54-63, 1998.
[6] Qun Ren, Margaret H. Dunham, “Using Semantic Caching to Manage Location Dependent Data Mobile Computing,” In proceedings of MobiCom 2000, Boston, Massachusetts, pages 210-221, 2000.
[7] Baihua Zheng, Jianliang Xu, Dik L.Lee, “Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environment,” IEEE Transactions on Computers, vol. 51, No. 10, pages 1141-1153, October 2002.
[8] Wen-Chih Peng and Ming-Syan Chen, “Mining User Moving Patterns for Personal Data Allocation in a Mobile Computing System,” In Proceedings of the 29th International Conference on Parallel Processing (ICPP-2000), August 21-24, 2000.
[9] Dhananjay S. Phatak and Rory Mulvaney, “Clustering for Personalized Mobile Web Usage”, In Proceedings of the IEEE FUZZ'02, pages 705-710, 2002.
[10] Yasuhiko Morimoto, “ Mining Frequent Neighboring Class Sets in Spatial Databases,” In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 353-358, 2001.
[11] 백형종, 구경이, 김유성, "이동 컴퓨팅 환경에서 위치기반 질의를 지원하는 이동 트랜잭션 모델",한국 정보 과학회 데이터베이스 연구회지 17권 3호, pages 17-28, 2001.
[12] 류근호, 안윤애, 이준욱, 이용준, "이동 객체 데이터베이스와 위치 기반 서비스의 적용", 한국 정보 과학회 데이터베이스 연구회지 17권3호, pages 57-74, 2001.
[13] 서영덕, 안경환, 홍봉희, "인터넷 GIS의 사용 분석", 한국 정보 과학회 데이터베이스 연구회지 18권 1호, pages. 41 -51, 2002.
[14] Ho-Sook Kim, Hwan-Seung Yong, “Associative class mining for location-based information retrieval “, EWHA-DBLAB-TR-2003-1, 2003.
[15] Ayse Y. Seydim, Margaret H.Dunham, “A Location Dependent Benchmark with Mobility Behavior,“ International Database Engineering and Applications Symposium (IDEAS'02), pages 74 –85, 2002.