고정 그리드 인덱싱에서 공간과 시간 필터링을 이용한 범위 질의 처리

전 세 길[†] · 나 연 묵[††]

요 약

최근 들어 이동 통신 분야에서 이동하는 고객을 위한 위치 기반 서비스가 중요한 서비스로 부각되고 있다. 이동객체 응용의 경우 갱신 연산이 많고, 부하가 특정 지역에 집중되는 특징이 있다. 윈도우 나 원형 형태의 범위질의는 위치 기반 서비스에서 중요한 질의 중 하나이다. 이러한 범위질의에서는 부분 포함된 셀과 완전 포함된 셀을 구분해야할 필요가 있다. 또한 올바른 한정된 객체를 골라내기 위해 시간 영역을 고려할 필요성이 있다. 본 논문에서는 갱신연산을 최소화하기 위해 고안되어진 2번째 단계에 고정 그리드 구조를 적용한 2단계 인덱스 구조를 적용한다. VP 필터링과 윈도우 셀 필터링 기법을 이용한 공간 셀 필터링 기법과 Time Zone 개념을 사용하여 시공간 개념이 결합된 필터링 기법을 제안한다. 제안된 방법의 성능 측정을 위해서 다른 필터링 조합을 가지고 다양한 윈도우 질의와 원 질의에 대해서 실험 결과를 보인다.

Range Query Processing using Space and Time Filtering in Fixed Grid Indexing

Segil Jeon[†] · Yunmook Nah[††]

ABSTRACT

Recently, the location-based service for moving customers is becoming one of the most important service in mobile communication area. For moving object applications, there are lots of update operations and such update loads are concentrated on some particular area unevenly. Range queries, whose range can be window or circular, are the most essential query types in LBS. We need to distinguish completely contained cells from partially contained cells in those range queries. Also, it is necessary to consider temporal dimension to filter out qualifying objects correctly. In this paper, we adopt two-level index structures with fixed grid file structures in the second level, which are designed to minimize update operations. We propose a spatial cell filtering method using VP filtering and a combined spatio-temporal filtering method using time zone concepts. Some experimental results are shown for various window queries and circular queries with different filtering combinations to show the performance tradeoffs of the proposed methods.

키워드 : 위치 기반 서비스(Location Based Service), 고정 그리드 인덱싱(Fixed Grid Indexing), 범위 질의(Range Query), 공간 필터링(Space Filtering), 시간 필터링(Time Filtering)

1. 서 론

최근에 위치 기반 서비스(Location Based Service)와 관련된 기술이 중요한 기술로 부각되고 있다. 대부분의 위치 기반 서비스가 시간, 공간 모두와 관련이 있으며, 각 시간대별로 위치 정보를 저장하게 된다. 즉, 저장 대상이 이동 객체가 되며 질의 형태도 기존의 지리정보시스템(Geographic Information System)에서와 다르게 시간적 특성과 공간적 특성 모두를 이용하는 시공간 질의의 형태가 대부분이다.

이동객체의 특징은 크게 두 가지로 볼 수 있다. 첫째 연속적으로 이동하기 때문에 빈번한 갱신 연산이 수행된다. 서비스 가입자의 위치변경은 인덱스 구조의 변경을 초래하기 때문에 인덱스 구조가 자주 변하지 않는 구조가 적합하다. 둘째 핸드폰 가입자의 경우 도시의 중심가에 특정 시간에 밀집되는 현상이 발생하는 특징이 있다.

이동 객체를 효율적으로 검색하기 위한 공간 인덱스 구조로 일반적으로 R-트리 기반 구조와 고정 그리드 방식이 대표적인 색인 구조이다. 이동 객체의 경우 빈번한 갱신에 의해 트리 구조를 사용하는 경우 많은 비용을 소요하게 된다. 본 논문에서는 이동 객체의 빈번한 갱신을 저비용으로 처리하기 위한 인덱스 구조로 고정 그리드 방식을 사용하며 고정 그리드에서 각 셀 간의 순서를 결정하는데 Z-순서(Z-ordering) 기법을 사용한다[1].

※ 본 연구는 대학 IT연구센터 육성지원 사업의 지원과 한국과학재단의 특정기초연구과제(R01-2003-10133-0)의 일부지원에 의해 수행되었음.
† 준 회 원 : 단국대학교 대학원 전자컴퓨터공학과
†† 정 회 원 : 단국대학교 전기전자컴퓨터공학과 교수
논문접수 : 2003년 8월 4일, 심사완료 : 2004년 4월 30일

특정 시간, 특정 지역에 이동객체가 밀집되어 부하가 집중되는 경우 부하를 분산 시키는 방법으로 비 균등 2단계 그리드 구조가 제안되었다[4-6]. 이 구조에서 1단계 인덱스는 그리드내의 각 셀의 부하에 따라 부하 분산을 위해 셀의 모양이 유동적으로 변화하는 단계이며, 2단계 인덱스는 1단계 인덱스의 특정 셀을 다시 고정된 크기로 분할한 그리드 구조이다. 본 논문은 2단계 인덱스에서 고정된 그리드의 범위 질의 처리를 주 대상으로 한다.

일반적으로 위치 기반 시스템은 현재의 위치 정보뿐만 아니라 과거의 위치 정보까지 유지해야하는데 이동 객체의 경우 계속적으로 움직이기 때문에 시간이 지남에 따라 그 데이터의 양이 기하급수적으로 증가하게 된다. 또한, 시간에 따라 계속적으로 변하는 연속적인 시공간 질의의 결과는 다음의 조건을 만족해야 한다.

① 관련된 공간 질의를 만족하는 객체
② 유효한 시간 내에 있어야 함
③ 유효 시간에 따라 계속적으로 변화

시공간 질의를 효과적으로 수행하기 위해서는 위의 3가지 조건을 만족하는 시공간 질의 방법이 필요하다.

예를 들어, 움직이고 있는 핸드폰 사용자가 현재 위치에서 5km 범위 내에 있는 호텔을 찾는다고 했을 때 현재 결과가 A, B, C라고 가정하면 사용자가 움직이는 방향과 속도를 고려했을 때 1분후에는 B만 포함 될 수도 있다. 위의 예에서는 결과 대상 객체는 고정되고 질의 영역이 움직이는 경우이며, 다른 예로 질의 영역이 움직이고 질의 대상 객체도 움직이는 경우도 있을 수 있다. 핸드폰 사용자로부터 5km 범위 내에 있는 순찰차를 찾는다고 한다면 다음 시간에는 질의 범위의 위치와 대상 객체들의 위치가 모두 변하게 된다.

위의 두 상황 모두 특정 시간 범위가 주어진다면 질의 범위에 대한 공간적인 필터링뿐만 아니라 시간적인 필터링이 필요하게 된다. 시간 범위는 질의 대상 시작시간, 종료시간 및 머무른 시간으로 표현될 수 있다.

질의 대상 시작 시간, 종료 시간은 질의 대상 객체의 시간 정보와 비교하여 해당되는 객체를 쉽게 찾을 수 있지만 각 객체가 그 시간 내에 머무른 시간은 알 수가 없다. 주어진 시간 내에 객체가 머무른 시간을 구할 수 있는 시간 필터링 기법이 필요하다.

기존의 시공간 질의를 지원하는 인덱스 구조로 TPR 트리(Time Parameterized R-tree)가 있으며, 시간을 고려한 질의 방법으로 TP 질의(Time Parameterized Query)가 있다[16]. TPR 트리나 TP질의는 R-트리에 적합한 방안이며 본 논문에서 사용하는 고정 그리드 방식에서는 부적합 하다.

본 논문에서는 계속적으로 변화하는 이동객체의 부하 분산과 갱신 연산을 줄이기 위해 비 균등 2단계 고정 그리드 구조를 사용한다. 2단계의 고정 그리드 방식에서 시공간 질의 중에서 범위 질의를 위해 공간 필터링 방법을 제안하고 특정 시간 범위 내에 존재하는 객체를 구하기 위해 Time Zone 기법을 사용한다. 공간 필터링 기법으로 윈도우 범위 질의 시에 질의 범위와 각 객체간의 좌표 비교를 줄이기 위해 윈도우 필터링 기법을 제안하였고, 원 범위 질의 시에 각 객체와 원 중심과의 거리 계산량을 줄이기 위해 R트리 계열에서 사용되는 VP 필터링 기법을 고정 그리드 방식에 확장 적용하였다[11, 12]. Time Zone 기법을 이용하면 주어진 시간 범위 내의 질의 대상 객체 수를 줄일 수 있다. 그 객체가 머무른 시간을 구하기 위한 시간 범위 필터링 기법을 제안한다. 본 논문에서 제안한 범위 질의를 위한 공간 필터링 기법과 Time Zone 기법의 유용함을 보이기 위해 성능 평가 한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구에 대해서 기술하고, 3장에서는 공간과 시간을 개별적으로 필터링 하는 방법을 설명하고 시간과 공간을 통합 필터링 하는 기법에 대해서 기술 한다. 4장에서는 본 논문에서 제안한 기법에 대한 성능을 보이고 마지막으로 5장에서 결론과 향후 연구 과제를 제시한다.

2. 관련 연구

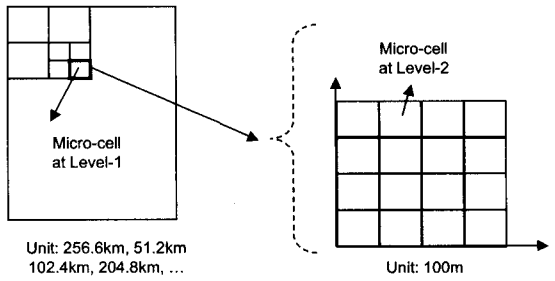
2.1 비 균등 2단계 고정 그리드 인덱싱

시공간 객체의 위치정보를 인덱싱하기 위한 구조로 기존에 제안된 RT-트리, 3D R-트리, TB-트리, TPR-트리등이 있다[7-9]. 이러한 기존의 R-트리 기반 구조들은 대량의 이동객체의 움직임에 따라서 많은 노드의 분할 및 합병이 일어난다. 트리에서 노드의 분할, 합병은 삽입, 삭제 시 디스크에 많은 부하를 초래하며 처리 시간도 기하급수적으로 증가한다.

본 논문에서 사용하는 비 균등 2단계 고정 그리드 구조는 이동 객체의 이동정보를 저장 시에 트리 기반 인덱싱 에서의 문제점을 해결하기 위해 고정 그리드를 기반으로 한 2단계의 인덱싱 구조이다. (그림 1)은 이동 객체의 위치 정보를 인덱싱 하기 위한 비균등 2단계 고정 그리드 구조를 나타낸다.

1단계는 전체 지역을 담당하는 최 상위 단계의 비 균등 그리드 구조이고 2단계는 균일한 크기로 분할되는 균등 그리드 구조이다. 1단계에서는 인구 밀도나 기타 정보에 따라서 지리적인 지역으로 분할되며 기본적으로 25.6km를 단위로 해서 사각형 형태로 분할된다. 소위 이러한 사각형을 매크로 셀(macro-cell)이라 하며, 담당하는 지역에 따라 그 크기가 달라지게 된다. 1단계의 각 셀을 한 컴퓨터가 담당하는 것으로 가정하였는데 각 셀에 객체수가 일정양 이상으로 증가하면 셀이 분할되어 각각을 다른 컴퓨터가 담당하

게 된다. 1단계의 각 셀은 담당하는 지역의 객체수에 따라 분할, 합병되어 각각 컴퓨터를 할당함으로써 대량의 데이타가 집중되어 부하가 집중되는 현상을 막을 수 있다[4-6].



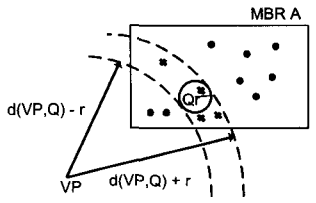
(그림 1) 위치 정보의 인덱싱을 위한 비균등 2단계 고정 그리드 구조

2단계에서는 각 영역이 균일한 형태인 100m×100m 크기로 분할되며 마이크로 셀(micro-cell)이라 한다.

본 논문에서는 2단계에서의 균등 그리드 구조를 고정 그리드 구조라 명명하고 시간을 고려했을 경우의 질의 처리 방안을 기술한다.

2.2 VP 필터링 기법

VP 필터링 기법의 기본 개념은 공간 도메인 내에 있는 객체들과 질의 점과의 거리 계산시간을 줄이자는 것이다. 즉, 질의 중심점과 해당 질의 영역 내에 있는 모든 객체와의 거리 계산을 수행하지 말고 질의 결과가 될 가능성이 없는 객체들은 거리 계산에서 미리 제외하자는 의미이다. 이를 위해서는 어떤 객체가 질의 결과가 될 가능성이 있는지 없는지의 판단을 위한 기준이 필요하다. 이때 기준으로 사용하는 것이 VP(Vantage Point)와 이를 기준으로 하는 각 객체와의 거리 값이다. VP 필터링을 위해서는 VP를 사전에 선정하는 작업과 객체정보가 데이타베이스 테이블에 삽입 시에 계산되어 반영되어야 한다.



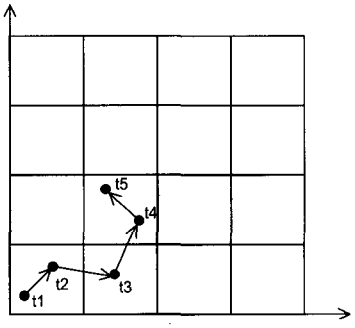
(그림 2) R트리 계열에서 VP 필터링 기법의 사용예

(그림 2)는 R트리 계열에서 한 MBR내의 객체 들 중에서 중심점 Q로부터 거리 r 내에 있는 객체를 찾는 경우 MBR 내에 있는 모든 객체에 대해서 중심점 Q로부터 거리계산을 하는 것이 아니라 각 객체의 VP와의 거리가 d(VP,Q)-r 과 d(VP,Q)+r 내에 있는 객체에 대해서만 거리계산을 수행한다. 각 객체의 VP와의 거리는 미리 계산되어 각 트리의 노드에 객체 정보와 함께 저장 된다.

대상 도메인 공간에서 VP를 선택하는 방법은 도메인 공간의 구석에서 한 객체를 선택하는 방법과 경험적 방법 등이 있다. 도메인 공간의 구석에서 VP를 선택하는 방법은 Yianlos에 의해 소개되었다. 실험에 의하면 도메인 공간 내에서 VP를 선택하는 것보다 임의의 한 구석에서 VP를 선택하는 것이 객체의 변별력을 높이는 관점에서 높은 성능을 보이는 것으로 나타나 있다[12]. 경험적 방법은 VP를 선택하기 위해 먼저, 도메인 객체 중에서 임의의 한 객체를 선택하여 다른 모든 객체들과의 거리를 계산한다. 계산된 거리를 기준으로 VP와 가장 먼 거리에 있는 객체를 VP로 선택한다. 본 논문에서는 고정 그리드에서의 질의 특성을 고려하여 양쪽 구석 지점을 VP로 선정하는 방법을 사용하였다. 이 방법은 VP를 선정하기 위해 계산하는 시간을 소비하지 않아도 되며, 검증된 성능을 나타낸다.

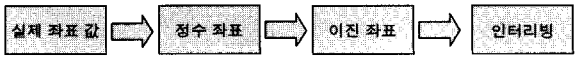
2.3 Z-순서를 이용한 고정 그리드 방식 인덱싱

고정 그리드(fixed grid) 방식은 주어진 공간 도메인을 일정한 크기의 셀로 분할하여 나누는 방식이다[9]. (그림 3)은 이동 객체의 경로 저장을 위해 특정 공간 도메인을 고정 그리드로 분할한 예이다. 특정 객체의 이동 경로는 매 시간 모든 정보가 데이타베이스에 반영되는 것이 아니라 객체가 셀 경계를 벗어날 경우에만 데이타베이스에 반영하게 된다. (그림 3)은 특정 객체의 샘플링 시간별 위치를 나타내며 t1, t3, t4 시간의 위치만 데이터베이스에 반영된다.



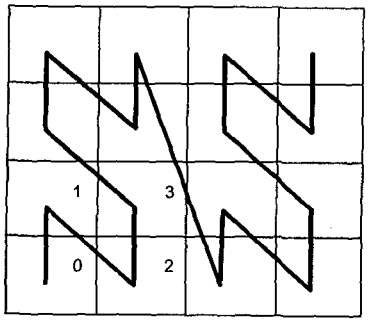
(그림 3) 고정 그리드를 이용한 공간 분할

Z-순서는 2D 고정 그리드 인덱스 구조에서 이웃한 셀 들에 순서를 부여하는 여러 방법 중 하나이다. 각 셀의 번호는 기준 좌표로부터 X축, Y축 에서 몇 번째 셀인지를 구하여 이진수로 변환한 후 X축, Y축 각 자리의 값을 인터리빙 시켜 구할 수 있다. 특정 객체가 속한 셀 번호를 구하기 위해서는 다음 (그림 4)와 같은 과정을 거친다.



(그림 4) 객체가 포함되는 셀 번호 구하는 과정

(그림 5)는 Z-순서를 나타내며, 고정 그리드 인덱스 구조에 Z-순서를 사용하면 특정 셀과 일정 거리 내에 인접한 셀을 찾을 때 위에서 설명한 셀 번호 구하는 계산을 역으로 적용하면 쉽게 찾아 낼 수 있다.



(그림 5) Z-순서

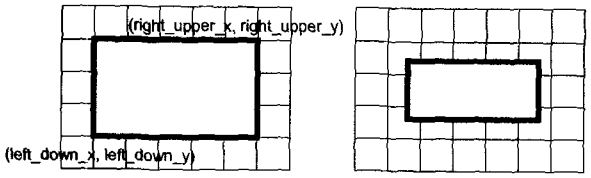
예를 들어, 실제 좌표를 변환한 정수 좌표가 (1, 2)인 경우 정수 좌표를 각각 이진좌표로 변환 하면 X = 01, Y = 11이 된다. X, Y를 좌표를 인터리빙시키면 shuffle("01", "11") = 0111이며 10진수 7이 된다. 즉, 해당 좌표는 셀 7번에 해당된다고 볼 수 있다.

3. 시공간 통합 필터링

3.1 공간과 시간의 개별 필터링

3.1.1 윈도우 질의 시 셀 필터링

(그림 6)은 윈도우 질의 시 질의 영역과 셀 간의 포함 관계를 나타내고 있다. 그림에서 굵은선으로 표시 된 것이 질의 영역이며 범위는 좌측하단의 좌표와 우측 상단의 좌표로 표현될 수 있다. (그림 6)(a)는 질의 영역이 완전 포함 셀로만 이루어진 경우이며 이 경우에 필터링된 셀이 모두 완전 포함 셀이므로 별도의 좌표 비교가 필요없게 된다. (그림 6)(b)의 경우에는 외곽 부분의 12개의 셀이 부분 포함 셀에 해당되며 안쪽의 완전 포함 셀을 제외한 부분 포함셀의 객체에 대해서는 질의 범위에 대해서 좌표 비교가 필요하다.



(a) 완전 포함 셀만 포함한 윈도우 질의 (b) 부분 포함 셀을 포함한 윈도우 질의

(그림 6) 윈도우 질의 예

(그림 7)은 윈도우 질의 시에 완전 포함 셀과 부분과 부분 포함 셀을 구별해 내기 위한 알고리즘이다. 윈도우 질의 시 셀 필터링을 수행하면 테이블 검색 양을 줄일 수 있다.

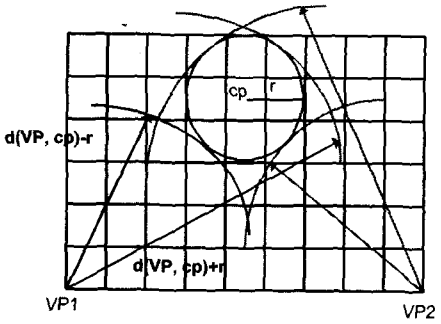
```
INPUT    ldx, ldy : 질의 범위의 좌측 하단 좌표
         rux, ruy : 질의 범위의 우측 상단 좌표
OUTPUT   Partially_Included_Cell_List : 부분적으로 포함된 셀
         Fully_Included_Cell_List : 완전히 포함된 셀

Partilally_Included_Cell_List = 다음 조건을 만족하는 정점을 포함하는 셀 리스트
    x = ldx,  ldy <= y <= ruy   or   x = rux, ldy <= y <= ruy
    y = ldy,  ldx <= x <= rux   or   y = ruy, ldx <= x <= rux
// 질의 영역의 외곽선과 겹치는 셀 리스트
Fully_Included_Cell_List = Partilally_Cell_List에 포함되지 않는 나머지 관련된 셀
```

(그림 7) 윈도우 질의 시 셀 필터링 알고리즘

(그림 7)에서는 질의 영역에 해당되는 셀 리스트를 구하였다는 것을 가정하고 질의 영역 좌표와 각 셀의 범위를 비교하여 질의영역과 겹치는 영역이 있는 셀은 부분 포함 셀로 정하여 그 안에 포함된 객체에 대해서는 질의 범위에 대해서 모두 좌표 검사를 수행한다. 질의 영역과 겹치는 부분이 없는 셀은 완전포함 셀로 정하여 그안의 포함된 객체에 대해서 별도의 좌표 검사를 수행하지 않는다.

3.1.2 원 범위 질의 시 셀 필터링

본 논문에서는 VP 필터링을 통해 분류되는 셀을 질의 중심점과 객체간의 별도의 거리계산이 필요 없는 완전 포함 셀과 거리 계산이 필요한 부분 포함 셀로 분류 하였다. 완전 포함 셀과 부분 포함 셀이 되기 위한 조건은 (그림 9), (그림 10)과 같다. 기존의 VP 필터링 기법을 고정 그리드 구조에 적용하기 위해서 양쪽의 구석점에 2개의 VP점을 선택하였다.



(그림 8) VP 필터링을 이용한 원 범위 질의

```
(조건 1)
   질의 범위 내에 있는 셀
(조건 2)
   d (VP, point) < d (VP, cp) - r or
   d (VP, point) > d (VP, cp) + r
   인 점을 포함하지 않음
```

(그림 9) 완전 포함 셀 조건

(그림 8)에서 point는 특정 셀 내에 있는 임의 객체에 대

한 좌표이며 (그림 9)에서와 같이 cp와 r은 질의 중심과 반지름 이다. d(VP, point)는 VP와 객체와 거리이다.

```
(조건 1)
   질의 범위 내에 있는 셀
(조건 2)
   d(VP, point) < d(VP, cp) - r or
   d(VP, point) > d(VP, cp) + r 인 점을 포함
```

(그림 10) 부분 포함 셀 조건

(그림 11)은 결정된 VP1, VP2와 삽입 객체 사이의 거리를 저장하는 컬럼을 포함한 객체 위치 정보 저장 테이블 구조이다. VP1_DIST와 VP2_DIST는 객체의 위치 정보가 삽입 시에 한번 계산되어 저장된다. (그림 11)의 테이블은 데이타 객체를 위치 정보를 저장하는 테이블 구조이며 본 논문에서는 랜덤한 좌표 데이타를 발생시켜 각 좌표와 VP간의 거리를 미리 계산하여 테이블에 저장한 후 시뮬레이션 하였다.

(그림 11)에서 OID는 핸드폰 번호와 같은 객체 식별값이며 X, Y는 각 좌표 값이며 Time은 시간 CELL_NO는 해당 좌표에 대한 셀 번호이다. VP1_DIST, VP2_DIST는 좌측 VP와의 거리와 우측 VP와의 거리를 의미한다.

OID	X	Y	Time	CELL_NO	VP1_DIST	VP2_DIST

(그림 11) 객체 위치 정보 저장 테이블 구조

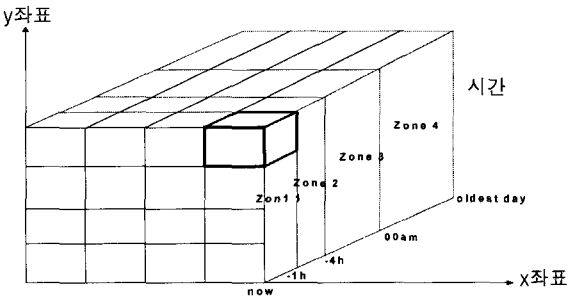
VP 필터링을 이용한 원 범위 질의 처리 알고리즘은 (그림 12)와 같다. 원 질의시의 부분 포함 셀과 완전 포함 셀을 구분하기 위해서 우선 원 질의와 관련된 셀 목록을 구한 후에 VP 필터링 조건을 비교하여 부분 포함 셀과 완전 포함 셀을 구분한다. 결과는 각 셀 목록이 된다.

```
INPUT    cp : 질의 영역의 중심점, r ; 범위 반지름
OUTPUT   Partially_Included_Cell_List : 부분적으로 포함된 셀
         Fully_Included_Cell_List : 완전히 포함된 셀

Cell_List = Z-순서를 이용해서 원과 관련된 셀 리스트를 구함
FOREACH Cell_List 내의 셀 번호 C
  Point = C내에 존재하는 객체들의 정점 값
  IF exists((d(VP, point) < (d (VP, cp)-r)ORd(VP, point)
          (d(VP, cp) + r)))
    Partilally_Included_Cell_List 에 셀번호 C를 삽입
  ELSEIF not exists ((d(VP, point) < (d(VP, cp)r)
                    ORd(VP, point) > (d (VP, cp) + r)))
    Fully_Included_Cell_List에 셀번호 C를 삽입
```

(그림 12) VP 필터링을 이용 원 범위 질의 시 셀 필터링 알고리즘

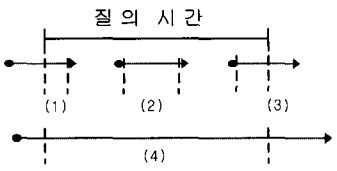
3.1.3 Time Zone 기법과 시간 범위 필터링

이동 객체의 위치정보는 계속적으로 변화하므로 샘플링 간격을 길게 한다고 해도 대량의 정보가 된다. 대량의 정보를 하나의 테이블에 저장할 경우 데이타 검색 시간이 기하급수적으로 증가하게 된다. 따라서, 본 논문에서는 위에서 설명한 위치 정보 저장 테이블을 Zone 이라는 시간의 범주에 따라서 시간대 별로 다른 테이블에 정보를 저장한다.



(그림 13) Time Zone이 적용된 시공간 데이타 모델

(그림 13)은 Zone의 개념이 적용된 시공간 데이타 모델이며, X, Y축으로 이루어진 각 면은 Z-순서를 이용한 고정 그리드이며 Z축은 시간에 따른 변화를 나타낸다. Z축의 Zone1은 100m 이상 움직였으며 현재 시간으로부터 1시간 전의 위치정보를 저장하고 있으며, Zone2는 400m 이상을 움직였으며 1시간에서 4시간 사이의 위치 정보를 저장한다. Zone3는 3.2km 이상을 움직이고 4시간 이후부터 그날의 시작까지의 위치 정보를 저장하며, Zone4는 이전날짜의 데이타로써 백업의 역할을 한다. 해당 Zone은 각각 해당 테이블에 저장되며 검색 시간대에 따라 해당 Zone의 테이블을 검색하면 된다.



(그림 14) 질의시간과 이동 객체가 움직인 시간과의 관계

(그림 14)는 해당 질의시간과 이동 객체가 움직인 시간과의 관계를 나타낸다. (그림 14)에서 각 점은 객체에 해당되며 질의 영역에 대한 해당 필터링 결과 내에서 질의 영역내에 있는 객체들이다. 화살표는 움직인 시간을 나타낸다. (1)과 (3)은 객체가 움직인 시간이 질의 시간에 부분 포함되며 (2), (4)는 완전 포함되어 있는 경우이다. 여기서 (2)는 질의 시간 내에 포함된 상태 이며, (4)는 객체의 이동 시간이 질의 시간을 포함한 상태이다.

(그림 15)는 각 객체와 질의의 시간 영역이 (그림 14)에서 보인 경우 같이 완전 교차하는 경우와 부분 교차하는 경우를 구분하여 교차하는 시간을 구하는 알고리즘이다. (그림 15)에서 o.time은 객체가 움직인 시간이며 q.time은 해당 질의 시간 범위를 표현한다. enter 함수는 객체의 움직인 시

간이 질의 범위 시간으로 들어오는 중인지 검사하고 leave 함수는 나가는중 인지 검사한다.

contained 함수는 객체의 시간이 질의 시간 안에 완전히 포함되어 있는지 검사하는 함수이다.

```
INPUT  OID : 객체 ID
       Q   : 질의 = (질의영역, 시작시간, 종료시간)
OUTPUT (Ts, Te) : Ts는 교차 시작 시간, Te는 교차 종료시간

[Ts, Te] = [0, ∞]
 FOR EACH 객체 o
  {o.time} = 질의 시간과 연관되어 움직인 시간 간격
 FOR EACH o.time
 IF q time contained (o.time)
      Ts = start time of o.time
      Te = end time of o.time
      return [Ts, Te]
ELSE
      Ts = start time of q.time
      Te = end time of q.time
      return [Ts, Te]
```

(a) 질의 시간에 완전 포함

```
 INPUT  OID : 객체 ID
        Q   : 질의 = (질의영역, 시작시간, 종료시간)
OUTPUT (Ts, Te) : Ts는 교차 시작시간, Te는 교차 종료시간

[Ts, Te] = [0, ∞]
 FOR EACH 객체 o
  {o.time} = 질의 시간과 연관되어 움직인 시간 간격
  FOR EACH o.time
  IF enter (o.time)
      Ts = start time of q. time
      Te = end time of o. time
      return [Ts, Te]
  ELSE  IF leave (o. time)
      Ts = start time of o. time
      Te = end time of q. time
      return [Ts, Te]
```

(b) 질의 시간에 부분 포함

(그림 15) 각 객체와 질의가 교차하는 객체의 교차시간을 구하는 알고리즘

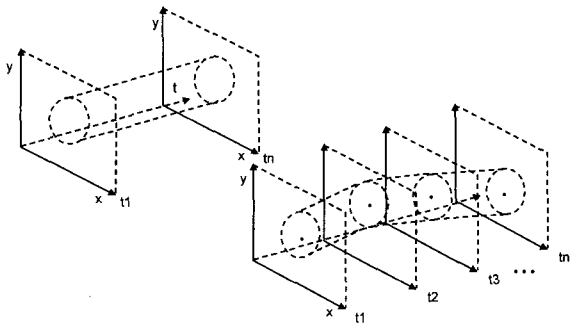
3.2 시공간 통합 필터링

3.2.1 시공간 통합 필터링 알고리즘

이동 객체에 대한 범위 질의 유형은 다음 (그림 16)과 같다. (그림 16)(a)의 경우 질의 중심점은 고정 되어 있고 시간 범위만 주어진 경우이며 (그림 16)(b)의 경우는 시간 범위가 주어지고 질의 중심점이 변화하는 경우이다.

(그림 16)에서 t1과 t2사이는 이동 객체의 위치정보를 샘플링하는 간격이 된다. 샘플링 시간 사이의 이동은 무시한다. (그림 16)의 두 가지 경우를 고려하여 특정 시간 범주 내에서 특정 공간상의 범위 내에 있는 객체를 검색하기 위한 절차는 다음 (그림 17)과 같다.

위의 알고리즘을 수행 한 후 리턴되는 결과 값은 공간 범위에 의한 필터링과 시간 필터링을 수행 후 질의 조건을 만족하는 객체들의 리스트이다.



(a) 질의중심 고정 시 (b) 질의중심 이동 시

(그림 16) 범위 질의 유형

```
INPUT  OID : 객체 ID
       Q   : 질의 = (질의영역, 시작시간, 종료시간, 머무른 시간)
OUTPUT  Result_List : 결과 객체 리스트

Zone_No = 질의 시간을 커버하는 Zone 번호를 구함
IF Object = 동적 and Query = 정적
 < Partially_Included_Cell_List, Fully_Included_Cell_List >
   = Cell 필터링을 수행 결과
 IF 원범위 질의
    Object_List = Partially_Included_Cell_List 셀 내에 포함된 객체
                  들에 대해거리 d를 구한 후 d<r 인 객체 ID
    Object_List = Object_List + Fully_Included_Cell_List의 모든
                  객체 ID
 ELSE IF  윈도우 범위 질의
    Object_List = Partially_Included_Cell_List의 셀 내의 객체에
                  대해 범위 내에 있는  ID
    Object_List = Object_List + Fully_Included_Cell_List의 모든
                  객체 ID
 FOR (시작 시간 <= t <= 종료시간)
 Time_Object_List = Object_List 내에서 time = t 인 객체 ID
 FOREACH Time_Object_List내의 객체
 { (Ts, Te) } = Time 필터링 결과 교차 시간 집합
 Result_List = (Te - Ts)의 합계 <= 머무른 시간 인 객체의 ID

ELSE IF Object = 동적 and Query = 동적
 FOR (시작 시간 <= t <= 종료시간 )
 < Partially_Included_Cell_List, Fully_Included_Cell_List >
   = Cell 필터링을 수행 결과
 IF 원범위 질의
   Object_List = Partially_Included_Cell_List 셀 내에 포함된 객체
                 들에 대해거리 d를 구한 후 d < r인 객체 ID
   Object_List = Object_List + Fully_Included_Cell_List의 모든
                 객체 ID
 ELSE IF 윈도우 범위 질의
   Object_List = Partially_Included_Cell_List의 셀 내의 객체에
                 대해 범위 내에 있는 객체 ID
   Object_List = Object_List + Fully_Included_Cell_List의 모든
                 객체 ID
 Time_Object_List = Object_List 내에서 time = t인 객체
 { (Ts, Te) } = Time 필터링 결과 교차 시간 집합
 Result_List = (Te - Ts)의 합계 <= 머무른 시간인 객체의 ID
```

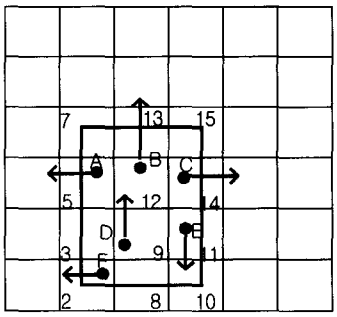
(그림 17) 셀 필터링과 Time Zone, 시간 범위 필터링을 이용한 이동 객체 검색 알고리즘

(그림 17)에서 Zone_No는 질의 시간에 해당하는 Zone 번호이다. 우선 Object가 동적으로 움직이면서 Query가 정적인 경우와 Object가 동적으로 움직이면서 Query가 동적으로 움직이는 경우를 구분한다. 그런 다음 원 범위 질의와 윈도우 범위를 구분하여 본 논문에서 제안한 셀 필터링을 수행하면 Partially_Included_CellList와 Fully_Included_CellList로 구분 되어 진다. 즉, 부분 포함 셀과 완전 포함 셀로 구분되어진다.

마지막으로 Ts(시작시간), Te(종료시간)을 이용해 객체가 머무른 시간을 체크하여 해당 시간만큼 머무른 객체의 ID만을 반환한다.

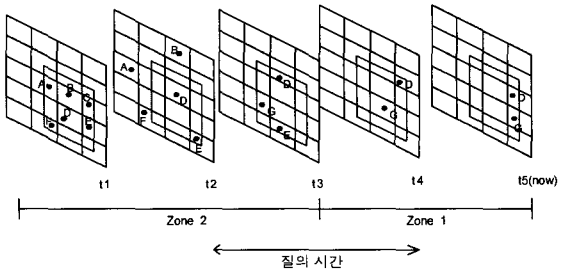
3.2.2 윈도우 범위 질의 예

(그림 18)에서와 같은 윈도우 질의에서 범위가 고정된 경우 한번의 셀 필터링만 수행하면 된다. (그림 18)에서 굵은 선은 질의 범위를 나타내고 각 점은 객체이며 화살표는 다음 시간에 이동하는 위치이다. 셀 필터링 결과 완전 포함된 셀은 9, 12가 되며 2, 3, 5, 7, 8, 10, 11, 13, 14, 15는 부분 포함이 된다. 완전 포함에 해당 되는 셀들의 객체는 모두 질의 범주 안에 들어가며 부분 포함에 해당 되는 셀들의 객체는 질의 범위를 표현하는 좌표와 비교하는 연산을 수행해야 한다. 셀 필터링과 포함하고 있는 객체를 찾는 연산을 수행 후 현재는 A, B, C, D, E, F가 모두 포함되며, 다음 시간에는 D, E만 해당되게 된다.

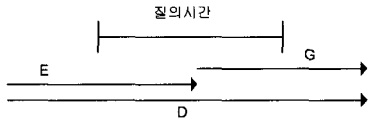


(그림 18) 질의 범위가 고정된 윈도우 질의

(그림 19)에서와 같이 각 시간 대 별로 객체의 위치가 일어난다고 했을 때, 주어진 범위와 시간(t2~t4)내에 2초간 머무른 객체들을 검색하라고 한다면 t1, t2와 관련된 Zone은 Zone1과 Zone2가 된다. (그림 20)의 질의는 정적이며 객체는 동적인 경우이므로 공간적 범위 조건에 의한 셀 필터링은 한번만 수행하면 된다. 셀 필터링 결과에 있는 셀이 포함하는 객체 중에서 주어진 시간 범위를 만족하는 객체를 구해보면 t2에 D, t3에 D, G, E, t4에 D, G가 된다. 현재 질의와 관련된 객체 D, G에 대해서 시간 범위 필터링을 수행하면 시간이 부분 참여하는 객체는 G(t3~t4)가 있으며 완전 참여 하는 객체는 D(t2~t4)가 있다. G의 경우 객체의 이동정보가 Zone1, Zone2 각각에 저장되어 있으며, D는 Zone1과 Zone2에 걸쳐서 저장되어져 있다. 위의 예에서 각 시간 간격을 1초라고 가정하면 G는 질의 시간 내에 머무른 시간이 1초이고 D는 2초이므로 D만이 결과에 포함되게 된다.



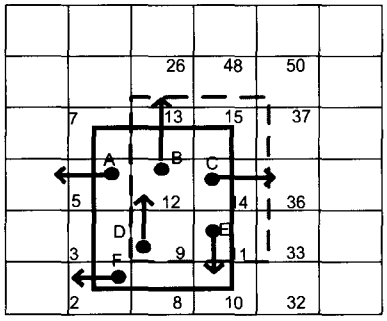
(a) 시간 대 별 객체의 이동



(b) 질의시간과 객체의 이동 시간과의 관계

(그림 19) 질의 범위가 고정된 시간에 따른 윈도우 질의

(그림 20)은 질의 범위가 움직이는 경우에 해당되며 그림에서 점선으로 된 질의 범위는 다음 시간의 질의 범위가 된다. 이러한 경우 각 시간대 별로 질의 공간 범위에 대해서 셀 필터링을 매번 수행해야 한다.



(그림 20) 질의 범위가 움직이는 윈도우 질의

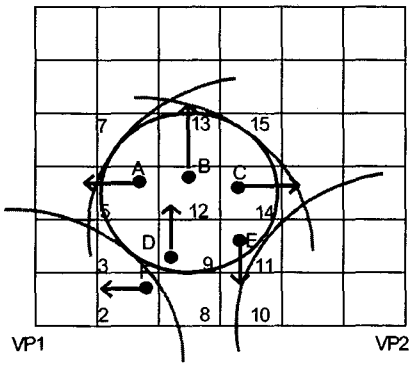
점선으로 표현된 질의범위의 경우 완전 포함된 셀은 9, 11, 12, 13, 14, 15가 되며 부분 포함된 셀은 26, 33, 36, 37, 48, 50이 된다. 각 시간대 별로 셀 필터링 결과에 해당되는 결과 객체를 구한 후 시간 필터링을 수행한다. (그림 19)에서 두 번째 시간에 포함되는 객체는 B, C, D가 된다.

3.2.3 원 범위 질의 예

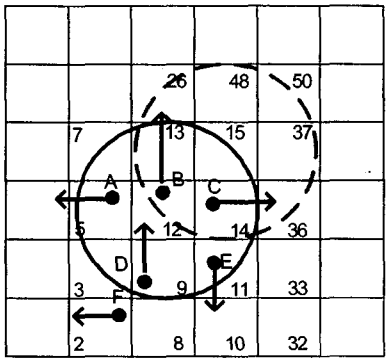
(그림 21)(a)는 질의 범위가 고정된 상태에서 원 범위 질의를 수행하는 상황이다. 윈도우 질의에서와 마찬가지로 질의 범위가 고정되어 있으므로 셀 필터링은 매 시간대 별로 수행할 필요가 없으며 오직 한번만 수행하면 된다.

우선 VP 필터링 기법을 사용하여 셀 필터링을 수행하면 VP1에 의해 3, 15 가 부분 포함 셀이 되며, VP2에 의해 7, 11이 부분 포함 셀이 된다. 나머지 5, 9, 12, 13, 14는 완전 포함 셀이 된다. 현재 범위 내에 포함된 셀 내의 객체는 A, B, C, D, E가 되면 다음 시간에는 D만이 해당된다. 셀 필터링을 수행 후 해당 객체가 결정되면 윈도우 질의에서와 마찬가지로 시간 범위 필터링을 수행하여 특정 범위, 특정 시간에 해당되는 결과 객체를 얻는다.

(그림 21)(b)는 원 범위 질의 중에서 질의 범위가 움직이는 경우이다. 윈도우 질의에서와 마찬가지로 각 시간대 별로 셀 필터링을 수행해야 하며 (그림 21)(a)에서는 두번째 시간에 셀 필터링을 수행하면 VP1에 의해 12, 50이 부분 포함 셀이 되며, VP2에 의해 26, 36이 부분 포함 셀이 된다. 각 시간대 별로 셀 필터링 결과에 해당되는 결과 객체를 구한 후 시간 범위 필터링을 수행한다. (그림 21)(b)에서 두번째 시간에 포함되는 객체는 B, C가 된다.



(a) 질의중심 고정 시



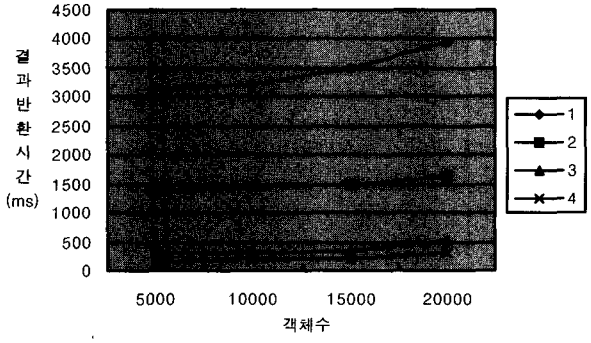
(b) 질의중심 이동시

(그림 21) 원 범위 질의 예

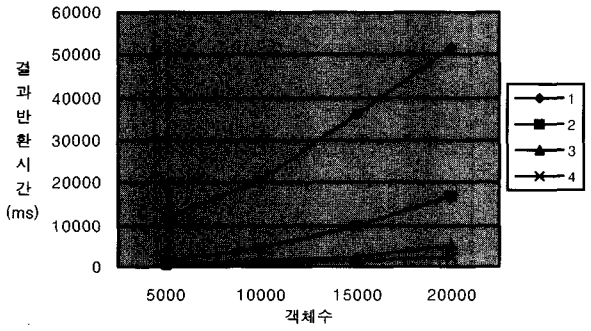
4. 성 능

본 논문에서 제시한 알고리즘의 효율성과 유용성을 보이기 위해 알고리즘을 구현하고 실험 하였다. 여러 알고리즘 중에서 고정 그리드 방식 인덱싱에서 Time Zone 기법과 공간 필터링을 이용한 윈도우 범위 질의와 원 범위 질의 처리 알고리즘의 성능을 측정하기 위해 MS SQL Server 2000의 저장 프로시저(stored procedure)를 작성 한 후 질의 시간을 측정하였다. 윈도우 질의와 원 질의에 대해서 Time Zone 개념과 셀 필터링을 사용한 경우를 비교 평가하였다.

성능 평가를 위해 메모리 128Mbytes의 PentiumIII-866 MHz PC에서 수행하였다. 운영체제는 MS Windows 2000 이다. 실험을 위해 전체 영역을 1600m×1600m로 하고 Zone1, Zone2의 최근 4시간 데이터를 가정하였다.



(a) 윈도우 질의



(b) 원 질의

1 : Zone1을 Zone 개념만을 이용해 질의
2 : Zone1을 Zone 개념과 셀 필터링 개념을 이용해 질의
3 : Zone2를 Zone 개념만을 이용해 질의
4 : Zone2를 Zone 개념과 셀 필터링 개념을 이용해 질의

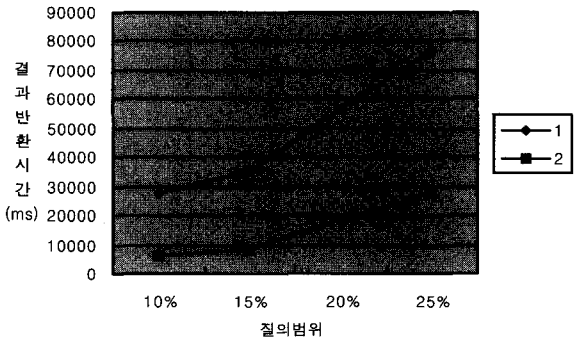
(그림 22) 객체 수에 따른 질의 성능평가 예

(그림 22)(a)는 질의 영역을 전체 영역의 25%로 고정하고 객체의 수를 변화 시켜 가며 Zone1과 Zone2의 내용을 윈도우 질의한 결과이다. Zone1의 질의 대상 데이터의 양이 Zone2 보다 많으므로 전체적으로 Zone2를 대상으로한 질의보다 반환시간이 오래 걸렸다. Zone1, Zone2 질의 모두 Zone개념과 셀 필터링을 사용한 경우에 반환시간이 더욱 빨라짐을 알 수 있다.

(그림 22)(b)는 질의 영역을 25%로 고정하고 Zone1과 Zone 2의 내용을 각각 원 질의 한 결과이다. 윈도우 질의에서와 마찬가지로 Zone1의 데이터양이 Zone2에 비해 많으므로 전체적으로 결과 반환 시간이 높게 나왔다. 셀 필터링 기법을

쓰지 않은 경우 객체 수가 증가함에 따라서 그래프의 기울기가 급해지는 이유는 원 질의시에는 객체수만큼 거리계산이 많아지기 때문이다.

(그림 23)은 객체 수를 15,000개로 고정하고 시간상으로 Zone1과 Zone2에 걸치는 상황을 가정하고 질의 범위를 변화 시켜 가며 실험한 결과이다. 질의 범위가 증가할수록 결과반환시간이 증가하는 것을 볼 수 있으며, 셀 필터링을 사용한 경우가 Zone 개념만을 사용한 경우에 비해 거리계산 대상 객체 수가 적어져 결과 반환시간이 빠른 것을 알 수 있다.



1 : Zone 개념만을 이용해 질의
2 : Zone 개념과 셀 필터링 개념을 이용해 질의

(그림 23) 질의범위에 따른 원질의 성능평가

5. 결 론

이동 객체의 현재 위치와 과거 위치를 기반으로 하는 위치 기반 서비스가 점차 증가하고 있다. 이러한 위치 기반 서비스의 저장 및 인덱싱의 대상이 되는 이동 객체는 시간에 따라서 위치가 계속적으로 변화하며 시간이 지남에 따라 그 양은 기하급수적으로 증가하게 된다. 또한, 이동 객체를 핸드폰 가입자라고 했을때 이동은 불규칙적이며 특정 노드에 집중되는 현상이 발생할 수 있다. 이 경우 부하를 분산시킬 수 있는 방안이 있어야 하며 누적되는 데이터의 양을 줄일 수 있는 방법이 있어야한다. 또한 이동 객체의 경우 현재 위치가 계속 변화하므로 그 변화에 따라서 전체 인덱스 구조의 변화를 최소화 할 수 있는 방안이 있어야 한다.

본 논문에서는 이동 통신 서비스에서 계속적으로 이동하는 고객의 위치정보를 저장 및 검색하기 위해 비 균등 2단계 고정 그리드 인덱싱 방법을 사용하여 처리했다. 기존의 R-트리 계열의 인덱싱 방법은 이동 객체의 움직임에 따라 갱신 연산이 이루어지므로 위치 정보를 인덱싱 하기에는 부적합 하다고 볼 수 있다.

비 균등 2단계 고정 그리드 인덱싱에서 1단계는 담당하는 지역의 객체 수에 따라 동적으로 변하는 단계이며, 2단계는 특정 지역을 균등 분할하여 사용했다. 따라서 비 균등 2단계 고정 그리드 인덱싱을 사용하면 부하를 분산 시킬 수 있는 효과가 있다. 시간이 지남에 따라 데이터베이스에 저장되는 객체의 양은 기하급수적으로 증가하는 현상을 줄이기 위해 Zone 기법을 제안했다. Zone 기법을 사용하면 과거로 갈수록 데이터의 양이 줄어들어 검색 시간을 감소시킬 수 있다. 또한 본 연구에서는 특정 지역을 균등한 크기로 분할하는 2단계에서의 고정 그리드를 가정하고 효율적인 범위 질의 방안을 제안했다. 윈도우 질의 시 비교 대상이 되는 객체의 수를 줄일 수 있는 셀 필터링 기법과 원 질의 시 거리 계산 대상이 되는 객체의 수를 줄이는 VP 필터링 방법을 제안하였다. 본 논문에서 질의는 공간뿐만 아니라 시간도 다루었으며, 시간상으로는 Time Zone 기법을 사용하여 질의 대상 객체의 양을 줄일 수 있으며 머무른 시간을 구하기 위한 시간 범위 필터링 기법을 기술하였다.

본 논문에서 제안한 범위 질의 시 셀 필터링 기법과 Zone 기법의 유용함을 보이기 위해 성능평가 하였다.

향후 연구 과제로는 현재 연구는 단일 노드에서의 이동 객체의 저장 및 질의를 가정하고 평가하였으므로 노드가 분산 형태에서의 저장 및 질의 방안에 대한 연구가 필요하다.

참 고 문 헌

[1] J.A.Orenstein and T.H. Merrett. A class of data structures for associative searching, Proc. of SIGACT-SIGMOD, pp. 181-190, April, 1984.

[2] 박원순, 전세길, 나연묵, "대용량 이동 객체의 위치 정보 인덱싱", 정보과학회 추계학술발표논문집, Vol.29, No.2, pp. 49-51, Oct., 2002.

[3] 전세길, 나연묵, "고정그리드 인덱싱에서 VP 필터링을 이용한 범위 질의 처리", 정보처리학회 춘계학술발표논문집, 제10권 제1호, pp.1531-1534, May, 2003.

[4] Y. Nah, M. H. Kim, T. Wang, K. H. Kim, Y. K. Yang, "TMO-structured Cluster-based Real-Time Management of Location Data on Massive Volume of Moving Items," STFES 2003, Hakodate, Japan.

[5] M. H. Kim, T. Wang, K. H. Kim, Y. Nah, Y. K. Yang, "Distributed Adaptive Architecture for Managing Very Large Volume of Moving Items," IDPT 2003, Beijing, China.

[6] 나연묵, K. H. Kim, 왕태형, 김문희, 이종훈, 양영규, "GALIS : LBS 시스템의 클러스터 기반 신축성소유 아키텍처", 데이터베이스연구, 제18권 제4호, pp.33-47, Dec., 2002.

[7] Saltenis, S., et al., Jensen, E., Leutenegger, S., Lopez, M., "Indexing the Positions of Continuously Moving Objects," Proc. ACM SIGMOD, 2000.

[8] Theodoridis, Y., Vazigannis, M. and Sellis, T., "Spatio-Temporal Indexing for Large Multimedia Applications," Proc. 3rd IEEE Conf. on Multimedia Computing and Systems(ICMCS), 1996.

[9] J. Nievergelt, H. Hinterberger and K. C. Sevcik. The grid file : an adaptable, symmetric multikey file structure. ACM TODS, Vol.9, No.1, pp.38-71, March, 1984.

[10] Xu, X., Han, J. and Lu, W., "RT-tree : An Improved R-tree Index Structure for Spatiotemporal Databases," Proc. 4th Int'l Symp. on Spatial Data Handling(SDH), 1990.

[11] Jeffrey K. Ujlmann, "Satisfying General Proximity/Similarity Queries with Metric Trees," Information Processing Letters, Vol.40, pp.175-179, 1991.

[12] Tolga Bozkaya and Meral Ozsoyoglu, "Distance-Based Indexing for High-Dimensional Metric Spaces," Proceedings of the ACM SIGMOD Conference, pp.357-368, 1997.

[13] P. N. Yiannilos, "Data Structures and Algorithms for Nearest Neighbor search in General Metric Spaces," ACM-SIAM Symposium on Discrete Algorithms, pp.311-321, 1993.

[14] 김병곤, 이재호, 임해철, "R-tree 계열의 인덱싱 구조에서의 효율적 질의 처리를 위한 VP 필터링", 한국정보과학회 논문지, Vol.29, No.6, pp.453-463, Dec., 2002.

[15] Philippe Rigaux, Michel, Scholl, Agnes Voisard, *Spatial Databases with Application to GIS*, Morgan Kaufmann.

[16] Yufei Tao, Dimitris Papadias, "Time-Parameterized Queries in Spatio-Temporal Databases," ACM SIGMOD, 2002.

[17] Terry, D., Goldberg, D., Nichols, D., Oki, B., "Continuous Queries over Append-only Databases," ACM SIGMOD, 1992.

[18] Chen, J., DeWitt, D. J., Tian, F., Wang, Y, "NiagaraCQ : A Scalable Continuous Query System for Internet Databases," ACM SIGDMOD, 2000.

전 세 길

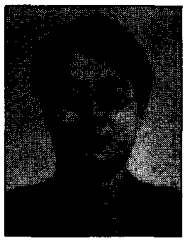
e-mail : sgjeon@dankook.ac.kr

1998년 단국대학교 컴퓨터공학과(공학사)

2000년 단국대학교 대학원 컴퓨터공학과 (공학석사)

2000년~2003년 단국대학교 대학원 전자컴퓨터공학과 박사수료

관심분야 : 데이타베이스, 멀티미디어 데이타베이스, 시공간 데이터베이스



나 연 묵

e-mail : ymnah@dku.edu

1986년 서울대학교 컴퓨터공학과(공학사)

1988년 서울대학교 대학원 컴퓨터공학과 (공학석사)

1993년 서울대학교 대학원 컴퓨터공학과 (공학박사)

1991년 미국 IBM T.J.Watson 연구소 객원연구원

1994년~1999년 DASFAA Steering Committee Secretary

1997년~1999년 한국멀티미디어학회 논문지 편집위원

2000년~2001년 DASFAA Program Committee Member

2001년~2002년 Univ. of California, Irvine 방문연구원

1993년~현재 단국대학교 전기전자컴퓨터공학부 부교수

1996년~현재 한국정보과학회 데이타베이스연구회 운영위원

1997년~현재 서울특별시 정보화추진위원회 위원

2000년~현재 한국정보과학회 논문지 편집위원

관심 분야 : 데이타베이스, 객체지향 데이타베이스, 데이타 모델링, 데이타베이스 설계, 멀티미디어 데이타베이스, 멀티미디어 정보 검색, 멀티미디어 시스템