임의의 다각형 질의 윈도우를 이용한 공간 선택 질의의 정제 전략

(A Refinement Strategy for Spatial Selection Queries with Arbitrary-Shaped Query Window)

유 준 범 † 최 용 진 † 정 진 완 ††
(Junbeom Yoo) (Yong-Jin Choi) (Chin-Wan Chung)

요 약 공간 선택 질의에 사용되는 질의 윈도우로는 직사각형이 주로 사용된다. 하지만, 공간 선택 질의의 윈도우로는 직사각형이 아닌 일반적인 다각형 모양도 가능하며, 최근에는 GIS 등과 같은 응용 프로그램들이 성능 향상으로 인해 보다 많은 공간 데이터를 다룰 수 있게 됨에 따라, 여러 다양한 종류의 응용도 많이 등장하고 있다. 따라서, 직사각형뿐만 아니라 임의의 다각형 형태의 질의 윈도우에도 적합한 정제 단계 수행 전략에 대해 고려해 볼 필요가 있다. 이러한 전략으로는 기존의 공간 조인에서와 같이 plane-sweep 알고리즘을 이용하는 방법이 일반적이다. 하지만, 공간 데이터와 질의 윈도우의 특성을 관찰해보면, 일반적으로 질의 윈도우가 공간 데이터보다 훨씬 간단한 모양으로 구성되어 있음을 알 수 있으므로, 본 논문에서는 이러한 상황에 보다 적합한 정제 단계 수행 방법을 제시하고 있다. 실험을 통해 알 수 있듯이, 질의 윈도우를 구성하는 점의 개수가 약 20개 이하인 일반적인 경우에는, 본 논문에서 제시하는 새로운 방법이 기존의 방법보다 20% 정도 향상된 성능을 보이고 있다.

키워드 : 공간 데이터베이스, 공간선택질의, 정제 전략

Abstract The shape of query windows for spatial selection queries is a rectangle in many cases. However, it can be issued for spatial selection queries with not only rectangular query widow, but also polygonal query window. Moreover, as the applications like GIS can manage much more spatial data, they can support the more various applications. Therefore it is valuable for considering about the query processing method suitable for not only rectangle query window, but also general polygonal one. It is the general state-of-the-art approach to use the plane-sweep technique as the computation algorithm in the refinement step as the spatial join queries do. However, from the observation on the characteristics of spatial data and query windows, we can find in many cases that the shape of query window is much simpler than that of spatial data. From these observations, we suggest a new refinement process approach which is suitable for this situation. Our experiments show that, if the number of vertices composing the query window is less than about 20, the new approach we suggest is superior to the state-of-the-art approach by about 20% in general cases.

Key words : Spatial Database, Spatial Selection Query, Refinement Strategy

1. 서 론

최근에 공간 데이터(spatial data)는 지도제작(cartography), 지형분석(terrain analysis), 항법 시스템(navigation system), GIS(Geographic Information System) 및 공간 데이터베이스(spatial database) 등에서 널리 사용되고 있으며, 그 사용 빈도도 증가 추세에 있다. 공간 데이터 즉, 공간 객체는 매우 복잡한 구조로 이루어져 있으며 크기 또한 다양할 뿐만 아니라 그 양도 매우 크다. 따라서, 일반적인 데이터보다는 질의 즉, 공간 질의(spatial query)의 수행이 복잡하며 또한 수행

† 비 회 원 : 한국과학기술원 전산학과
jbyoo@salmosa.kaist.ac.kr
omni@islab.kaist.ac.kr
†† 종신회원 : 한국과학기술원 전산학과 교수
chungcw@islab.kaist.ac.kr
논문접수 : 2001년 12월 17일
심사완료 : 2003년 3월 11일

하는데 많은 시간이 요청된다[1]. 이러한 공간 질의를 보다 효과적으로 수행하기 위한 연구가 지난 십 여 년 동안 지속되어 왔는데, 이러한 연구는 인덱스를 이용해서 공간 데이터를 효과적으로 접근하려는 연구와 질의의 수행 시간을 줄이기 위한 연구로 분류할 수 있다.

일반적으로 공간 데이터베이스 시스템에서 공간 질의는 전체 데이터 중에서 일부분만을 사용해서 해당 질의를 수행한다. 따라서, 필요한 데이터만을 직접 접근해서 읽어올 수 있게 해주는 인덱스와 같은 데이터 구조체(data structure)를 사용하면, 신속한 데이터의 접근을 통해 보다 효과적으로 질의를 수행할 수 있다. 공간 데이터를 다루는 인덱스인 SAM(Spatial Access Method)는 그 동안 널리 연구되어 왔으며, 대표적으로 R*-tree[2], Hilbert R-tree[3]와 hB-tree[4] 같은 SAM이 있다.

질의의 수행 시간을 줄이려는 연구로는[5]의 여과 및 정제 전략이 대표적이다. 여과 단계는 질의 윈도우와 MBR(Minimum Bounding Rectangle)이 겹치는 객체만을 후보(candidates)로서 일단 걸러내는 단계이며, 정제 단계는 이러한 후보들 중에서 실제로 질의 윈도우와 겹치는 객체만을 얻기 위해서 각 객체의 실제 모양을 이용한 정교한 테스트를 수행하는 단계이다[6]. 정제 단계는 객체의 실제 모양을 이용하는 정교한 검사 알고리즘이 요구되므로 여과 단계보다는 많은 시간이 요구된다[1,6].

앞서 언급한 바와 같이 질의 수행 시간을 줄이기 위한 연구는 여과-정제 전략에 기초하고 있다[1,5]. 특히 다른 기본적인 질의 유형들 보다 많은 수행 시간이 요구되는 공간 조인 질의(Spatial Join Query)에 대한 연구가 많이 진행되어 왔는데, [7,8,9]는 여과 단계에,[10]은 정제 단계에 중점을 둔 연구들이다. 이와 같이 공간 조인을 보다 효과적으로 수행하기 위한 기법들이 많이 연구된 반면에, 공간 선택 질의(Spatial Selection Query)에 대해서는 진행된 연구가 드물다. 이는 공간 선택 질의는 질의 윈도우 내에 있는 공간 데이터들을 검색해 내는 질의인데, 일반적으로 직사각형의 질의 윈도우를 많이 사용했기 때문이다. 하지만, 공간 선택 질의는 직사각형이 아닌 다각형 질의 윈도우로도 효과적으로 수행될 수 있어야 하며, 더욱이 요즘은 GIS와 같은 응용 프로그램들이 기존보다 더욱 다양한 응용들을 수행함에 따라 직사각형 모양의 윈도우만으로는 다양한 응용을 수행할 수 없게 되었다. 이에 따라[6]에서는 처음으로 이러한 상황에 대한 처리의 필요성을 언급하였는데, 이 연구에서는 정제 단계에서 공간 조인에서와 같이 plane-sweep 알고리즘을 사용하여 실제 겹침 여부를 검사하고 있다.

이와 같이 질의 윈도우의 모양을 직사각형이 아닌 일반적인 다각형 모양으로 간주하는 것이 현재의 응용 프로그램들의 요구에 부응할 수 있을 것이다[6]. 따라서, 본 연구에서도 공간 선택 질의의 질의 윈도우의 모양을 일반적인 다각형 모양으로 간주하였다. 또한, 앞서 언급된 바와 같이 공간 데이터는 수 천 개의 점으로 구성되어 있을 정도로 매우 복잡한 구조를 보이고 있다. 반면에, 질의 윈도우는 비록 일반적인 다각형의 형태를 취한다 하더라도 수 십 개의 점으로 구성될 수 있는 것이 일반적이다. 따라서, 이러한 관찰로부터 질의 윈도우는 공간 데이터에 비해서 훨씬 간단한 모양이라고 간주하고 있다.

현재 일반적인 다각형 형태의 질의 윈도우를 사용하는 공간 선택 질의의 정제 단계 수행을 위한 접근 방법으로는 plane-sweep이 사용되고 있다[6]. 이는 질의 윈도우와 공간 데이터가 모두 일반적인 다각형 모양이기 때문이다. 하지만, 앞에서의 관찰에서 알 수 있듯이 질의 윈도우는 공간 데이터보다는 훨씬 간단한 모양이므로, 이러한 상황에서는 복잡하고 정교한 plane-sweep 알고리즘을 사용하는 것에 개선에 여지가 있을 것으로 생각된다. 왜냐하면, plane-sweep은 그 수행을 위해서 많은 선행 작업이 요구되므로, 두 다각형이 모두 복잡한 다각형일 때에 더욱 유리하기 때문이다. 본 연구에서는 이러한 상황에 적합한 정제 단계 수행 알고리즘으로서 PBST(Partition Based Sequential Testing)를 제안하고 있으며 이 기술은 progressive approximation의 일종인 PEC(Partial Enclosed Circle)에 기초하고 있다[1].

본 논문의 구성은 다음과 같다. 2장에서는 공간 선택 질의의 수행과 관련된 기초적인 연구들을 간단히 기술하고 있으며, 3장에서는 본 논문에서 제안하는 PBST와 PEC에 대해서 설명하고 있다. 4장에서는 기존의 방법과의 비교실험 결과를 보이고 있으며, 5장에서 결론을 맺고 있다.

2. 관련 연구

이 장에서는 본 연구의 기저로서 공간 선택 질의를 수행하기 위한 기본적인 과정들에 대해서 설명하고 있으며, 이는 [5]에서 제시된 여과-정제 전략에 기초하고 있다. 또한, 본 연구와 관련된 연구로서 progressive approximation과 plane-sweep algorithm을 소개하고 있다.

2.1 공간 선택 질의의 수행 과정

하나의 공간 객체는 수 개의 점에서부터 수 천 개의 점으로 구성될 수 있으며 각각의 크기도 매우 다양하다.

또한, 공간 객체 하나의 크기가 디스크의 한 페이지 크기보다 큰 경우도 많다. 이러한 특성을 지니는 공간 데이터를 효과적으로 처리하고 이들의 지역성(locality)을 유지하기 위해서 SAM을 사용하며, R-tree의 일종인 R*-tree가 널리 사용된다. R*-tree는 인덱스를 구축하는데 있어서 데이터 자체의 정확한 모양을 사용하지 않고 데이터의 conservative approximation을 이용하며, MBR(Minimum Bounding Rectangle)이 주로 사용된다. 이러한 MBR에 기초한 SAM을 이용해서 공간 선택 질의를 수행하는 과정은 다음의 두 단계로 나누어 진행될 수 있다[5].

여과 단계: 질의 윈도우와 겹치는 MBR을 가지는 객체들을 후보로서 찾는다.

정제 단계: 여과 단계를 통해 검출된 후보 객체들이 실제로 질의 윈도우와 겹치는지를 각 객체들의 실제 모양을 이용해서 검사한다.

그림 1에서 알 수 있듯이 여과 단계는 질의 윈도우와 SAM에 있는 각 객체의 MBR이 겹치는 지를 검사하는 단계로서 비교적 쉽게 수행될 수 있다. 반면에, 정제 단계는 질의 윈도우와 객체가 실제로 겹치는 지를 검사하기 위해서 각 객체의 정확한 모양 정보를 디스크로부터 읽어와야 한다. 또한 이 검사를 위해서 plane-sweep 알고리즘과 같은 정교하고 복잡한 알고리즘을 이용해야만 한다. 따라서, 정제 단계는 여과 단계보다 더 많은 시간이 소요된다[6]. 여과 단계에서 선택된 후보 객체들은 최종 결과의 수퍼셋(superset)이다. 따라서, 이러한 후보 객체들이 실제 결과에 속하는지 여부를 검사하는 정제 단계는 비록 여과 단계보다 많은 비용이 요구된다 하더라도 올바른 결과를 얻기 위해서 필수 불가결한 단계이다. 그러므로 공간 선택 질의를 보다 효과적으로 수행하기 위해서는 이러한 정제 단계에서 소요되는 비용을 줄이는 것이 중요한 관건이라 할 수 있다.

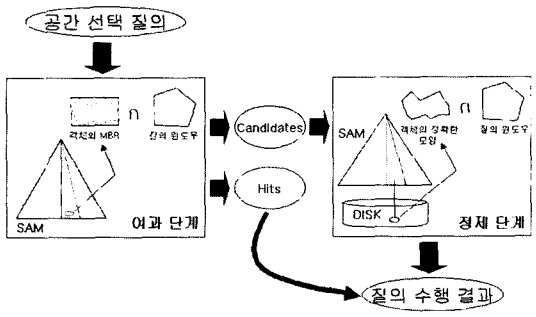


그림 1 공간 선택 질의의 수행 과정

2.2 Progressive Approximation

한 공간 객체의 Progressive approximation은 그 객체에 의해 최대로 둘러싸이는 도형을 의미한다. 만약 두 객체의 progressive approximation이 겹친다면, 두 객체도 역시 겹친다고 결론지을 수 있다. 따라서, 이 방법은 여과 단계로부터 검출된 지원자 객체들로부터 true-hit을 미리 검사하는데 간접적으로 사용될 수 있다. Progressive approximation은 conservative approximation에 비해서 미리 계산하는 비용이 크기 때문에 비교적 간단한 MEC(Maximum Enclosed Circle)와 MER (Maximum Enclosed Rectangle)이 사용된다. 그림 2는 MEC와 MER의 예를 보이고 있다.

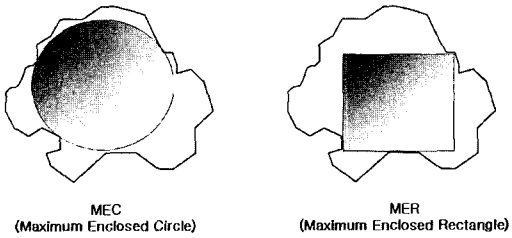


그림 2 Progressive approximation의 예

이러한 progressive approximation을 사용하여 검사를 하는 경우에, 검사가 성공하면 30% 정도의 true-hit을 미리 검출해 낼 수 있다. 따라서 검사가 실패한 나머지의 경우에 대해서만 객체의 정확한 모양을 이용하는 정제 단계의 과정을 수행하면 된다. MEC와 MER을 R*-tree의 노드에 저장하기 위해서는 각각 3, 4개의 인수를 저장하기 위한 공간이 더 필요하게 된다. 또한, progressive approximation을 미리 계산하는 데에도 추가적인 시간이 요구된다. 하지만, 이러한 공간과 계산 시간의 추가적인 요구에도 불구하고 30%의 true-hit을 미리 검출해 내는 것은 매우 효과적인 결과라 할 수 있다[1].

2.3 Plane-Sweep Algorithm

Plane-Sweep 알고리즘은 선분들의 집합에서 겹침 관계를 조사하기 위해서 계산 기하학의 분야에서 고안되었다[11]. 이 알고리즘은 두 다각형을 구성하는 선분들 간의 겹침 관계를 O(nlogn)의 시간 내에 조사하는 알고리즘으로 쉽게 수정될 수 있다. 단, n은 다각형을 구성하는 선분들의 합을 의미한다. 이 알고리즘은 공간 조인의 수행에서 주로 사용되었다. 하지만 질의 윈도우도 공간 객체와 같이 일반적인 다각형 모양인 상황에서는 이 알고리즘 또한 공간 선택 질의에 적용될 수 있으므로, 최선의 방법이라고 할 수 있다[6].

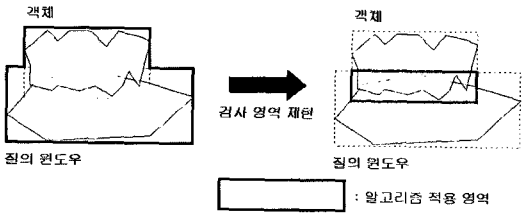


그림 3 검사 영역 제한

Plane-sweep을 이용하는 정제 단계 수행 과정은 다음과 같다.

(1) 검사 영역을 제한한다.

(2) 모든 선분들을 정렬한다.

(3) Plane-sweep 알고리즘을 수행한다.

Plane-sweep 알고리즘의 기본적인 수행 과정은 다음과 같다. 먼저 선행 작업으로서 두 다각형을 구성하는 모든 점들을 x(y) 축에 따라 정렬한다. 그 다음으로 수직(수평)의 sweep-line이 전체 데이터 공간을 좌에서 우로(위에서 아래) 이동하면서 두 다각형간의 겹침 관계를 조사하게 된다. 이러한 plane-sweep 알고리즘을 적용하기 전에 (1)의 검사 영역 제한의 작업이 선행되어야 하는데, 이는 그림 3에서 볼 수 있듯이 질의 윈도우와 공간 객체는 반드시 두 다각형의 MBR이 겹치는 부분에서만 겹칠 수 있기 때문에, 검사 작업을 전체 영역에 대해서 수행하는 것보다는 (1)의 과정에 의해서 제한된 영역에 대해서만 수행하는 것이 보다 효과적이기 때문이다. 그러므로, 검사 영역을 제한하는 (1)의 작업이 (2)보다 선행되어야 한다. 그 후, 마지막으로 plane-sweep 알고리즘이 수행된다. 그림 4는 plane-sweep 알고리즘을 이용하는 정제 과정의 전체 과정을 보여주고 있다.

분석 : 두 다각형(질의 윈도우와 공간 객체)을 구성하는 점들에 대한 순차 검색을 통해서 (1)의 작업은 완료될 수 있으므로, (1)은 O(n) 시간이 요구된다. 또한, (2)와 (3)의 작업은 각각 O(nlogn)이 요구된다. 그러므로, plane-sweep 알고리즘을 이용해서 질의 윈도우와 공간 객체의 겹침 여부를 검사하는데 걸리는 시간은 O(nlogn)임을 알 수 있다. 단, n은 질의 윈도우와 공간 객체를 구성하는 모든 점들의 합을 의미한다. 보다 정확한 분석을 위해서 n_1과 n_2가 각각 질의 윈도우와 공간 객체를 구성하는 선분(점)의 개수라고 할 때, (1)은 $O(n_1+n_2)$의 시간이 요구된다. (2)와 (3)은 각각 $O((n_1+n_2)\ \log(n_1+n_2))$의 시간이 요구된다. 그러므로, 기존에 널리 사용되는 방법은 $O((n_1+n_2)\log(n_1+n_2))$의 시간이 요구된다고 할 수 있다.

```
procedure Intersect_test_Plane_Sweep(Query, Object)
    Line_set ← extract line segments from Object and Query
               which intersect the intersection rectangle of
               the MBRs of Object and Query
    Sorted_list ← sort the vertices of Line_set according to
                  their x-coordinates
    for each point p in the Sorted_list
        do Sweep_line shifts from left to right
        if edge e is right from the Sweep_line
            insert e into Sweep_line_status
            if e intersects with its new neighbors in Sweep_line_status
                then return TRUE
        if edge e is left from the Sweep_line
            delete e from Sweep_line_status
            if the former neighbors of e in Sweep_line_status intersect
                then return TRUE
    return FALSE
end
```

그림 4 Plane-sweep을 이용하는 정제 단계 수행 과정

이와 같이 plane-sweep 알고리즘은 O(nlogn)의 시간 내에 질의 윈도우와 공간 객체의 겹침 여부를 검사하는데 응용할 수 있는 알고리즘이다. 이 알고리즘은 그 수행을 위하여 모든 선분들을 정렬하는 것과 같은 예비 작업이 요구되므로, 두 다각형이 모두 복잡한 다각형일 때 보다 효율적이다. 그러나, 앞에서 언급한 바와 같이 일반적으로 질의 윈도우는 공간 데이터에 비교했을 때 매우 간단한 모양이므로, plane-sweep 알고리즘을 사용함으로써 얻을 수 있는 이익이 줄어들게 된다. 또한, 그림 3에서 설명된 검사 영역을 제한하는 작업을 거치면 plane-sweep 알고리즘으로 검사해야 할 선분의 개수가 더욱 많이 줄어들게 된다. 이러한 관찰로부터 우리는 질의 윈도우는 공간 데이터보다 훨씬 간단한 모양이라고 할 수 있다. 따라서, 위의 복잡도(complexity)식에서 n_2의 크기가 n_1 보다 훨씬 작으므로 $n2_{의}$ 영향을 무시할 수 있다. 그러므로, 이렇게 질의 윈도우가 공간 데이터보다 훨씬 간단한 모양인 경우에는 이러한 상황에 보다 적합한 다른 정제 단계 수행 알고리즘을 고려해볼 여지가 있다. 특히, n_1과 n_2가 비슷한 정도의 크기가 아니므로 nested-loop과 같은 간단한 순차 검색을 응용한 알고리즘의 적용도 고려해 볼 가치가 있다.

3. 정제 단계 수행 전략

공간 선택 질의의 정제 단계는 여과 단계를 통해서 선택된 객체들의 정확한 모양을 이용해서 이들이 실제로 질의 윈도우와 겹치는지를 검사하는 단계이다. 공간 선택 질의의 전체 수행 과정은 앞의 그림 1에 설명되어 있다. 공간 객체는 수 천 개의 점으로 구성된 매우 복잡한 구조를 가지므로, 이러한 공간 객체를 효과적으로 검사하기 위해서는 정제 단계에서 보다 효율적인 검사 알

고리즘을 이용해서 검사하는 것이 중요하다. 더욱이 질의 윈도우의 모양이 일반적인 다각형이므로 효과적인 정제 단계의 수행이 더욱더 중요하다.

다음절에서는 PEC(Partial Enclosed Circle) approximation와 PBST(Partition Based Sequential Testing) technique을 사용하는 새로운 정제 전략에 대해 설명하고 있다.

3.1 PEC (Partial Enclosed Circle) Approximation

2.2절에서 언급한 바와 같이, progressive approximation은 여과 단계에서 30%의 true-hit을 미리 찾아낼 수 있다. 따라서 미리 찾아내지 못한 나머지에 대해서만 정확한 객체의 모양을 이용하는 정교하고 많은 시간이 요구되는 작업을 수행하면 된다. 이 approximation은 SAM의 노드에 추가적인 공간이 요구되며, 또한 이를 계산하는 데에도 추가적인 연산이 요구되므로 널리 사용되고 있지는 않다. 하지만, 이러한 추가적인 공간 사용과 계산 요구의 오버헤드를 줄일 수 있다면 매우 유용하게 사용될 수 있는 방법이다. 따라서, 본 논문에서는 이러한 progressive approximation을 정제 단계 과정에 적용하여 이러한 추가적인 오버헤드는 줄이면서 progressive approximation으로서의 기능은 충분히 할 수 있게 하는 PEC(Partial Enclosed Circle)를 제안하고 있다.

검사 영역을 제한하는 작업은 정제 단계의 작업에서 상당히 중요한 역할을 하고 있다. 그림 3에서 자세히 표현된 바와 같이, 이를 통해서 검사 영역 중의 상당 부분을 줄일 수 있으며, 또한 이 작업은 두 다각형을 구성하는 점들 전체를 한 번 순차 검색함으로써 쉽게 수행될 수 있다. 본 논문에서 제한하는 PEC는 정제 단계에서 질의 윈도우와 공간 데이터의 MBR들이 겹치는 영역 내에 존재하는 객체의 선분들을 모두 포함할 수 있도록 충분히 크게추정(progressive approximation)하고 있다.

PEC는 그림 5에서 알 수 있듯이 질의 윈도우와 공간 데이터의 overlapped MBR에 기초하고 있다. PEC의 중심은 overlapped MBR에서 공간 객체의 방향에 있는 선분 위에 위치하며, PEC의 반지름의 계산은 이 중심점으로부터 검사 영역을 제한하는 작업과 같이 진행된다. PEC approximation은 기존의 여과 단계에서 사용하던 progressive approximation에 비해서 저장 공간과 추가 연산에 대한 오버헤드가 거의 요구되지 않는데, 이는 PEC는 보다 크게 포함되도록 공간 객체를 추정(progressive approximation)하기 때문이다. PEC의 반지름은 그 중심점으로부터 그 공간 객체를 구성하는 점이나 선분까지의 최단 거리를 의미하므로, PEC의 중심과 반지름을 이용해서 질의 윈도우의 점과 선분에 대한 거리 검사가 성공한다면, 더 이상의 복잡한 검사 작업 없이 간단하게 true-hit을 검출해 낼 수 있다. 이러한 PEC approximation의 soundness는 아래의 <정리 1>에 의해 증명할 수 있다.

<정리 1>

어떤 다각형 A의 내부에 존재하는 원 C가 다른 다각형 B와 겹친다면, 다각형 A도 B와 역시 겹친다.

증명 : 모순을 보임으로써 증명할 수 있다. 만약, 다각형 A가 다각형 B와 겹치지 않는다고 가정하면, 원 C는 A의 내부에 있으므로 B와 겹칠 수가 없다. 이는 C가 B와 겹친다는 사실에 모순되므로, 따라서 위의 정리는 모순에 의해 사실임을 알 수 있다.

그림 6은 PEC에 의해서 추정(approximate)된 overlapped MBR 영역의 예를 보여주고 있다. (a)는 중심에 하나의 PEC만을 사용하는 경우이고, (b)와 (c)는 각각 MBR의 길이비에 따라 PEC를 여러 개 위치시킨 경우와, 좀더 많은 수의 PEC를 만들기 위해서 PEC의 중심들의 위치를 아래로 1/3 하향 조절한 경우이다. 또한, (d)는 PEC를 중심과 양 끝 부분에 위치시킨 경우이다. PEC의 중심점의 위치는 다양한 기준에 의해서 선택될

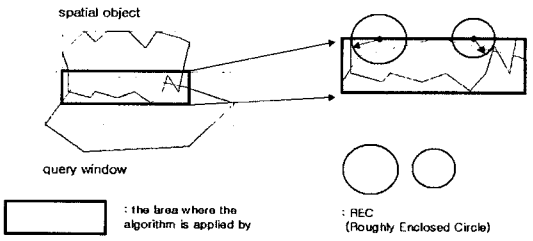


그림 5 PEC progressive approximation

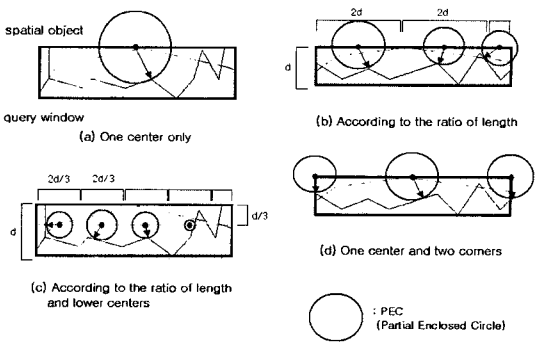


그림 6 PEC approximation의 예

수 있지만, 본 논문에서는 연산의 오버헤드를 줄이기 위해서 위와 같은 간단한 방법들을 사용하였다. 위의 모든 방법들은 실험상 비슷한 성능을 보이고 있는데, 이는 이 방법들 모두가 일종을 휴리스틱이므로 지속적인 특징(persistent property)을 보일 수가 없기 때문이다. 이러한 PEC의 중심과 위치와 개수를 이상적으로 정할 수 있는 방법에 대한 연구는 앞으로도 계속되어야 하겠다.

실제 응용에서 사용되는 데이터를 이용해서 실험해본 결과 PEC approximation은 평균적으로 약 30%의 true-hit을 미리 찾아낼 수 있으며, 실험에 대한 자세한 정보는 다음 장에 설명되어 있다. 하지만, 이러한 성능은 사용하는 데이터의 특성과 PEC의 중심점을 정하는 방법에 따라 큰 편차를 보이고 있다. 따라서, PEC의 중심점의 위치와 개수를 이상적으로 정할 수 있는 방법에 대한 연구가 성능의 지속적인 향상을 위해서 반드시 선행되어야 하겠다.

3.2 새로운 정제 전략 방법: PBST를 사용한다.

PBST(Partition Based Sequential Testing)는 PEC approximation에 기초한 순차 검색 기법이다. 앞서 언급한 바와 같이 PEC는 progressive approximation이므로 실패한 작업만 plane-sweep 검사와 같은 다음 과정이 요구된다. 만약 PEC 검사가 실패하면, 우리는 PEC가 있지 않는 다른 영역에서 질의 윈도우와 겹치는 현상이 발생할 것임을 예측할 수 있다.

그림 7에서 보면, (A) 영역은 (B) 영역보다 질의 윈도우와 겹치는 선분이 있을 가능성이 크다. 하지만, 기존의 방법에서 사용되던 plane-sweep 검사는 x나 y 좌표에 따라서 한 방향으로 진행되므로 이러한 상황을 이용하기가 어렵다. 하지만, 분할된 영역(partitioning region)에 기초하여 질의 윈도우와 공간 데이터의 각 선분들에 대한 순차 검색을 이용한다면, 이러한 상황을 잘 이용할 수가 있다.

본 논문에서 제안하는 PBST(Partition Based Sequential Testing) 방법은 이러한 partitioning region을 이용한다. PBST는 질의 윈도우와 공간 데이터를 구성하는 선분들의 겹침을 순차적으로 검사하지만, 단순히 순차적으로 검사하는 것이 아니라 PEC approximation검사로부터 검출된 영역에 따라서 순서를 정해서 검사를 실행한다. 따라서, 위의 예에서는 먼저 (A) 영역을 검사하고 나중에 (B) 영역을 검사하게 된다. PBST의 전반적인 수행 과정은 다음과 같다.

(1) 검사 영역을 제한한다.
(2) PEC approximation 검사를 한다.
(3) 실패하면, PEC의 영역이 아닌 부분에 있는 선분들에 대해서 순차적으로 겹침을 검사한다.
(4) 실패하면, PEC의 영역 내에 있는 선분들에 대해서 순차적으로 검사한다.

분석 : (1)의 과정은 두 다각형에 대한 순차 검색을 통해서 O(n)에 수행할 수 있다. 단, n은 두 다각형을 구성하는 모든 점들의 개수의 합이다. (2)의 과정은 PEC의 개수에 질의 윈도우를 구성하는 점의 개수를 곱한 만큼의 시간이 요구된다. 따라서 크게 추정해서(rough) O(n) 시간이 요구된다. (3)과 (4) 과정은 질의 윈도우와 공간 데이터가 비슷한 개수의 점으로 구성되어 있다고 가정했을 때, 합쳐서 $O(n_2)$ 만큼의 시간이 걸린다고 할 수 있다. 하지만, 앞의 3.2절에서 언급한 바와 같이, 검사 영역을 제한하는 작업을 거친 후에 검사가 필요한 질의 윈도우의 점의 개수는 공간 데이터의 점의 개수보다 훨씬 작으므로, (3)과 (4)의 작업을 수행하는데 걸리는 시간은 $O(n_2)$ 보다는 작을 것으로 예상된다. 다음 장에 자세히 소개될 실험에 의하면, 질의 윈도우를 구성하는 점의 개수가 20개 내외인 경우에 대해서는 본 논문에서 제안하는 PBST 방법이 plane-sweep을 사용하는 방법보다 향상된 성능을 보이고 있다. 다음의 그림 8은 PBST 방법을 사용하는 정제 단계의 전체 수행 과정을 보여주고 있다. 먼저 질의를 구성하는 선(line)들과 PEC가 교차하면 두 객체가 겹침(intersection)을 의미하

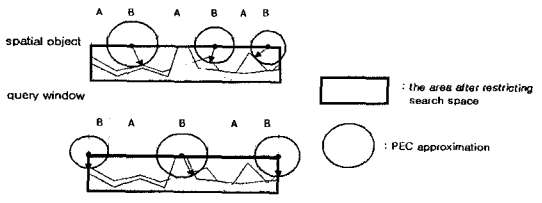


그림 7 PBST에서의 구간분할의 예

```
procedure Intersect_test_PBST(Query, Object)
    Line_set ← extract line segments from Object and Query which intersect
               the intersection rectangle of the MBRs of Object and Query
    PEC_set ← make the PECs with Line_set according to the distance
               between the center point of PEC and Line_set
    PEC_in, PEC_out ← classify the edges of Query and Object in Line_set
               according to the region of PEC_set
    for each edge e_Query of Query in the Line_set
      if e_Query intersects PEC_set
          then return TRUE
    for each edge e_Query of Query in the Line_set and PEC_out
      for each edge e_Object of Object in the Line_set and PEC_out
          if e_Query intersects e_Object
              then return TRUE
    for each edge e_Query of Query in the Line_set and PEC_in
      for each edge e_Object of Object in the Line_set and PEC_in
          if e_Query intersects e_Object
              then return TRUE
    return FALSE
end
```

그림 8 PBST 방법의 정제 단계 수행 과정

는 True임을 알 수 있다. 다음으로 만약 질의 윈도우가 PEC와 교차하지 않는다면, 그림 7에서와 같이 분할(partitioning)된 영역 중에서 PEC들이 속해있지 않은 영역에 있는 객체의 선들을 먼저 질의 윈도우와 겹치는지를 검사한다. 이 검사에도 겹침이 발견되지 않으면 마지막으로 PEC의 내부 영역에 있는 객체의 선들을 검사하게 된다. 그림 9는 두 방법의 수행 과정의 차이를 보여 주고 있다.

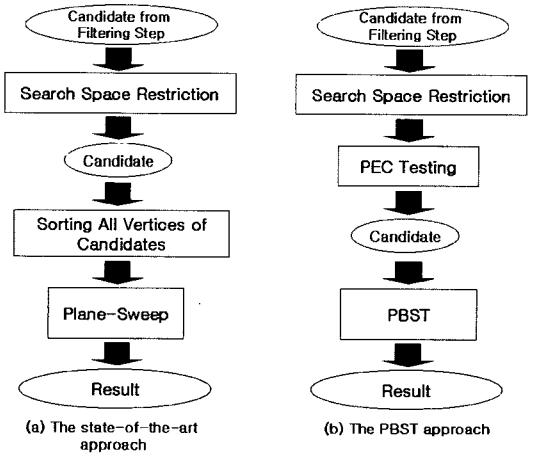


그림 9 두 방법의 수행 과정의 비교

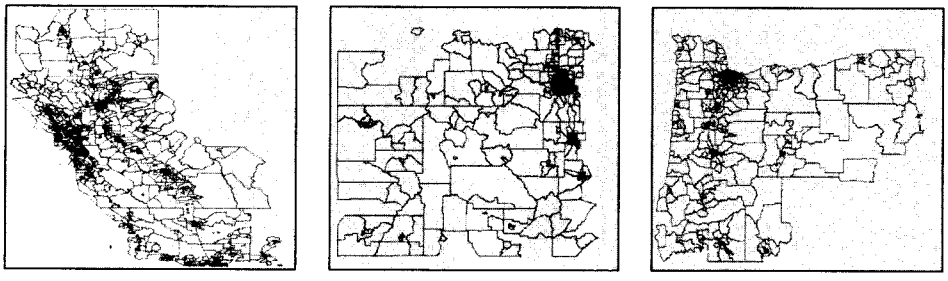
4. 실험 평가

이 절에서는 일반적인 다각형 모양의 질의 윈도우를 사용하는 공간 선택 질의의 경우에, 본 연구에서 제안하는 PBST, PEC 방법론이 기존의 방법에 비해서 어느 정도의 성능을 보이고 있는지를 예제를 통해서 설명하고 있다. 이를 위해서, 실제 응용에서 사용되는 지도 데이터를 이용해서 실험한 결과를 보이고 있다. 4.1절에서는 실험 환경을 제시하였으며, 다음절부터 실험 결과와 성능 분석을 제시하였다.

4.1 실험 Setup

Data Sets: 본 논문에서는 실험 데이터로서 TIGER/Line Files[12]에서 캘리포니아 지역의 CTBNA(Census Tracks and Block Numbering Areas) 데이터를 사용하였으며, 이로부터 3개의 test data set을 구성하였다. Test data set 1은 각각이 4개에서 4586개의 점으로 구성된 2878개의 다각형으로 구성되어 있으며, 각 다각형들의 평균 구성 점의 개수는 168개이다. 또한, 100개 이하의 점으로 구성된 다각형 객체들의 비율은 75.12%이다. 그림 10은 각각의 test data set에 대한 이러한 특질들을 요약하고 있으며 또한, 각 data set의 실제 모양을 보여주고 있다. M_{avg}, M_{min}, M_{max}는 각각 각 test data set을 구성하는 객체들의 평균, 최소, 최대의 구성 점의 개수를 나타내며, # objects는 각 test data set을 구성하는 다각형 객체들의 개수를 의미한다.

test data set	# objects	distribution of data (%)		M_{avg}	M_{min}	M_{max}
		under # of 100	over # of 100			
Set 1	2878	75.12	24.88	168	4	4586
Set 2	838	70.52	29.48	215	4	2809
Set 3	689	47.61	52.39	380	5	5382



(a) Set 1 (b) Set 2 (c) Set 3

그림 10 test data sets의 특징 분석

Query Sets: 실험에서 사용할 공간 선택 질의의 질의 윈도우를 위하여, 각각의 test data set에 사용할 test query set을 구성하였다. 이 test query set은 각각 4, 8, 12, 16, 20개의 점으로 구성된 100개의 임의의 질의 윈도우들로 구성되어 있으며, 이들은 전제 데이터의 영역에서 0.3%~10%에 해당하는 질의 영역을 이룬다.

Performance Metric: 본 논문에서 제안하는 방법의 우수성을 평가하기 위한 performance metric으로서 average relative estimation을 사용하였다. 비교 평균 측정은 다음과 같이 정의된다.

$$\frac{\left(\begin{array}{c}\text{measured cost of the state}\\ -\text{of}-\text{the}-\text{art approach}\end{array} - \begin{array}{c}\text{measured cost of}\\ \text{the new approach}\end{array}\right)}{\text{measured cost of the state}-\text{of}-\text{the}-\text{art approach}}$$

기존 방법의 측정된 비용은 각 test data set에서 모든 test query set을 수행한 시간의 합으로 측정된다. 실험을 위해 공간 데이터에 대한 R*-tree 인덱스를 고려하였다.

Run-time Environment: 실험은 384 MB의 메인 메모리를 지니는 SUN Ultra II 170 MHz 기계에서 진행되었으며, SUN OS 5.5.1을 사용하였다. 인덱스로서 사용한 R*-tree의 한 페이지의 사이즈는 1 KB이다.

4.2 성능 분석

이 실험을 통해서 우리는 임의 모양의 질의 윈도우를 이용하는 공간 선택 질의를 수행하는데 사용될 수 있는 두 방법의 성능을 평가하고 있다. 특히, 질의 윈도우의 모양이 공간 데이터에 비해서 매우 간단한 경우에는 우리가 제안하는 PBST 방법이 기존의 방법에 비해서 우수함을 결과를 통해 유추할 수 있다. 두 방법은 모두 인덱스로서 R*-tree를 사용하였으며 이 인덱스를 이용하여 여과 단계에서 지원자 객체들을 결정하고 있다. 정제 단계부터는 각각의 방법을 이용하여서 최종 결과를 얻고 있다.

실험에서 성능의 척도로서 사용되는 수행 시간은 모든 질의의 수행을 완료한 때까지의 시간으로 측정된다. 그림 14의 실험 결과로부터 알 수 있듯이, 전체 수행 시간에 비해서 여과 단계가 차지하는 비중은 1~3%에 해당될 정도로 작으므로, 정제 단계에서만 소요되는 시간 대신에 전체 질의 수행 시간을 performance measure로서 사용하였다.

그림 11은 test data set 1에 대해서 4개부터 20개의 점으로 구성된 다각형 질의 윈도우를 이용하여 공간 선택 질의를 수행한 성능 비교 결과이다. 먼저, 질의 윈도우가 전체 데이터 영역에서 차지하는 영역의 변화에 따른 성능 변화를 살펴보면, PBST 방법이 전반적으로

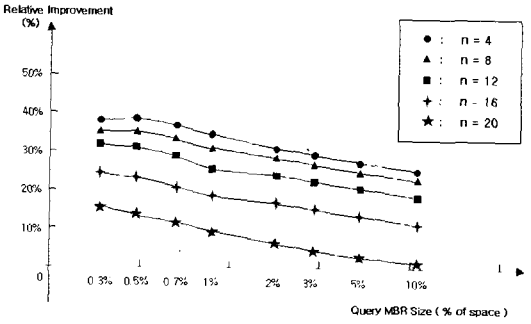


그림 11 test data set 1에 대한 성능 평가

20% 정도의 성능 향상을 보이고 있으며, 1% 미만의 작은 범위의 질의 윈도우에 대해서는 30% 이상의 성능향상을 보이고 있다. 이러한 실험 결과에 대한 원인 분석은 다음과 같다.

질의 윈도우의 크기가 커짐에 따라, 많은 점으로 구성된 데이터를 처리해야 할 가능성이 더욱 커지며, 검사 영역을 제한하는 작업을 마친 후에도 정확한 모양을 이용해서 검사해야 할 점들이 많이 남아있게 된다. 또한, 질의 윈도우의 크기가 커질수록 여과 단계로부터 여과된 지원자 객체들 중에서 실제로는 질의 윈도우와 겹치지 않는 것으로 판별되는 것들이 많아짐에 따라, PEC를 계산하는 데에 따른 오버헤드도 가중된다. 따라서, PBST 방법이 성능이 plane-sweep 기술을 사용하는 기존의 방법에 비해서 점점 성능이 저하되는 현상을 볼 수 있다. 그러나, 대부분의 응용에서 사용되는 공간 선택 질의의 질의 윈도우는 전체 데이터 공간의 1%를 넘지 않는 것이 일반적이므로, 본 논문에서 제안하는 PBST 방법이 기존의 방법보다 약 20% 정도 우수하다고 할 수 있다.

질의 윈도우의 complexity에 따른 성능을 살펴보면, 질의 윈도우를 구성하는 점의 개수가 늘어남에 따라, PBST 방법의 성능이 크게 저하되는 것을 볼 수 있다. 이는 질의 윈도우를 구성하는 점의 개수가 늘어나면 PBST 방법은 $O(n_2)$의 complexity를 갖게 되기 때문이다. 반면에, plane-sweep 기술을 사용하는 기존의 방법은 원래 이러한 상황에 적합하기 때문에, 점점 더 향상되는 성능을 보이고 있다. 그러나, 질의 윈도우를 구성하는 점의 개수가 16개 미만인 경우에는 PBST를 사용하는 새로운 방법이 기존의 방법보다 약 20% 정도 우수함을 확인할 수 있다.

결론적으로, 질의 윈도우를 구성하는 점의 개수가 20개 미만이고 질의 윈도우가 전체 데이터의 영역에서 차

지하는 비중이 기존의 응용에서 많이 사용되는 것처럼 1% 내외일 경우에는, PBST를 사용하는 새로운 방법이 plane-sweep을 사용하는 기존의 방법보다 약 20% 정도 우수함을 알 수 있다. Test data set 2와 3에 대한 실험 결과도 이와 비슷한 경향을 보이고 있으며, 그림 12와 그림 13에 제시되어 있다.

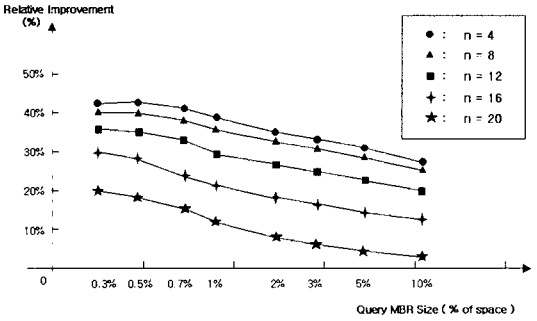


그림 12 test data set 2에 대한 성능 평가

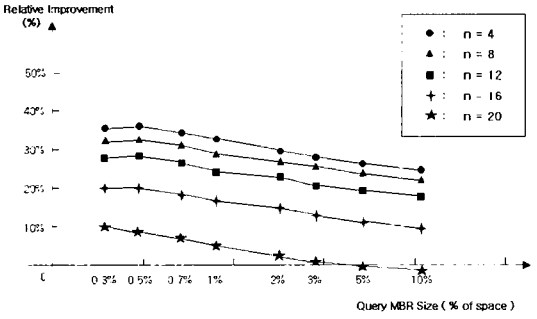


그림 13 test data set 3에 대한 성능 평가

지도 데이터에 따른 성능의 특성에 대해서 살펴보면, test data set 2에서 얻을 수 있는 성능 향상이 다른 두 set에서 보다 많음을 알 수 있다. 이는 test data set 1,3이 test data set 2보다 많은 수의 점으로 구성된 복잡한 다각형 객체로 구성되어 있기 때문이다. 또한, test data set 3는 세 개의 set 중에서 가장 복잡한 객체들로 구성되어 있으므로, 가장 적은 성능의 향상을 보이고 있다. 따라서, 사용하는 지도 데이터가 보다 적은 수의 점으로 즉, 보다 간단한 객체로 구성되어 있고 객체의 개수 또한 적을수록 PBST는 보다 나은 성능 향상을 보임을 알 수 있다.

또한, 그림 14는 기존의 방법과 새로 제안하는 방법의 전체 성능을 비교 분석한 결과를 나타내는데, 두 방법의

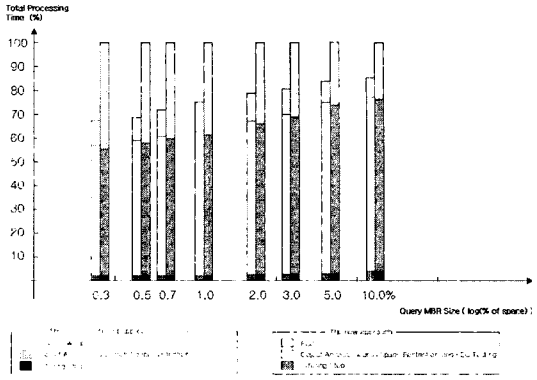


그림 14 전체 성능 분석

각각의 단계에서 소요되는 시간을 보여주고 있다. PEC approximation testing을 위한 계산 비용이 객체를 접근하거나 검사 영역을 제한하는 작업에 비교했을 때, 아주 작은 오버헤드임을 알 수 있다. 그러므로, PEC testing은 적은 계산 overhead로 많은 수의 hit을 미리 찾을 수 있음을 알 수 있다.

전체적으로 PBST 방법을 사용하는 새로이 제안하는 방법이 plane-sweep 기술을 사용하는 기존의 방법에 비해서 약 20% 정도의 성능 향상을 보이 있다. 물론, 질의 윈도우는 20개 미만의 점으로 구성된 비교적 간단한 모양으로 구성되어 있으며, 질의 윈도우가 차지하는 영역이 전체 데이터 영역에서 1% 내외를 차지할 정도로 작은 질의 윈도우를 사용할 경우이다. 비록, 사용하는 지도 데이터의 특성에 따라 다소 성능의 차이를 보이기는 하지만, 일반적으로 이와 같이 언급할 수 있겠다.

5. 결 론

공간 선택 질의에 사용되는 질의 윈도우로는 직사각형이 주로 사용된다. 하지만, 공간 선택 질의의 윈도우로는 직사각형이 아닌 일반적인 다각형 모양도 가능하다. 또한, 최근에는 GIS 등과 같은 응용 프로그램들이 성능 향상으로 인해 보다 많은 공간 데이터를 다룰 수 있게 됨에 따라, 여러 다양한 종류의 응용도 많이 등장하고 있다. 따라서, 직사각형뿐만 아니라 임의의 다각형 형태의 질의 윈도우에도 적합한 정제 단계 수행 전략에 대해 고려해 볼 필요가 있다. 이러한 전략으로는 질의 윈도우와 데이터가 모두 임의의 다각형이므로 기존의 공간 조인에서와 같이 plane-sweep 알고리즘을 이용하는 방법이 가장 일반적인 방법이다. 하지만, 공간 데이터와 질의 윈도우의 특성을 관찰해보면, 일반적으로 질

의 윈도우가 공간 데이터보다 훨씬 간단한 모양으로 구성되어 있음을 알 수 있다. 하지만, 이 plane-sweep알고리즘은 두 대상이 모두 복잡한 임의의 다각형일 때 적합한 알고리즘이므로, 데이터에 비해 질의 윈도우가 훨씬 간단한 상황에는 개선의 여지가 있다.

따라서, 본 연구에서는 일반적인 다각형 형태의 공간 선택 질의를 위한 정제 단계 수행 방법으로서 PEC (Partial Enclosed Circle) approximation에 기반하는 PBST(Partition Based Sequential Testing) 방법을 제안하고 있으며, 기존의 방법과의 성능을 실세계에서 사용되는 지도 데이터를 이용해서 비교평가하고 있다. 비록 실험에 사용되는 실제 지도 데이터의 특성에 따라 약간의 편차가 존재하지만, 일반적으로 볼 때, 질의 윈도우를 구성하는 점의 개수가 약 20개 이하인 경우에는, PBST를 사용하는 새로운 방법이 plane-sweep을 사용하는 기존의 방법보다 약 20% 정도의 성능 향상을 보이고 있음을 알 수 있다. 따라서 일반적인 다각형 형태의 질의 윈도우를 이용한 공간 선택 질의를 수행할 때, 사용되는 응용 프로그램이 약 20개 이하의 점으로 구성되는 질의 윈도우를 주로 사용하는 일반적인 경우라면, 기존의 방법보다는 본 논문에서 제시하는 PBST를 사용하는 방법이 보다 효과적이라 할 수 있다.

참 고 문 헌

[1] T. Brinkhoff, H.P. Kriegel, R. R. Schnieder, and B. Seeger, "Multi-Step Processing of Spatial Joins," ACM SIGMOD conference pp197-208, 1994.

[2] N. Beckmann, H.-P. Kriegel, R. Schnieder, and B. Seeger, "The R*-tree: an efficient and robust access method for points and rectangles," ACM SIGMOD conference pp322-331, 1990.

[3] I. Kamel and C. Faloutsos, "Hilbert R-tree: An Improved R-tree Using Fractals," VLDB pp500-509, 1994.

[4] G. Evangelidis, D. Lomet, and B. Salzberg, "The hb-tree: A modified hB-tree supporting concurrency, recovery, and node consolidation," VLDB pp551-561, 1995.

[5] J. A. Orenstein, "A Comparison of Spatial Query Processing Techniques for Native and Parameter Space," ACM SIGMOD conference pp343-352, 1990.

[6] A. Aboulnaga and J. F. Naughton, "Accurate Estimation of the Cost of Spatial Selections," IEEE ICDE pp123-134, 2000.

[7] T. Brinkhoff, H. Kriegel, and B. Seeger, "Efficient Processing of Spatial Joins Using R-trees," ACM SIGMOD conference pp237-246,1993.

[8] M. L. Lo and C. V. Ravishankar, "Spatial Hash-Joins," ACM SIGMOD conference pp247-258, 1996.

[9] J. M. Patel and D. J. DeWitt, "Partition Based Spatial-Merge Joins," ACM SIGMOD conference pp259-270, 1996.

[10] Y. W. Huang, M. Jones, and E. A. Rundensteiner, "Improving Spatial Intersect Join Using Symbolic Intersect Detection," Symposium on Large Spatial Databases(SSD) pp165-177, 1997.

[11] M. I. Shamos and D. J. Hoey, "Geometric Insertion Problems," IEEE Symposium on Foundations of Computer Sciences(FOCS) pp208-215, 1976.

[12] Bureau of the Census, "TIGER/Line Files, 1995: Technical Documentation," 1996.

유 준 범

1999년 홍익대학교 컴퓨터공학과 학사. 2001년 KAIST 전자전산학과 전산학전공 석사. 2001년~현재 KAIST 전자전산학과 전산학전공 박사과정. 관심분야는 소프트웨어공학, 안전성분석, 정형기법

최 용 진

1997년 성균관대학교 정보공학과(학사). 1999년 한국과학기술원 전산학과(석사). 1999년~현재 한국과학기술원 전산학과 박사과정 재학 중. 관심분야는 공간데이타베이스, GIS, 시공간데이타베이스

정 진 완

정보과학회논문지 : 데이타베이스
제 30 권 제 1 호 참조