R^+-tree를 이용한 연결성에 관한 질의 처리 방법

강 명아, 이 기준
부산대학교 전자계산학과

A Spatial Query Processing Technique for Connectivity Using R^+-tree

Myoung-A Kang, Ki-Joune Li
Department of Computer Science Pusan National University
e-mail : {makang, lik}@spatios.cs.pusan.ac.kr

요 약

지리 정보 시스템의 응용 분야의 확대로 인해 위상적인 질의처리에 관한 필요성이 점차 커지고 있으나, 아직까지는 그에 관한 연구가 거의 이루어지지 않고 있는 실정이다. 본 논문에서는 위상적 질의들 중에서 가장 기본이 되는 연결성에 관한 질의를 처리하는 방법을 제공한다. 추가의 위상적인 정보없이 객체들의 지리적인 정보만을 이용함으로써 위상적인 질의 처리를 위한 부가적인 경비를 없앤다. 그리고, R^+-tree 인덱스 기법을 사용하여 처리해야 할 대상 객체들을 걸러내는 여과 과정을 거침으로써 디스크 참조 횟수를 줄인다. 이 방법의 성능과 성능에 영향을 미치는 요인의 관계에 대해서 연구한다.

1. 서론

지리 정보 시스템에 있어서 사용되는 질의의 종류는 다양하다. 질의는 크게 기하학적인 질의와 위상적인 질의로 나눌 수 있다. 현재 지리 정보 시스템의 질의 처리에 관한 연구는 주로 기하학적인 질의에 관한 것이 대부분이다. 서로 겹치는 다각형을 찾는 질의나 선분들의 교차점을 구하는 질의등 공간적인 데이타의 위치나 크기, 모양등의 기하학적인 성질을 주로 이용하였다.

최근 들어 지리 정보 시스템의 응용 분야의 확대로 인해 기하학적인 질의뿐만 아니라, 연결성이나 근접성, 최단거리등을 결정하는 위상적인 질의 처리에 대한 연구도 필요하게 되었다. 파이프라인 네트워크나 전력배전등을 위한 시스템은 위상적인 질의 처리를 요구하는 대표적인 예이다.

본 논문에서는 위상적인 질의들 중 가장 기본이 되는 연결성을 결정하는 질의처리에 대해서 알아본다. 예를 들면,

" 폭우로 인해 A 지역이 침수되었을 때 갑의 전화선은 을전화국에 연결되어 있는가 "

라는 질의에 대해 참인지 거짓인지 또는 연결 경로를 알기 위해 추가의 위상적인 정보없이 기존의 지리적인 정보만을 가지고 연결성를 결정하는 방법을 제안한다. 질의 처리의 성능을 높이기 의해 대부분 인덱스 기법을 사용하는데, 본 방법에서는 지리적인 질의 처리의 효율을 높이기 위해 많이 이용되는 R^+-tree 인덱스 기법을 그대로 사용하여 지리적인 질의 처리와 위상적인 질의 처리에 쓰이는 인덱스 기법에 통일성을 가질 수 있도록한다.

본 논문의 구성은 다음과 같다. 2장에서는 연결성을 결정하는 데 사용되는 기존의 방법들과 그 문제점에 대해 알아보고, 3장에서는 본 논문의 주제인 R^+-Tree를 이용하여 연결성을 결정하는 방법과 기본 개념에 대해 알아본다. 그리고 4장에서 본 논문에서 제시한 방법의 성능과 성능에 영향을 미치는 요인들에 대해 알아보고, 끝으로 5장에서는 결론과 향후 연구과제에 대해 알아본다.

2. 기존의 접근 방법

지금까지 지리 정보 시스템에서 위상적인 질의 처리연구는 거의 이루어지지 않았는데. 기존의 방법들을 다음과 같이 크게 세가지로 나누어 볼 수 있다.

첫째, 기존의 알고리즘상에서 존재하는 깊이 우선 탐색이나 넓이 우선 탐색등 그래프 순회 알고리즘에 대한 응용방법이다.[2] 현실적으로 처리해야 할 객체수가 많아서 주 메모리에 한번에 다 올라갈 수 없는 지리 정보 시스템에서 이와 같이 디스크 참조 횟수를 고려하지 않는 방법은 상당히 비효율적이다.

둘째, 이행적 폐쇄(transitive closure)를 계산하는 사용하는 방법이다.[3] 일단 이행적 폐쇄 정보가 만들어진 다음에는 질의 처리시 디스크 참조를 많이 하지 않기 때문에 시간이 적게 걸리는 장점이 있다. 그러나, 이행적 폐쇄 행렬을 위한 추가의 메모리가 필요하고, 삽입이나 삭제가 자주 일어나는 환경에서는 이 정보를 구성하는 시간에 대한 과부하를 무시할 수 없게된다. 셋째, [1]에 의해 제안된 대규모 그래프에서의 탐색기법을

이용한 방법이다. 질의의 대상이 되는 객체들을 노드와 링크로 이루어진 대규모 그래프로 보는 방법으로 효과적이고, 주목할 만한 방법이다. 그러나, 위상적인 추가의 정보를 필요로 하지 않고, 질의를 처리하려는 본 논문의 연구와는 접근 방향이 다르다.

지금까지 살펴본 방법들의 문제점을 극복하고, 질의 처리의 성능을 높이기 위해 본 논문에서는 다음의 세 가지를 기본 방침으로 한다.

첫째, 질의 처리시 성능에 가장 많은 영향을 미치는 디스크 참조 횟수를 고려한다.

둘째, 추가의 위상적인 정보없이 객체의 지리적인 정보를 그대로 이용함으로써 추가의 정보 유지에 드는 비용을 없앤다.

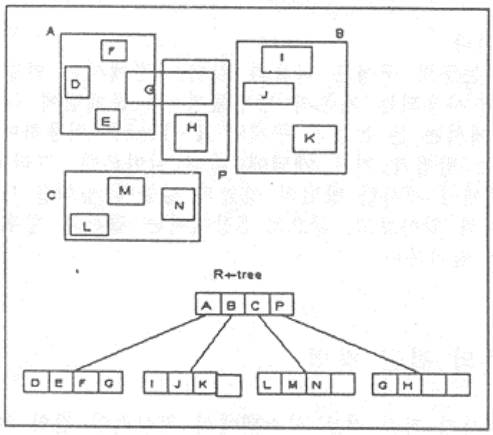
셋째, 지리 정보 시스템에서 널리 쓰이는 공간적인 인덱스 기법중 하나인 R* - Tree를 그대로 이용한다.

3. R*-Tree와 연결성(connectivity) 결정

R*-tree는 R-tree의 중간 최소경계사각형(MBR : Minimal bouding Rectangle)들이 겹치게 됨으로써 발생하는 탐색시의 과부하를 없애기 위해 최소경계사각형간의 겹침을 피한 방법이다.[4] 이는 최소경계사각형들의 겹침을 최대한 줄임으로써 효과적인 여과 과정을 수행할 수 있게 한다.

R*-tree의 한 중간 노드는 (rect, children)으로 나타나는데 rect는 이 노드의 최소경계사각형을 가리키고, children은 자식 노드의 pointer뿐만 아니라 자식 노드의 최소경계사각형에 대한 정보도 함께 가지고 있다.

다음 그림1은 R*-tree의 한 예이다.



[그림1] R*-tree의 예

이렇게 주어진 R*-tree를 이용하여 처리하려고 하는 연결성 결정 질의은 다음과 같은 예로서 나타낼 수 있다.

" A객체와 B객체는 서로 연결되어 있는가 "

질의 처리 결과는 "연결되어 있다", 또는 "아니다"이며 연결되어 있을 경우 그 경로도 알 수 있다.

질의 처리를 위한 방법의 기본적인 개념은 [8]에서 제안된 공간 조인 처리 방법과 유사하다. 조인 처리시 질의 처리를 두 단계로 나누는데, [8]

단계 1] 여과 단계(filtering step) :
R*-Tree의 최소경계 사각형에서 조인할 가능성이 없는 최소경계사각형을 미리 제거시킴으로써 실제 조인을 수행할 대상 객체의 수를 줄인다.

단계 2] 정제 단계(refinement step) :
여과 단계를 거친 객체들을 대상으로 실제 조인 연산을 수행한다.

여기서 각 최소경계사각형은 R*-tree의 노드에 대응되는데, 일반적으로 한 노드는 한 디스크 페이지에 저장된다. 따라서, 이 공간 조인 처리 방법의 성능은 제거되는 최소경계사각형, 즉 노드의 수에 의해 측정된다. 제거되는 최소경계사각형의 수가 많으면 불필요한 디스크 페이지의 참조를 그만큼 적게 하게된다.

일반적으로, 지리 정보 시스템의 성능을 좌우하는 가장 큰 요인은 디스크 참조 횟수이다. 여과 과정에서 가능성이 없는 것들을 제거하여 조인 연산의 대상 객체수를 줄임으로써 디스크 참조 횟수를 줄인다.

이와 같은 이유에서, 본 논문에서 제시하는 방법도 R*-Tree를 이용하여 불필요한 최소경계사각형을 제거하는 여과 과정을 두어 디스크 페이지 참조 횟수를 줄인다. 연결성 결정을 위한 질의 처리 방법은 다음과 같다.

단계 1] 여과 단계 :
R*-Tree에서 라인 객체가 속한 MBR과 서로 연결된(connected component) MBR을 찾는다.

단계 2] 정제 단계 :
여과 단계를 거친 MBR내의 라인 객체들이 연결성 여부를 묻는 대상 라인들을 서로 연결하고 있는 지 조사한다.

두 라인을 연결하는 경로가 될 수 있는 라인들의 MBR들은 서로 연결 요소(conneced component)이다. 따라서, 연결 여부를 묻는 대상 객체들을 잇는 객체인지 아닌지의 연산을 수행할 객체들을 걸러내는 방법은 연결 요소를 찾는것이다. 이 방법을 알고리즘으로 나타내면 알고리듬 1과 같다.

앞서 설명한 바와 같이 본 알고리즘은 여과 단계와 정제 단계로 나누어져 있다. 그러나, 본 논문에서는 여과 단계를 거친 후보들에 대한 알고리즘은 생략할다. 그 이유는 이 알고리즘은 기존의 알고리즘이 그대로 사용될 수 있기 때문이다.

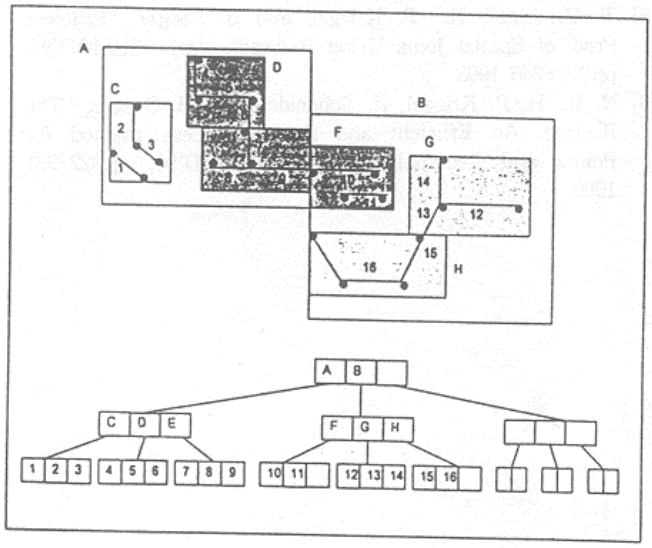
```
Procedure Connectivity(R:Root Node of R*-Tree,  A,B:object)
// find connected path from A to B //
        CCM <- Filtering ( R->children, A, B )
        Path <- Refinement ( CCM, A, B)
End Procedure
```

```
Procedure Filtering(Nodes, A, B, level )
// remove unnecessary nodes //

        CandidateSet <- {}
        for each node in Nodes
                if n is connected with A then
                    CandidateSet <- CandidateSet+{n}
        end for

        if B is not contained by CandidateSet then
                return FAIL

        if( level=leaf level ) then
                return    CandidateSet
        else // intermediate node //
                ChildrenSet <- set of children of
                                        each node inCandidateSet
                return
                        Filtering(ChildrenSet,A,B,level+1)
End Procedure
```



[그림 2] 연결성 질의 처리를 위한 R*-tree의 예

본 논문에서 제시된 Connectivity 알고리즘은 그림 2의 예로 설명된다. 즉 예를 들어 " 선분 1과 선분 12가 연결되어 있는가 " 라는 질의가 주어졌다고 하자. 그러면 함수 Filtering에서 선분 1과 선분 12가 속한 최소경계사각형과 연결된 것을 찾는다. 먼저 {A, B}가 연결된 최소경계사각형의 후보집합으로 구해진다. 그 다음 레벨에서 A, B의 자식 노드 집합 { C, D, E, F, G, H }중에서 연결된 최소경계사각형을 찾는데 실패한다. 따라서, 두 객체 A, B는 연결되어 있지 않다는 질의 처리 결과를 얻을 수 있다. 이 경우 노드 참조 횟수는 뿌리 노드만을 이용하므로 단 한 번뿐이다.

만일, 선분 14와 선분 16의 연결성을 묻는 질의를 가정하면, 함수 Filtering에 의해 레벨 1에서는 후보 집합이 { A, B }가 되고, 그 다음 레벨에서 { C, D, E, F, G, H }중에서 { H, G }만이 후보 집합으로 계산된다. 따라서, Refinement 과정에서는 이 두 노드에 속한 객체들만에 대하여 연결성을 조사하면 된다. 이 경우 노드의 참조 횟수는 뿌리 노드, A, B노드, H, G노드의 참조를 위해 총 5번 일어난다.

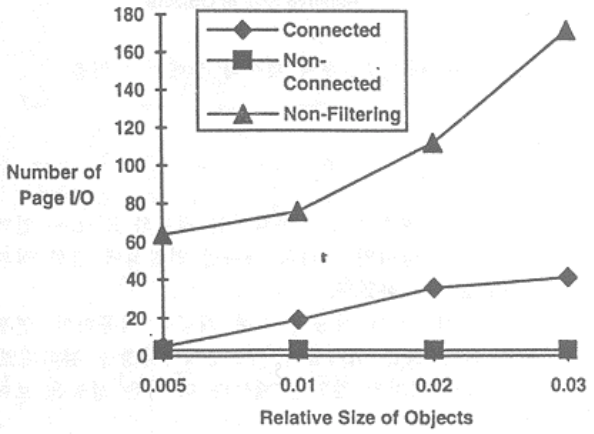
만약 여과 과정을 거치지 않는다면, Rfinement 단계ㅇ의 대상, 즉 실제로 연결되어 있는 지를 조사해야 할 객체는 잎 노드에 있는 모든 객체가 될 것이다. 이는 위의 두 예에서보다 더 많은 노드를 참조하게 되는 것이다.

4. 성능 측정

본 장에서는 앞에서 제시된 방법을 사용하지 않고 기존의 그레프 탐색 알고리듬을 그대로 사용하였을 경우와, 정제 단계를 거쳤을 경우의 디스크 입출력횟수를 비교한다. 비교의 기준으로 사용한 것은 디스크의 입출력 횟수이다. 물론 CPU처리시간도 경우에 따라서는 질의처리성능에 어느정도 영향을 미칠 수있으나, 주로 성능을 결정하는 요소는 디스크의 입출력 시간이다.

본 논문에서는 성능의 비교를 위하여 모의 실험을 하였다. 이 실험을 위하여, 가상 객체를 2000개의 생성하였는데. 이 객체의 분포는 임의의 좌표로 결정했으나, 실제 현실에서 사용되는 경우와 유사하게 하기위하여 어느 정도의 집중성을 가지도록 하였다. 즉 일반적인 경우 자료는 특정지역에 집중적으로 분산되기 때문에 고른 분산 보다는 집중성을 갖는 자료를 생성하였다. 또한 객체의 생성시, 전체 크기에 대한 객체의 상대적크기를 여러가지로 변화시켜보았다.

이렇게 만들어진 객체를 이용하여 R*-tree를 생성하였다. 이 경우 R*-tree의 하나의 노드가 1K byte디스크페이지에 저장되기 위하여 분기율을 50으로 정하였더니 높이가 3 이 되었다.



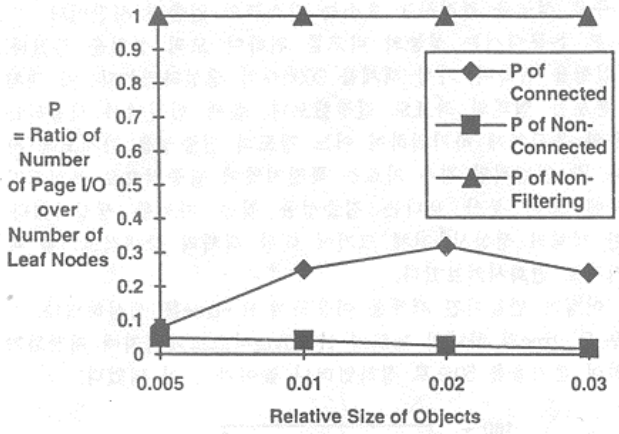
Connected : 연경된 쌍의 경우
Non-Connected : 연결된 쌍이 아닌 경우
Non-Filtering : 논문의 방법을 사용하지 않았을 경우

[그림 3] 연결성 질의처리를 위한 디스크입출력 횟수

그림 3은 임의의 200개 객체쌍을 선정하여 이쌍의 연결성을 위한 질의처리를 할 때 요구되는 디스키입출력의 횟수를 나타낸 것이다. 이경우 우리는, 연결된 쌍의 경우나 연결 안된 쌍의 경우 모두 논문에서 제안된 방법을 사용 않하였을 경우에 비해 매

우 큰 효과를 얻는 다는 것을 발견할 수 있다. 특히 연결되지 않았을 경우의 쌍은 매우 효과적이다. 그 이유는 단말 노드까지 가지 않고 중간 노드에서 이미 연결이 않되었다는 결론을 얻을 수 있기때문에 그만큼 디스크 입출력이 필요하지 않기 때문이다. 객체의 상대적인 크기가 증가함에 따라 입출력의 횟수도 함께 증가하는 것은 R*-tree의 노드수는 상대적 크기가 커짐에 따라 증가하기 때문이다. 즉 증가된 노드만큼 입출력이 많아지기 때문이다.

그림 4는 본 논문에서 제안된 방법에 대한 상대적인 효율을 나타낸 것이다. 즉, 본 논문의 방법을 사용하지 않았을 경우와 비교하여, 연결된 쌍의 경우와 연결되지 않았을 경우 쌍의 비율을 나타낸 것이다. 여기서도 우리는 많은 성능의 향상을 확인 할 수있다. 여기서 사용된 비율 P는 결국 전체 필요한 디스크입출력 횟수에 비하여 얼마만큼의 많은 양이 정제단계에서 제거되었는가를 나타내는 것이다.



[그림 4] 디스크입출력 횟수의 상대적인 비율

5. 결론

본 논문에서는 객체들의 기하학적인 정보와 R*-tree 인덱싱 기법을 이용하여 위상적인 질의의 하나인 연결성에 관한 질의를 처리하는 방법을 제안하였다.

공간 조인 기법에서 사용된 기본 개념을 적용하여, 연결성을 조사할 필요가 없는 객체들의 최소경계사각형을 걸러내었다. 이는 여과 과정을 거치지 않았을 때보다 더 나은 성능을 보이는 것을 알았다.

본 논문에서 쓰인 인덱싱 기법이외의 다른 인덱싱 기법을 고려할 수도 있다. 그러나, 그리드(grid) 화일 인덱싱 기법등은 객체들을 경계사각형으로 나타내지 않기 때문에 효율적인 여과 과정을 거칠 수 없기 때문에 사용하기가 힘들다.

앞으로의 연구과제는 위상적인 질의처리를 위한 위상적인 인덱싱 기법을 개발하거나, 지리 정보 시스템의 실제 응용분야에서 많이 필요한 최단경로나, upstream, downstream등의 질의처리를 위한 방법을 개발하는 것이다. 본 논문에서 제안한 방법의 성능을 향상시키기 위해서 질의 처리를 병렬화하는 Parallel R-tree등의 접근방법도[6] 고려해 볼 만하다 .

6. 참고 문헌

[1] R. Agrawal and H. V. Jagadish, "Algorithms for Searching Massive Graphs", IEEE TOKDE, Vol.6 No.2, 1994, pp.225-238

[2] Robert Sedgewick "Algorithms", Addison-Wesley, 1988

[3] Yannis E. loannidis, Raghu Ramakrishnan,"Efficient Transitive Closure Algorithms", Proc. of the 14th VLDB'88, pp.382-394

[4] T. Sellis, N. Roussopoulos, and C. Faloutsos, "The R*-tree : A Dynamic Index for Multidimensional Objects", Proc. VLDB 1987, pp. 507-518

[5] O. Guenther and A. Buchmann, "Research Issues in Spatial Databases", SIGMOD Record. Vol.19 No.4, 1990, pp.61-68

[6] I. Kamel and C. Faloutsos, "Parallel R-trees", Proc. SIGMOD'92, pp. 195-204 1992

[7] O. Gunther and J. Bilmes, "Tree-Based Access Methods for Spatial Databases: Implementation and Performance Evaluation", IEEE TOKDE Vol 3 No. 3, pp. 342-356 1991

[8] T. Brinkhoff, H. -P. Kriegel, and B. Seeger, "Efficient Proc. of Spatial Joins Using R-trees", Proc. SIGMOD'93, pp.237-246 1993

[9] N. B., H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An Efficient and Robust Access method for Points and Rectangles", Proc. SIGMOD'90, pp.322-331 1990