제22회 한국정보처리학회 추계학술발표대회 논문집 제11권 제2호 (2004. 11)

GIS와 VLSI Design을 위한 효율적인 공간 색인구조

방갑산
한성대학교 소프트웨어시스템공학과
e-mail: ksbang@hansung.ac.kr

Efficient Spatial Index Structure for GIS and VLSI Design

Kapsan Bang
Dept of Software System Engineering, Hansung University

요 약

공간 색인구조는 공간 데이터를 효율적으로 관리하기 위한 도구로써, GIS와 같은 공간 데이터베이스의 성능을 결정하는 중요한 요소라 하겠다. 대부분의 응용분야에서 공간 데이터베이스는 보조기억장치에 저장된 방대한 양의 공간데이터 처리를 요구하므로 디스크 접근의 수를 줄이는 것이 전체 데이터베이스의 성능을 향상시키는데 중요한 요소이다. 이 논문에서는 SMR-tree라는 공간색인구조의 여러 응용분야에서 활용 가능성을 기존의 색인구조들과의 비교를 통해 확인한다. SMR-tree는 R-tree 계열의 구조로써 기존의 R-tree계열의 구조들과 동일한 노드의 형태를 가지고 있으나, 여러 개의 data space를 사용하여 data object를 배분함으로써 R^*-tree의 말단노드 내에 존재하는 잉여공간을 제거하면서, R-tree의 단점인 색인노드들 사이에 중첩을 허용치 않는다. SMR-tree의 성능은 여러 종류의 테스트 데이터(VLSI layout data, Tiger/Line file data)를 사용하여 R-tree, R^+-tree, R^*-tree와 비교된다. SMR-tree는 높은 공간 활용도와 다른 색인구조에 비해 빠른 질의 성능을 보임으로써 GIS와 같은 공간 데이터베이스를 위한 효율적인 색인구조로 사용이 될 것으로 기대된다

1. 서론

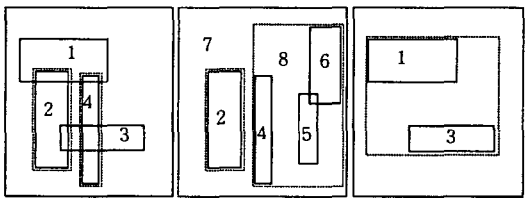
현대의 데이터베이스 시스템에 있어 geometric data(e.g., 점, 선, 다각형, 곡선, ...etc.)의 역할은 대단히 중요한 자리를 차지하고 있으며, 그 응용분야를 점차로 늘려가고 있다. 공간 데이터는 많은 non-standard 응용분야(e.g., GIS, CAD, VLSI 설계, 컴퓨터비전, 로보틱스, 이미지 데이터베이스,... etc.)에서 사용되고 있다[5, 8, 10]. 또한 공간 색인구조는 일반적인 1차원 데이터베이스에서 여러 개의 속성에 대한 다차원 질의를 할 때 k-속성을 k-차원상의 한 점으로 표현함으로서 빠른 접근을 제공하기 위해 사용이 될 수도 있다. 방대한 양의 공간 데이터를 처리하기 위한 효율적인 공간색인구조는 데이터베이스관리시스템(DBMS)의 중요한 요소 가운데 하나이다.

공간 색인구조의 연구가 처음으로 시작된 것은 70년대 후반이지만 본격적인 연구는 지난 약 15여년 동안으로써 현재 이 분야에 대한 연구가 활발하게 진행이 되고 있다. 그러나, 1차원 색인구조분야와는 달리 공간 색인구조 분야에서는 어느 구조도 모든 응용분야에서 안정된 성능을 보여주고 있지 못하고 있다. 그 이유는 응용분야에 따라 데이터의 분포와 질의의 형태가 다르며 공간 색인구조에서 공간분할에 고려되어야 할 요소가 많기 때문이다. 공간 데이터를 활용하는 시스템을 구축하는데 있어서 응용분야에 따라 데이터의 성격과 데이터에 대한 질의의 형태가 특정한 어느 색인구조를 더 선호하도록 할 수 있기 때문이다.

2. SMR-tree

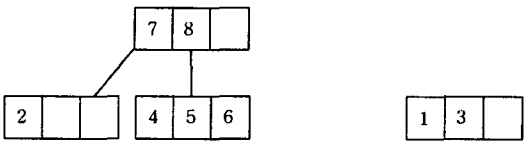
2.1 SMR-tree의 구조

SMR-tree는 다른 R-tree계열의 구조들과 동일한 노드구조를 가진다. 즉, 색인노드와 말단노드 두 종류의 노드타입을 갖고 있다. 말단노드의 각 엔트리는 데이터베이스내의 tuple을 가리키는 일종의 포인터인 tuple-identifier와 공간 데이터 object를 둘러싸는 n-차원의 MBR(Minimum Bounding Rectangle)의 좌표 값으로 구성이 되어 있다. 색인노드는 R-tree의 루트노드로부터 말단노드를 제외한 모든 노드를 말한다. 색인노드의 각 엔트리는 하위 레벨의 노드를 가리키는 자식포인터와 하위 레벨 노드 내에 있는 모든 사각형들을 둘러싸는 n-차원의 MBR의 좌표 값으로 구성이 되어 있다.



(a) (b) (c)

그림1 (a)overflowing 노드 (b) 1번째 자료공간에서의 데이터 object의 구성 (c) 2번째 자료공간에서의 데이터 object의 구성(실선은 object 사각형, 점선은 색인 사각형을 나타낸다)



(a) (b)

그림2 (a) 1번째 자료공간(그림1(b))에 대한 SMR-tree 구조 (b) 2번째 자료공간(그림1(c))에 대한 SMR-tree 구조

예를 들어, 그림1(b)의 색인사각형 7과 8은 그림2(a)의 색인노드안의 엔트리 7과 8로써 각각 나타내진다. 각 색인사각형들은 하위 레벨에 있는 모든 사각형들을 완전히 포함한다. 이러한 특성이 R^+-tree와 가장 다른 점이라 하겠다. 그러나 색인사각형들은 R^+-tree에서와 같이 단절되어 있다. SMR-tree는 R-tree의 장점인 말단노드 잉여공간의 제거와 R^+-tree의 장점인 단절된 색인사각형을 유지한다. SMR-tree는 기존의 구조에서 하나의 자료공간을 사용해 데이터 object들을 구성하는 것과 달리 여러 개의 가상적인 자료공간상에서 데이터 object들을 배분하여 구성함으로써 위에서 언급된 특성을 유지한다. 각 자료 공간 내에 데이터 object들은 그림2에서 볼 수 있듯이 하나의 독립된 트리 내에서 조직이 된다.

2.2 SMR-tree의 삽입 알고리즘

SMR-tree는 새로운 데이터 object를 삽입하기 위해서 적절한 하나의 트리를 선택한다. 만일, 삽입하는 시점에 SMR-tree의 특성(단절된 색인사각형, 하위 레벨 사각형들의 완전한 포함)을 지키면서 새로운 데이터 object를 받아들일 수 있는 트리가 존재하지 않는다면 SMR-tree는 새로운 트리를 생성하고 그 안에 데이터 object를 삽입한다. 적절한 삽입 경로를 찾기 위해 삽입 알고리즘은 루트노드에서부터 시작해 노드 안에 있는 엔트리들 가운데 SMR-tree의 특성을 만족시키는 엔트리가 있는지 검색을 한다. 새로운 데이터 object를 받아들일 수 있는 조건을 가진 엔트리 가운데, 새로운 데이터 object를 받아들인 후 색인 사각형의 확장을 최소로 하는 엔트리가 선택이 된다. 선택된 엔트리를 따라서 자식노드로 내려가면서, 말단노드에 이를 때까지 조건검색을 반복한다. 조건검색 가운데 색인 사각형의 면적을 가장 먼저 고려를 하는데 그 이유는 SMR-tree가 색인사각형의 중첩을 허용치 않기 때문에 하나의 자료공간에 들어 갈수 있는 색인사각형의 숫자는 색인 사각형의 크기가 작은 경우에 더 많기 때문이다. 만일 색인 사각형의 면적확장 수치가 동일한 것이 하나이상인 경우, 색인 사각형내의 하위 사각형의 수가 적은 쪽을 선택한다. 이는 노드 사이에 엔트리의 숫자에 균형을 주어 노드가 overflow할 확률을 줄이기 위해서이다.

2.3 SMR-tree의 노드 분할 알고리즘

SMR-tree의 노드분할에서 고려하는 인자와 그들

의 적용 우선 순위는 다음과 같다: (1)노드사이의 엔트리개수의 균형, (2)분할선 위에 중첩하는 데이터 object수의 최소화, (3)분할 이후 두 색인 사각형들의 면적의 최소화.

각 차원에 대해, overflowing 노드내의 사각형들의 시작과 끝 좌표값을 정렬한다. 하나의 차원에 대해서 정렬된 좌표 값의 리스트를 P라고 하자. P= {P1, P2, P3, ...P2(C+1)}, 이때 C는 노드의 용량을 나타낸다. Overflowing 노드내의 사각형들의 시작과 끝 좌표 값의 개수는 총 2(C+1)이다. SMR-tree의 분할 알고리즘은 각Pi 에 대해 (Li, Gi)를 계산한다. Li 는 사각형의 해당 차원상의 시작과 끝의 두 좌표 값이 모두 Pi 보다 작거나 같은 사각형의 숫자이고, Gi 는 사각형의 시작과 끝의 두 좌표 값이 모두 Pi 보다 크거나 같은 사각형의 숫자이다. 각 (Li, Gi)에 대해 둘 중에 적은 수를 택하고 그 list의 이름을 M이라 한다, M= {M1, M2, M3, ...M2(C+1)}. 분할 알고리즘은 list M중에서 가장 큰 값의 Mi를 택한다. 이때 Pi 가 현재 고려하고 있는 차원에 대한 분할 위치가 된다. 모든 차원에 대해 같은 방식으로 분할 위치를 결정한다. 분할 알고리즘은 각 차원의 Mi 값들 중에서 가장 큰 값을 가진 차원을 분할할 차원으로 정하고 그 때의 Pi 가 분할할 위치가 된다. 만일, 각 차원의 Mi 값들 중에 최대치가 하나이상 존재한다면, 각 차원의 Pi 위치에서 중첩하는 사각형의 수가 가장 적은 차원이 선택이 되고, 또 다시 동일조건의 차원이 하나이상 존재한다면, 분할선에 의해 분할한 뒤에 두 색인 사각형들의 면적 합이 최소가 되는 차원을 택한다. 분할 알고리즘은 선택된 차원의 분할선을 사용해서 사각형들을 두개의 하위영역으로 나눈다. 만일 분할선과 중첩하는 사각형이 색인 사각형이라면 그것은 선택된 차원의 분할선에 대해 말단 레벨 사각형에 이를 때까지 연속적으로 하위 분할된다. 만일 분할선과 중첩하는 사각형이 object 사각형이라면 이것은 현재의 자료공간에서 제거되어 다른 적절한 자료공간에 재 삽입된다. 분할선과 중첩하지 않는 모든 사각형들은 그들의 위치에 따라 두개의 하위 영역중의 하나에 배치가 된다.

3. 성능 비교

SMR-tree의 성능은 R-tree, R^+-tree, R^*-tree와 2종류의 테스트 데이터를 사용해서 비교가 되었다. 모든 구조들은 주어진 알고리즘에 따라 구현되었다. R-tree의 분할은 Quadratic방식이 적용되었고, R^+-tree는 [7]에서 주어진 알고리즘이 사용이 되었고 [4]에서 언급된 바와 같이 색인 사각형에 대해서 MBR방식 대신에 k-d-b tree방식을 적용하는 구현이 적용되었다.

2개의 테스트 데이터 VLSI data set, Tiger/Line™ data set, 각각 V, T라 한다. 2개의 테스트 데이터은 4가지의 특성(object 의 수, object 밀도, object의 크기비율, 자료공간)로 표시된다. Object의 밀도는 아래와 같이 정의된다:

$$D = \frac{\textit{data object}\text{의 면적 합}}{\textit{dataspace}\text{의 면적}} \times 100$$

type	갯수	밀도	크기비율	data space
V	4085	0.34	0.00017 - 0.45249%	1320 x 1768
T	41058	0.25	0.00000 - 0.34798%	533552 x 349200

표1 테스트 데이터의 특성

테스트 데이터 V 는 VLSI 설계시스템(Magic)에 의해 만들어진 것이다. 테스트 데이터 T는 Tiger 데이터베이스 시스템의 Tiger/file에서 얻어진 것이다. Tiger 데이터베이스 시스템은 미국 Census Bureau map [9]상의 point, line, area를 나타내는 정보를 제공한다.

검색은 전체 자료공간의 0%(point query) ~ 12%까지를 임의의 위치로 500회씩 수행을 하고, 평균 반응시간을 구한 것이다. 그림3 ~ 그림4은 각 테스트 데이터에 대한 4개의 공간 색인구조의 영역질의의 크기변화에 대한 반응시간을 보여준다. SMR-tree는 테스트 데이터의 대부분의 검색범위에 대해 안정된 성능을 보이고 있다. R^*-tree는 삽입시간이 오래 걸리는 반면에 그 성능은 본래의 R-tree보다도 별로 향상된 것이 없다. SMR-tree는 여러개의 트리로 구성이 되어 있으나 대부분의 데이터는 첫 번째의 트리에서 구성이 된다. 표2 은 SMR-tree의 데이터 object 분포 비율을 보여준다.

data type	VLSI	Tiger/Line
T1	83.63%	89.16%
T2	8.52%	10.12%
T3	7.80%	0.70%
T4	0%	0%
T5	0%	0%

표2 SMR-tree의 데이터object 분포 비율(Ti는 SMR-tree구성하는 트리)

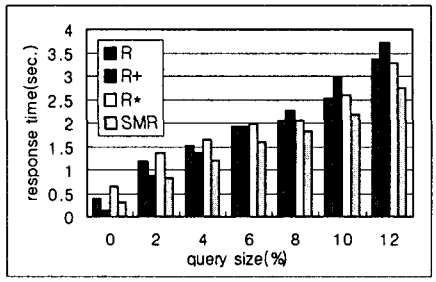


그림3 VLSI 데이터에 대한 질의성능

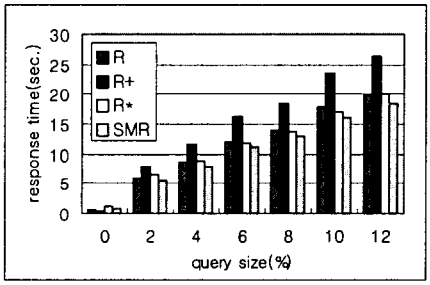


그림4 Tiger/Line 데이터에 대한 질의성능

4. 결론

이 논문에서 SMR-tree가 제시되었고 다른 R-tree, R^+-tree, R^*-tree들과 질의 성능이 실제의 데이터를 사용해 비교되었다. SMR-tree는 R-tree의 단점인 색인 사각형들의 중첩을 배제하고 R^+-tree의 말단 노드 레벨의 잉여공간을 제거함으로써 다양한 질의 크기에 대해 비교적 안정된 성능을 보였다. SMR-tree는 데이터의 밀도가 높은 경우 보다 안정된 성능을 보이는 특징을 가지고 있다. SMR-tree는 GIS와 같은 공간 데이터베이스와 VLSI 설계 도구등에 효율적인 공간 색인구조로써 사용될 수 있을 것이다.

참고문헌

[1]Beckmann, N and Kriegel, H. P., The R*-tree: An Efficient and Robust Access Method for Points and Rectangles, ACM SIGMOD, pp.322-331, 1990.

[2]Gunther, O., The Design of the Cell tree: An Object Oriented Index Structure for Geometric Databases, IEEE 5th International Conference on Data Engineering, pp. 598-605, 1989.

[3]Guttman, A., R-trees: A Dynamic Index Structure for Spatial Searching, Proc. of the ACM SIGMOD, pp. 47-57, 1984.

[4]Hoel, E. G., and Samet, H., A Qualitative Comparison tudy of Data Structures for Large Segment Databases, ACM SIGMOD, pp. 205-214, 1992.

[5]Lomet, D.B., A Review of Recent Work on Multiple attribute Access Methods, ACM SIGMOD Record, Vol. 21, No. 3, pp. 56-63, 1992.

[6]Osawa, Y. and Sakauchi, M., A New Tree Type Data Structure with Homogeneous Node Suitable for a Very Large Spatial Databases, Proc. of the IEEE6th International Conference on Data Engineering, pp. 296-303, 1990.

[7]Sellis, T., Roussopoulos, T. and Faloutsos, C., R^+-tree: A Dynamic Index for Multi-dimensional objects, Proc. of the 13th VLBD Conference, pp. 507-518, 1987.

[8]Samet, H., The Design and Analysis of Spatial Data Structures, Addison-Wesley, Reading, MA, 1990.

[9]Bureau of the Census, Tiger/Line File, 1992 Technical Documentation, Bureau of the Census, Washington, DC, 1993.

[10]Gunther, O., The Design of the Cell tree: An Object Oriented Index Structure for Geometric Databases, IEEE 5th International Conference on Data Engineering, pp. 598-605, 1989.