제36회 한국정보처리학회 추계학술발표대회 논문집 제18권 제2호 (2011. 11)

병렬처리 공간색인구조

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

A Study on Parallel Spatial Index Structure

Kapsan Bang
Dept of Information System Engineering, Hansung University

요 약

본 논문에서는 PAR-트리라는 공간색인구조를 제안한다. PAR-트리는 object distribution heuristic (객체분할법)으로 absolute crowd index(절대 복잡도)를 사용하여 공간 데이터 객체를 여러 개의 데이터 공간에 균일하게 배치함으로써 질의처리 속도를 향상시킨다. PAR-트리는 MXR-트리[8]에 비해 높은 공간활용도와 빠른 질의 반응시간을 보임으로써 다차원 데이터베이스를 위한 효율적인 색인구조로 사용이 될 것으로 기대된다.

1. 서론

병렬처리 공간 색인구조인 PAR(Parallel Absolute_crowd_index R)-트리의 설계가 소개된다. 병렬처리 공간색인구조에서 주어진 검색 영역과 중첩하는 엔트리가 서로 다른 디스크에 있는 경우 질의 처리동안에 동시에 엑세스된다. 효율적인 병렬처리 공간색인구조는 질의처리에 있어 전체 디스크 엑세스의 수가 적어야함은 물론 엑세스되는 노드가 디스크 상에 균일하게 분포되어 있어야 한다.

질의처리동안에 전체 디스크 엑세스의 수를 줄이기 위해서 R-트리[7], R^+-트리[11], R^*-트리[5]의 단점을 향상시켜야한다. PAR-트리는 공간 객체를 여러 개의 데이터 공간에 분배함으로써 색인 사각형들 사이의 중첩을 완전히 제거하고 말단 노드에 존재하는 중복된 엔트리들을 제거한다. 이를 위해, PAR-트리는 native space indexing (자연공간색)과 disjoint space decomposition(분리공간분할)방식을 사용한다. Disjoint space decomposition방식은 색인 사각형사이의 중첩을 R^+-트리와 같은 방식의 공간분할로 제거하여 불필요한 검색경로를 제거한다. 그러나 R^+-트리는 색인 사각형의 중첩을 제거하기 위해 색인 사각형에 완전히 포함되지 않는 공간 객체는 그 객체와 중첩하는 여러 말단 노드에 중복하여 저장하는 방식을 택한다. 이러한 중복된 객체 엔트리는 결국 노드의 수를 증가시켜 검색영역이 커지는 경우 성능에 치명적인 영향을 미치게 된다. 검색 영역이 아주 작은 경우 예를 들어 point(점 좌표) 검색(주어진 하나의 위치좌표와 중첩되는 객체의 검색)의 경우에 R^+-트리는 우수한 성능을 보인다. 또한 R^+-트리는 삭제 연산 시에 중복된 모든 객체 엔트리를 삭제하기 위해 다중의 삭제 과정을 거쳐야 한다. 대부분의 공간색인구조는 하나의 저장공간에(하드디스크) 구축된다. 이러한 방식의 공간색인구조는 참고문헌 [1, 2, 3, 4, 6, 7, 9, 10]에 기술되어있다. PAR-트리의 상위 레벨의 색인 사각형은 모든 하위 레벨 사각형들을 완전히 포함한다. 이러한 특성을 유지하기 위해 PAR-트리는 여러 개의 데이터 공간을 사용한다. 새로운 공간객체를 삽입하는 경우, 존재하는 데이터 공간 중 새로운 객체를 완전히 포함할 수 있는 공간이 있다면 그 공간에 삽입이 되고, 만일 어떠한 데이터 공간의 색인 사각형도 객체를 완전히 포함할 수 없다면 새로운 데이터 공간이 생성되고 그 곳에 삽입이 된다. 객체의 삽입과정에서 노드의 분할(node split)이 발생하는 경우 해당 공간에 저장될 수 없는 객체는 그 공간으로부터 제거되어 재 삽입된다.

병렬처리 공간색인구조의 성능을 향상시키기 위해서 공간 객체를 분배하는 heuristic(법칙)이 필요하다. 병렬처리 공간색인구조의 질의처리성능은 여러 개의 디스크를 동시에 엑세스할 때 가장 많은 디스크 엑세스를 갖는 디스크에 의해 결정된다. 효율적인 object distribution heuristic(객체분배법)은 데이터 구조의 특성을 유지하면서 공간 객체를 여러 개의 데이터 공간에 균일하게 분배함으로써 질의 반응시간을 줄여야한다. PAR-트리의 노드 분리 알고리즘(split algorithm)과 object distribution heuristic(객체분배법)은 또한 높은 공간 활용도를 유지한다. 병렬처리 공간색인구조들은 여러 개의 프로세서와 여러 개의 디스크가 사용되는 하드웨어 구조를 가지고 있다.

2. PAR-트리 구조

PAR-트리는 multiple-layer(계층)와 multiple-tree(트리)의 구조를 가지고 있다. 각 계층은 여러 개의 트리들로 구성되어 있고, 트리의 개수는 사용되는 디스크의 수와

같다. 각 데이터 공간은 하나의 디스크 상에 위치하는 독립된 하나의 트리와 연관되어있다. 하나의 디스크는 하나 이상의 트리를 가질 수 있다. PAR-트리에서는 각 트리들이 독립적인 구조를 가지고 있기 때문에 질의 연산 동안에 프로세스간의 정보교환이 불필요하다. 따라서 모든 디스크의 완전한 병렬처리가 가능하다. 그림2는 2개의 계층과 4개의 트리를 가진 PAR-트리의 layout(형태)을 보여준다.

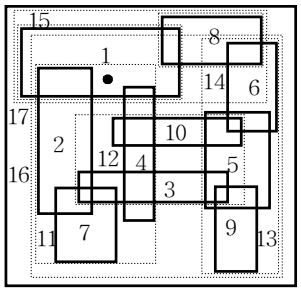

그림1. 점선은 색인 사각형, 실선은 객체 사각형표현

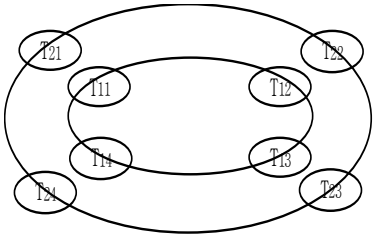

그림2. 2-계층 4-트리로 구성된 PAR-트리의 구조

각 트리들은 트리 ID인 T_{ij}(i는 계층의 번호이고 j는 트리의 번호를 나타낸다. 트리번호는 디스크의 번호이기도 하다)에 의해 식별될 수 있다. 디스크 j는 트리 ID T_{ij}(i= 1,2,3,...,L, L은 계층의 수)를 가진 트리들을 포함한다. 예를 들어, 그림2에서, 디스크 1은 트리 T_{11}과 T_{21}을 포함한다. 따라서 PAR-트리의 질의 반응시간은 디스크 엑세스의 최대 수에 비례한다. 만일 P_{ij}가 트리 T_{ij}안에 있는 노드들을 처리하는데 걸리는 시간이라면 PAR-트리의 질의 반응시간(R)은:

$$R = \max_{j=1\ldots D}\left(\sum_{i=1}^{L} P_{ij}\right)$$

D는 디스크의 수를, L은 계층의 수를 나타낸다.

그림3는 2차원 공간 객체들에 대한 PAR-트리의 객체 분배의 예를 보여준다. PAR-트리는 R-트리계열의 노드 구조를 가진다. 즉 모든 객체는 말단 노드에 저장이 되고, 색인 노드는 색인 사각형들을 나타내는 엔트리들로 구성이 된다. 그림3에서 PAR-트리는 객체 사각형 1에서 10까지를 3개의 데이터 공간에 간단한 heuristic인 minimum entry number(엔트리의 수가 최소인 트리의 선택)를 사용하여 분배한 것이다. 그림3(a), 3(b), 3(c)에서 모든 데이터 공간은 중첩되지 않은 색인 사각형들을 가지고 있고 모든 사각형들은 상위 레벨 사각형들에 의해 완전히 포함된다. 그림3(d), 3(e), 3(f)는 3개의 데이터 공간에 상응하는 트리를 보여준다. 그림3(d), 3(e), 3(f)에서 트리 T_{ij}에 있는 모든 노드들은 디스크 j에 저장이 된다. PAR-트리의 말단 레벨에는 중복된 엔트리가 존재하지 않는다. 그림3에서 PAR-트리는 1-계층 3-트리의 구조를 갖는다.

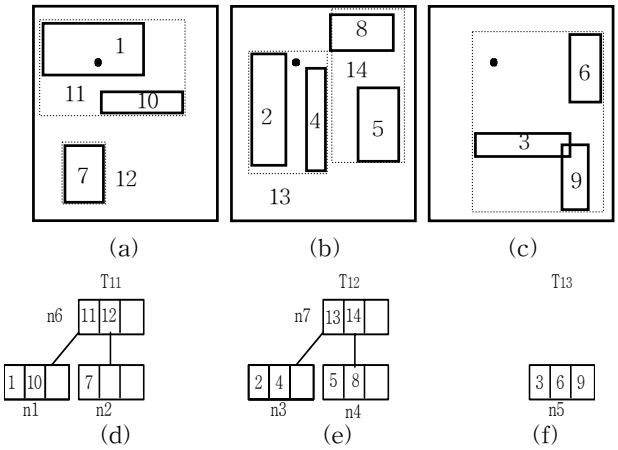

그림3. 점선은 색인 사각형, 실선은 객체 사각형을 각각 나타낸다: (a)첫 번째 데이터공간 (b)두 번째 데이터공간 (c)세 번째 데이터 공간; (d),(e)(f)는(a),(b),(c)에 대한 PAR-트리 구조

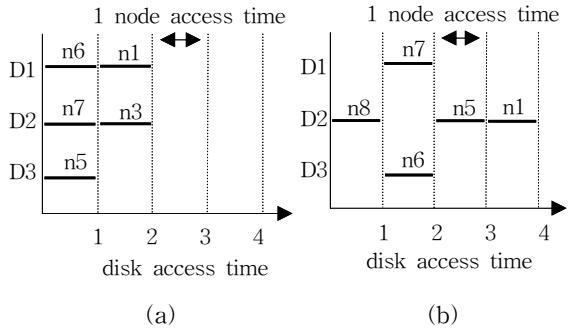

그림4. MXR-트리와 PAR-트리의 병렬처리 디스크 엑세스 시간

그림4(a)와 (b)는 그림1과 그림3의 예에서 주어진 point(점 좌표) 질의를 처리하기 위한 MXR-트리와 PAR-트리의 디스크 엑세스 시간을 각각 보여준다. 하나의 노드를 엑세스하는데 걸리는 디스크 엑세스 시간이 상수이고 한 노드의 크기가 1 페이지라 가정한다. 그림3에서 오직 공간 객체 1만이 주어진 검색 점과 중첩한다. 그림4(b)에서 주어진 점 질의를 처리하기 위한 PAR-트리의 디스크 엑세스의 수는 5이고, 이 5개의 노드를 병렬로 엑세스하는 시간은 단위시간 2가 걸린다. PAR-트리에서 각 프로세스는 하나의 디스크와 연계되어, 오직 주어진 검색 영역과 중첩하는 노드들만 엑세스하고 프로세스간의 정보교환이 없다. 그러나 MXR-트리는 주어진 point(점 좌표) 질의처리에 단위시간 4가 걸리는 것을 볼 수 있다.

3. Absolute Crowd Index(절대복잡도)

공간 객체는 임의의 크기와 모양을 갖는다. 또한 이러한 객체들은 서로 중첩될 수 있고 공간상에 임의의 위치에 분포한다. 이는 대부분의 응용분야에서 데이터 공간의 어느 한 부분이 다른 부분에 비해 공간 객체의 밀도가 아주 높거나 낮을 수 있음을 의미한다. 질의처리에서는 D개의 프로세스가 만들어져 각 프로세스는 할당된 디스크 안에 저장된 트리들을 주어진 검색 영역에 대해 검색한다. 따라서 좋은 성능을 갖기 위해서는 PML-트리의 같은 계층에 있는 트리들은 거의 엇비슷한 수의 노드를 각각의 검색경로에 가져야한다. 그림3의 예에서 사용했던 minimum entry number(최소 개체 수)에 의한 공간 객체의 분배는 객체의 공간상의 위치를 고려한 것이 아니기 때문에 각 트리의 검색 영역 안의 노드들의 숫자가 트리들 사이에 불균형을 이룰 수 있다. 삽입 경로를 따라 노드의 분리 가능성이 큰 트리는 보다 많은 노드(색인 노드와 말단 노드)를 만들려는 경향을 가지고 있다. 결과적으로, 높은 노드 분리의 가능성을 가졌던 삽입 경로에 해당하는 영역에 대한 검색 연산에서는 보다 많은 노드가 엑세스되어야한다. Absolute crowd index(AC)에 의한 객체 분배는 새로운 엔트리인 "NEW"를 삽입하기 위한 삽입경로를 따라서 최소한의 노드 분리 가능성을 가진 하나의 트리를 찾는 것이다. AC distribution heuristic은 노드 분리의 가능성을 결정하기 위해 삽입경로를 따라서 노드의 crowdness(복잡도)를 측정을 한다.

$$\sum_{j=1}^{h} (W \times j \times N_j \rightarrow EntryNum)$$

h는 트리의 오더, W는 가중치 상수, N_j는 삽입경로의 오더가 j인 노드, EntryNum은 노드 N_j안의 엔트리의 수이다.

가중치 상수인 W(>1)는 높은 오더의 노드에 보다 많은 가중치를 주기 위해서 사용이 된다. 오더가 높은 노드는 엔트리 NEW의 삽입에 의해 보다 쉽게 분리하려는 경향이 있다. 왜냐하면, 새로이 삽입되는 엔트리 NEW는 말단 노드에 삽입되고 말단 노드는 가장 높은 오더를 가지고 있기 때문이다. 가장 낮은 AC index값을 가진 트리가 엔트리 NEW의 삽입을 위해서 선택된다. 만일 말단 노드가 NEW를 받아들일 수 있다면 트리의 상태는 말단 노드 내의 엔트리의 수에 의해 A(accept)또는 S(accept and split)로 설정된다. 상태가 A 또는 S인 트리들에 대해서 알고리즘은 4가지의 범주(트리 오더, index 값, 트리 상태, 트리에 속한 엔트리의 수)를 사용을 해서 하나의 트리를 선택을 한다. 이들 4개의 criteria의 적용 우선순위는:

1. 트리 오더- 최소 오더를 가진 트리를 선택
2. index 값- 최소 AC index값을 가진 트리를 선택
3. 트리 상태-상태 A를 가진 트리를 상태 S를 가진 트리에 우선해서 선택
4. 트리에 속한 엔트리의 수-최소 숫자의 엔트리를 가진 트리를 선택

만일 현재의 계층에 어떤 트리도 새로운 엔트리인 NEW를 받아들일 수 없다면, 현재의 계층을 하위의 계층으로 설정을 하고 선택 프로세스를 반복한다. 만일 하위의 계층이 존재하지 않는다면 새로운 계층이 만들어진다.

AC distribution heuristic를 사용해서 새로운 엔트리를 삽입할 하나의 트리를 선택한 후, PAR-트리의 삽입 알고리즘은 선택된 트리를 root 노드에서 말단 노드까지 삽입경로를 따라 엑세스한다. 새로운 엔트리를 말단 노드에 삽입한 뒤에 알고리즘은 삽입경로에 있는 노드들을 나타내는 사각형들의 minimum bounding(최소 경계) 사각형들을 계산하여 업데이트한다. 만일 삽입경로에 있는 노드가 넘친다면(노드의 용량 이상의 엔트리가 삽입될 때) 노드 분리 알고리즘이 (용량+1) 개의 엔트리들을 2개의 노드(본래의 노드 N과 새로이 만들어진 분리 노드 SP)로 재분배한다. 만일 공간 객체(말단 레벨 엔트리)가 삽입 과정에서 노드 분리에 의해 제거가 된다면 이들 엔트리는 재 삽입된다.

3. 결 론

PAR-트리는 여러 개의 디스크를 사용하여 질의 연산에 따르는 디스크 I/O를 병렬 처리함으로써 질의 처리를 향상시켰다. 공간 객체를 디스크 상에 균일하게 분배하기 위해서 PAR-트리는 object distribution heuristic(객체분배법)을 제시하였다.

참고문헌

[1] Bang, K. S. and Lu, Huizhu, A Simulation on an Index Structure for the Spatial object, Proc. of the International Simulation Technology Conf.(SIMTEC'92),November, pp. 178-183, 1992.

[2] Bang, K. S. and Lu, Huizhu, An Application of the multi-R tree to the VLSI circuit layout design, Proc. 9th International Conf. on System Engineering, University of Nevada Las Vegas, pp.295-299, 1993.

[3] Bang, K. S. and Lu, Huizhu, SMR-tree: an efficient Index Structure for Spatial Databases, Proc. of the 1995 ACM Symposium on applied Computing, Nashville, February, pp. 46-50, 1995.

[4] Bang, K. S. and Lu, Huizhu, An Efficient Index Structure for Spatial Databases, Journal of Database Management, Vol. 7, No. 3, Summer, pp. 3-15, 1996.

[5]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.

[6] 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.

[7] Gűttman, A., R-trees: A Dynamic Index Structure

for Spatial Searching, Proc. of the ACM SIGMOD, pp. 47-57, 1984.

[8] Kamel, I. and Faloutsos, C., Parallel R-tree, ACM SIGMOD, pp. 195-204, 1992.

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

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

[11] 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.