합병 방법을 이용한 고정 격자 색인의 성능 개선

김동현°, 문정욱, 이기준
{dhkim, jwmoon}@quantos.cs.pusan.ac.kr, lik@pnu.edu

Fixed Grid File Packing using Merge

Tong-Hyon Kim°, Jung-Wook Moon and Ki-Joune Li
Dept. of Computer Science, Pusan National University

요 약

고정 격자 방식의 공간 색인 방법은 간단한 구조와 단순한 색인 과정, 구현의 용이성이라는 장점이 있으나 데이터의 분포에 영향을 많이 받아 밀집된 데이터를 처리하기에는 적합하지 못한 특성이 있다. 이에 본 논문에서는 고정 격자 색인 방법에 합병 정책을 적용하여 고정 격자 색인 방법의 성능을 향상시키는 방법을 제안한다. 본 논문의 방법에 따르면 공간 효율성이 매우 증가하고 다른 공간 색인 방법에 비해 색인 과정이 단순해 지며, 공간 색인의 성능이 다른 색인 방법에 비해 증가되는 장점을 가지고 있다.

1. 서론

대부분의 공간색인 방법들은 공간객체의 분포가 균일하지 않다는 가정 하에 만들어졌다. 실제로 이 가정은 대부분의 응용분야에 일반적으로 적용된다. 예를 들어 R-tree 계열의 공간색인 방법은 공간객체의 분포에 상관없이 좋은 성능을 보일 수 있도록 만들어졌다[1]. 그러나 공간 객체의 분포에 영향을 받지 않도록 하기 위해서 색인 구조가 복잡해지게 되었다. 반면에 격자 파일[2]이나 고정 격자 파일과 같은 공간 색인 방법은 비균등 분포의 공간 데이터에 대해서는 성능이 떨어지지만, 균등분포의 공간 데이터에 대해서는 좋은 성능을 보인다. 또한 이들 공간 색인 방법은 간단한 구조와 단순한 색인 과정, 구현의 용이성이라는 장점이 있다.

그러나 이와 같은 장점에도 불구하고, 대부분의 응용 분야에서 주어지는 실제 공간 객체의 분포는 비균등하기 때문에 이들 방법은 사용되기 곤란하다. 이에 본 논문에서는 고정 격자 색인 방법의 장점을 유지하면서 밀집된 데이터에 대해 디스크 저장 효율은 높이기 위해서 합병 정책을 고정 격자 색인 방법에 적용해 보았다.

2. 고정 격자 색인 방법의 장단점

고정 격자 색인 방법은 전체 데이터 영역을 일정 간격으로 나누어 하나의 격자를 하나의 디스크 블록에 저장하는 구조이다. 하나의 격자는 하나의 디스크 블록에 해당 되므로, 질의 영역과 겹치는 격자에 해당되는 디스크 블록을 읽으면 되므로 다른 공간 색인 방법과 같이 색인 과정의 디스크 접근이 불필요하다.

이렇듯 구조가 간단하여 색인 방법이 간단한 데도 불구하고 고정 격자 색인 방법이 널리 사용되지 못하는 이유는 처리할 수 있는 데이터의 한계성 때문이다. 고정 격자 색인 방법은 균등 분포의 데이터에는 좋은 성능을 보이는 반면, 비균등 분포의 데이터에 대해서는 데이터가 밀집되어 있을수록 급격히 성능이 떨어지게 된다. 하나의 격자가 하나의 디스크 블록에 해당되게 되므로, 하나의 격자에 포함된 데이터의 개수가 하나의 디스크 블록에 저장될 수 있는 최대의 데이터 개수보다 작아야 한다. 그런데 밀집된 데이터를 처리하게 되면 격자가 잘게 나누어지게 되고 이 때, 사장공간(Dead Space)이 증가하게 된다. 사장공간이 증가하게 되면 저장 공간의 효율성이 저하되어, 색인의 성능이 저하되게 된다.

몇 개의 격자를 짝지어 하나의 디스크 블록에 저장하면 저장 공간의 효율성이 증가되며 색인의 성능이 좋아질 것이다. 그래서 본 논문에서는 합병을 이용한 고정 격자 색인 방법의 성능 개선 방안을 제안한다.

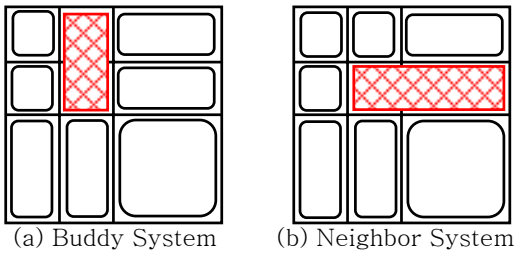
3. 합병(Merge) 방법

합병 방법은 크게 이웃 합병 방법과 Linear Clustering 방법으로 나눌 수 있다. 이웃 합병 방법은 기준이 되는 격자를 중심으로 주변의 격자를 병합하는 방법이다. 대표적인 이웃 합병 방법으로는 Tree구조를 이용하는 Buddy System이 있고[1], 본 연구에서 제안하는 4-Neighbor Merge, 8-Neighbor Merge방법이 있다. Linear Clustering 방법은 전체 격자들을 Linear Ordering한 뒤 그 순서대로 병합하는 방법이다. Linear Clustering 방법에는 Column-wise Scan, Column-wise snake Scan, Z-curve, Gray-code, Hilbert Order 방법이 있다[2].

3.1 이웃 합병 방법(Neighbor Merge)

이웃 합병 방법에는 기존에 소개된 방법으로는 그림 1에서 나타낸 Buddy System과 Neighbor System이 있다[1]. 이 방법은 Grid File에 사용되는데 트리 구조의 색인을 보조 장치로 사용하여 트리의 데이터 노드의 이웃하는 격자를 합병하는 방법이다. Buddy System은 같은 부모를 가지는 노드를 합병하고, Neighbor System은 이웃하는 부모의 노드까지 합병한

다.



(a) Buddy System (b) Neighbor System

그림 1. 이웃 합병 정책

본 논문에서는 보다 효율적인 합병 방법으로 4-Neighbor, 8-Neighbor 합병방법을 제안한다. 격자 구조에서 기준이 되는 격자를 중심으로 8개의 이웃 격자가 존재하는데 그림 2와 같이 그 이웃들을 우선순위를 정해 합병하는 방법이다. 실험 결과 8-Neighbor방법을 이용한 것이 성능이 조금 더 나았기 때문에 본 논문의 4장의 실험에서는 8-Neighbor 합병 방법을 이용하였다.

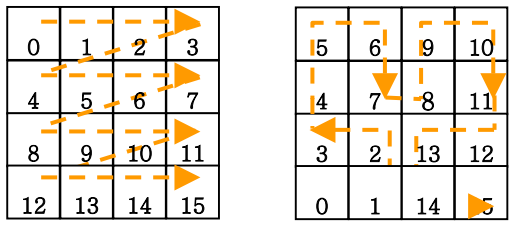
7	3	6
4	c	2
8	1	5

그림 2. 8-Neighbor 합병 방법

이웃 합병 정책을 이용하는 경우 중심이 되는 격자를 어떻게 이동시키는가 하는 것도 중요한 문제인데 이 문제는 3.2에서 다루기로 한다.

3.2 Linear Clustering을 이용한 합병 방법

Linear Clustering을 이용한 합병 방법은 전체 격자를 순서를 매긴 뒤, 순서대로 하나의 디스크 블록에서 수용할 수 있는 최대 데이터를 넘지 않도록 합병해 가는 방법이다. 전체 격자를 순서를 매기는 방법에는 Column-wise Scan 방법, Column-wise Snake Scan 방법, Z-Order 방법, Gray-code 방법, Hilbert Order 방법이 있다[2]. 본 연구에서는 Linear Clustering 방법으로는 성능이 가장 좋다고 알려진 Hilbert Order 방법을 이용하였다.



(a) Column-wise Scan (b) Hilbert Ordering

그림 3. Linear Ordering

앞 절에서 말한 이웃 합병 방법의 기준이 되는 격자의 이동 방법은 가장 간단한 Linear Clustering 방법인 Column-wise Scan 방법과 가장 좋은 Linear Clustering 방법인 Hilbert Order 방법을 이용하였다.

4. 실험

본 연구에서는 서울 데이터, 몽고메리카운티 데이터, 합성 데이터에 대해서 각 데이터를 고정 격자 방법으로 나누어 이웃 합병 방법, Linear Clustering 합병 방법으로 합병한 뒤 메모리 사용량과 질의 처리 성능을 평가해 보았다.

4.1 합병을 이용한 고정 격자 색인 방법

본 연구에서 제안하는 합병을 이용한 고정 격자 색인 방법은 다음과 같은 순서대로 진행된다. 먼저 전체 데이터 영역을 고정 격자 색인 방법으로 나눈다. 그리고 합병 정책에 따라 각 격자들을 합병하되, 합병된 격자들의 전체 데이터의 개수가 디스크 블록에서 수용할 수 있는 최대 데이터 개수를 넘지 않도록 한다. 마지막으로 각 합병된 격자들을 하나의 디스크 블록에 저장한다. 그러면 영역질의가 주어졌을 때, 각 격자들이 어느 디스크 블록에 할당되어 있는지를 알아와서 질의를 처리한다.

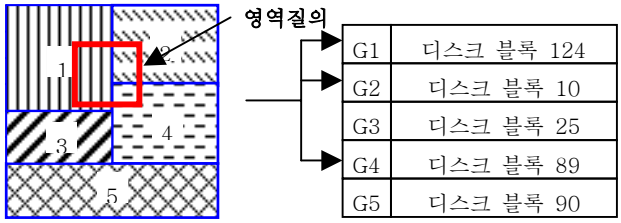
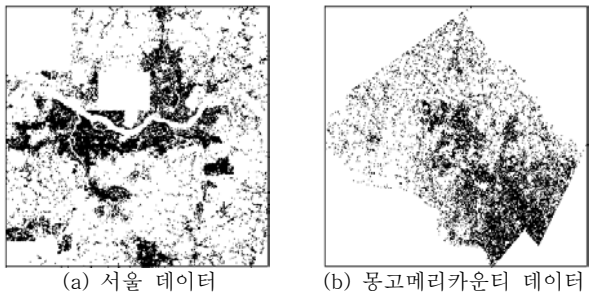


그림 4. 영역질의 처리

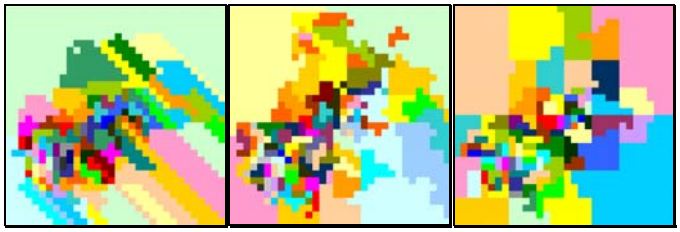
본 연구에서는 실험 데이터로 서울 데이터와 몽고메리카운티 데이터, 그리고 균등 분포의 합성데이터를 이용하였다. 각각 100,914개, 27,282개와 100,000개의 점 데이터로 이루어져 있는 데이터이며 실제 데이터는 그림 5와 같다.



(a) 서울 데이터 (b) 몽고메리카운티 데이터

그림 5. 데이터

실제 데이터를 고정 격자 방식으로 나누어 각 합병 방법을 이용하여 합병한 결과는 아래 그림 6과 같다. 아래 그림은 몽고메리카운티 데이터를 고정 격자 방식으로 변환한 뒤에, 8-Neighbor 이웃 합병 방법을 이용하여 합병하고, 각 격자가 묶여진 모양을 같은 색으로 나타낸 것이다. 같은 색으로 묶여



(a) NM_C (b) NM_H (c) LC_H

그림 6. 합병 결과

진 영역은 같은 디스크 블록에 저장되게 된다.

그림 6(a)는 중심이 되는 격자를 Column-wise Scan 방법으로 이동시키면서 8-Neighbor를 합병한 것이고(NM_C), (b)는 중심이 되는 격자를 Hilbert Order 방법으로 이동시키면서 8-Neighbor를 합병한 것이다(NM_H). 그리고 (c)는 Linear Clustering의 Hilbert Order대로 전체격자의 순서를 정해 순서대로 합병한 것이다(LC_H).

디스크 공간 활용도는 (a)가 85.11%, (b)가 90.91%, (c)가 89.89%로 기준이 되는 격자가 Hilbert Ordering된 이웃합병

방법이 디스크 공간 활용도가 가장 높은 것으로 보이나 대체적으로 비슷함을 알 수 있었다.

4.2 메모리 사용량 비교

표 1에서 R*-tree와 합병 정책을 적용한 고정 격자 색인 방법의 메모리 사용량을 비교해 보았다. R*-tree의 메모리 사용량은 트리 구조의 색인의 중간노드와 데이터노드의 크기를 합한 것이고, 고정 격자 색인 방법의 메모리 사용량은 각 격자 내부의 데이터가 저장되어 있는 위치를 알려주는 2차원 배열이다.

[표 1] 메모리 사용량

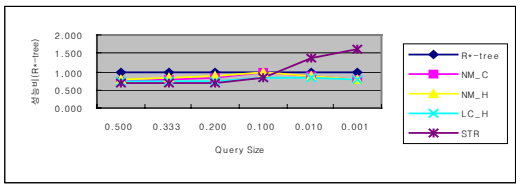
	R*-tree			N	N	LC_H
	중간노드	데이터노드	전체	격자	격자	격자
서울	28,672	3,129,344	3,158,016	16,384	16,384	16,384
몽고메리	12,288	1,015,808	1,028,096	1,024	1,024	1,024
합성	24,576	2,830,336	2,854,912	1,024	1,024	1,024

위 표에서 알 수 있듯이 R*-tree는 색인에 서울 데이터는 약 3M, 데이터의 객체 수가 가장 작은 몽고메리카운티 데이터에 대해서도 약 1M의 메모리가 필요하지만 고정 격자 색인 방법을 이용하는 NM_C, NM_H, LC_H 같은 방법은 가장 많은 서울데이터와 같은 경우에도 16K 정도만 사용하는 것을 알 수 있다.

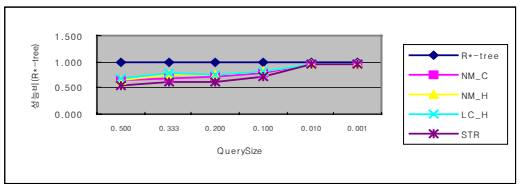
R*-tree의 메모리 사용량은 데이터의 분포에 크게 영향을 받지 않으나, 고정 격자 색인 방법의 메모리 사용량은 데이터가 밀집되어 있을수록 커지기 때문에 100,914개의 비균등 분포의 데이터를 가지는 서울 데이터가 100,000개의 균등 분포의 데이터를 가지는 합성 데이터에 비해서 16배의 메모리 사용량을 보이게 된다.

4.3 질의 처리 성능 비교

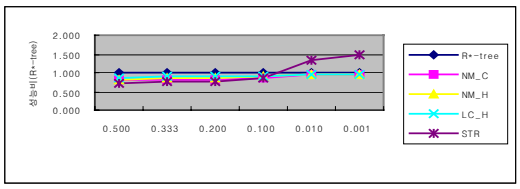
아래 그림 7은 질의처리 성능비교 그림이다. R*-tree를 기준으로 하고 질의 크기에 따라 각 색인 방법들의 성능비를 나타내었다.



(a) 서울 데이터



(b) 몽고메리카운티 데이터



(c) 합성 데이터

그림 7. 질의처리 성능비교

질의의 크기는 한 축의 길이가 1/2, 1/3, 1/5, 1/10, 1/100, 1/000 크기이며 균등분포 질의 1000개를 사용하였다. 위 그림 7은 R*-tree가 1000개의 질의를 처리할 때의 디스크 접근 횟수를 1로 보고 나머지 색인 방법의 질의를 처리할 때의 디스크 접근 횟수의 비율을 그래프로 나타낸 것이다. NM_C 방법은 R*-tree에 대해 평균 83.57%, NM_H는 평균 85.13%, LC_H는 평균 84.07%, STR 방법은 평균 89.6%의 디스크 접근 횟수를 보였다. 그래서 고정 격자 색인 방법에 합병을 적용한 것이 가장 좋은 질의처리 성능을 보이는 것을 알 수 있다. 또한 고정 격자 색인 방법에 여러 합병을 적용한 것들을 비교해 볼 때, 거의 비슷한 성능을 보이지만 그래도 NM_C 방법이 NM_H, LC_H방법보다 조금 더 좋은 것을 알 수 있었다.

본 논문에서 제안한 방법과 STR을 비교해 볼 때는, 몽고메리카운티 데이터와 같이 데이터가 전체 데이터 영역이 분포되어 있지 않고 밀집되어 있는 경우에는 전체적으로 좋은 성능을 보이는 것 같지만, 데이터가 전체 데이터 영역에 퍼져 있고 밀집되어 있는 서울 데이터 같은 경우에는 평균적으로 본 논문의 방법이 우수함을 알 수 있다. 또한 데이터가 균등분포를 이루는 경우에도 역시 본 논문의 방법이 우수함을 알 수 있었다.

5. 결론 및 향후 연구

본 논문에서는 고정 격자 방법의 단순하고 효율적인 특성을 비균일 분포의 데이터에도 적용하기 위해서, 고정 격자 방법으로 전체 데이터 영역을 나눈 뒤, 격자를 합병하는 방법을 제안하였다. 이 방법은 다른 종류의 공간색인 방법과 달리, 색인 구조를 위한 디스크 접근이 필요 없고, 단순히 데이터를 읽기 위한 디스크 접근 만이 필요하므로 성능이 매우 향상되었다. 또한 색인 방법이 단순하여 쉽게 다른 응용 분야에 적용될 수 있다.

이 방법은 단순히 합병만을 고려하여 합병된 디스크 블록들의 디스크 저장위치는 고려하지 않았는데, 이후의 연구에서는 합병된 디스크 블록의 저장위치가 지역성을 보장하도록 하여 성능을 더 향상시킬 수 있다. 그리고 현재 고정 격자 공간 색인을 이용하기 때문에 이 방법은 점 객체에 대해서만 적용이 가능하다. 또한 아직까지는 2차원 객체에 대해서만 적용을 하였다. 따라서 이를 비점객체와 3차원 이상의 다차원 공간의 객체에 적용하는 연구가 계속 이루어져야 한다.

6. 참고 문헌

[1] Jurg Nievergelt, Hans Hinterberger and Kenneth C. Sevcik, “The Grid files : An Adaptive, Symmetric Multikey File Structure”, ACM TODS, vol. 9, No. 1, pages 38-71, 1984.

[2] H. V. Jagadish, “Linear Clustering of bjects with Multiple Attributes”, In Proceeding of SIGMOD Conference, pages 332-342, 1990.

[3] Norbert Beckmann, Hans-Peter Kriegel, Ralf schneider and Bernhard Seeger, “The R*-tree: An Efficient and Robust Access Method for Points and Rectangles”, In Proceeding of SIGMOD Conference, pages 322-331, 1990.

[4] Scott T. Leutenegger, Mario A. Lopez and Jeffrey Edgington, “STR: A Simple and Efficient Algorithm for R-Tree Packing”, In Proceeding of ICDE, pages 497-506, 1997.

[5] Ibrahim Kamel and Christos Faloutsos, “On Packing R-trees”, ACM CIKM, pages 490-499, 1993.