이동 객체에 대한 최근접 질의처리

강혜영[0] 김경숙 이기준
부산대학교 전자계산학과 시공간데이타베이스연구실
{hykang, kskim}@quantos.cs.pusan.ac.kr, lik@hyowon.cc.pusan.ac.kr

Nearest Neighbor Query Processing for Moving object

Hye-Young Kang[0] Kyoung-Sook Kim Ki-joune Li
Dept of Computer Science, Pusan National University

요 약

시간에 따라 공간적 위치가 연속적으로 변하는 객체를 이동객체라 한다. 이동객체에 대한 최근접 질의는 시간에 따라 거리도 달라지게 됨으로써 기존의 방법을 그대로 이용하기 어렵다. 본 논문에서는 이동객체에 대한 최근접 질의를 3D R-tree를 이용한 최근접 탐색방법에 시간적 요소를 추가하여 처리하였다. 또, 3D R-tree를 구성하는 노드의 크기에 따른 사장공간(Dead Space)이 최근접 질의에 미치는 영향을 분석하였다. 실험결과에 따르면 이동객체를 하나의 노드에 저장하면 사장공간이 증가하여 불필요한 노드를 접근함으로써 성능이 떨어지고 여러 개로 분할하여 저장하게 되면 색인의 크기가 커짐으로써 질의 처리 성능이 떨어지게 된다. 따라서, 이동객체에 대한 적절한 분할 및 병합 정책이 필요하다는 것을 알 수 있다.

1. 서 론

시공간 데이터베이스는 공간적 객체의 위치나 면적 등의 기하학적인 속성이 시간에 따라 변하는 객체를 저장한 것을 말한다. 예를 들면, 행정 경계의 변경 또는 비행기나 차, 구름과 태풍과 같은 객체를 다루는 것이다. 특히, 비행기나 차와 같이 시간에 따라, 객체의 공간적인 위치가 연속적으로 변하는 것을 이동객체라고 한다.

이동객체에 대한 질의는 기존의 공간 데이터베이스나 시간 데이터베이스에서 제공하는 방법을 단순히 이용해서는 효율적으로 처리할 수 없다. 따라서, 이동객체의 특수성이 잘 반영된 질의 처리 방법이 필요하다. 예를 들어, " 5시에서 7시 사이에 폭풍지역과 가장 가까운 배를 찾아라." 와 같이, 주어진 질의와 가장 가까운 이동객체를 찾는 최근접 질의는 시간에 따라 그 거리가 달라짐으로써 시간적인 요소도 함께 처리해야 한다.

본 논문에서는 이동객체의 최근접 질의를 처리하는 방법을 소개하고 이동객체의 사장공간(Dead Space)에 의한 질의 처리 성능을 분석하였다. 이동객체에 대한 최근접 질의는 이동객체의 최소경계사각형(MBR)을 이용하여 기존의 공간색인 방법인 R*-tree를 3차원으로 확장한 3D R-tree[1]를 생성하고 [6]에서 제안된 최근접 탐색 방법을 이용하여 처리하는 방법을 제안하였다. 그러나 3차원 공간에서의 거리와 이동객체 사이의 거리는 다르게 해석됨으로 거리를 구하는 방법이 기존의 공간객체 사이의 거리와 달라진다. 또한, 이동객체의 MBR을 어떻게 구성하는가에 따라 색인의 크기와 사장공간의 크기가 달라짐으로 인하여 성능이 달라진다. 따라서, 이동객체의 색인을 생성하기 위한 적절한 분할 방법이 필요하다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 관련 연구에 대해 살펴보고, 3장에서는 이동 객체에 대한 최근접 질의를 처리하는 알고리즘을 제안한다. 4장에서는 이동객체에 대한 사장공간이 질의 처리에 미치는 영향을 알아보고, 5장에서는 실험 결과를 살펴본다. 마지막으로 6장에서 결론을 맺도록 한다.

2. 관련 연구

공간데이터베이스에서의 최근접 탐색방법은 많이 제안되었다. Roussopoulos는 MINDIST와 MINMAX를 이용한 가지치기(Branch-and-Bound) 알고리즘을 제안하였다[2]. 그 외에도 Quadtrees를 기반으로 우선순위큐(priority queue)를 이용한 최근접 탐색 방법이 제안되었다[3]. 그러나, 공간데이터베이스의 최근접 탐색 방법들을 그대로 이동객체에 이용하기에는 부적절하다. 이에 따라 이동 객체에 대한 최근접 질의를 처리하는 방법들이 계속해서 제안되고 있다.

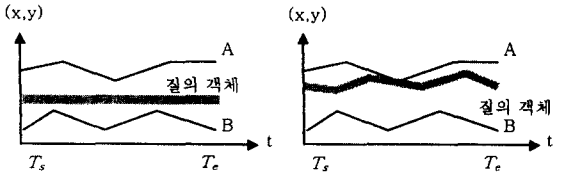
이동 객체에 대한 최근접 질의는 2가지 경우로 나누어 볼 수 있다. 이동하는 객체가 실제 질의를 처리하고자 하는 객체인 경우와 질의 자체인 경우이다. [4]에서는 1차원상의 이동객체에 대하여 시간차원을 추가하여 2차원으로 표현하여 이동객체에 대한 최근접 질의를 처리하는 방법을 제안하였고, [5]에서는 실제 처리되는 대상은 고정된 반면에 질의 자체가 시간에 따라 움직임으로써 기존의 공간데이터에 이동 최근접 질의를 처리하는 방법이 제시되었다. 그러나, 실제 시공간 데이터베이스에서는 2차원 공간에서 시간에 따라 움직이는 객체를 다루고 있고, 질의도 시간에 따라 다르게 주

어질 수 있다. 따라서, 이동의 개념이 객체와 질의 모두에 적용될 수 있어야 한다.

본 연구에서는 이동 객체가 2차원 공간에서 시간에 따라 변한다는 것에 중점을 두어, 이동 객체를 시간에 따른 3차원의 점으로 표현하여 최근접 질의를 처리한다. 최근접 질의는 공간적인 위치로 처리하게 됨으로써 영역을 가지는 객체도 그 중심으로 표현하여 최근접 질의를 처리할 수 있다.

3. 이동객체의 최근접 질의 처리 알고리즘

본 논문에서는 최근접 질의 처리에 Roussopoulos가 [2]에서 제안한 가지치기(Branch-and-Bound) 알고리즘에 CPU 시간을 줄여 성능을 개선한 [6]의 탐색 방법을 사용하였다. 가지치기 알고리즘은 MINDIST를 구하여 현재 유지하고 있는 최근접 객체와의 거리보다 큰 경우에는 가지치기를 하면서 R*-tree를 탐색한다. 그러나, 이동객체 사이의 거리는 단순히 3차원상의 유클리디안 거리(Euclid Distance)로 정의 될 수 없다. 따라서, 시간과 공간 사이의 관계를 이용하여 알고리즘에서 거리를 구하는 방법을 재정의한다. 본 논문에서 이동객체의 최근접 질의를 주어진 시간 구간 내에서 공간적 위치가 변하지 않는 고정 질의(Fixed Query)와 시간에 따라 공간적 위치가 변하는 이동질의(Moving Query)로 나누어 처리한다.



(가) 고정최근접질의 (나) 이동최근접질의

그림 1. 이동객체에 대한 최근접 질의

고정최근접질의는 그림1의 (가)와 같이 주어진 시간 구간 [T_s, T_e]내에서 (x,y)에 위치한 질의 객체와 가장 가까운 이동객체를 찾는 것이다. 고정최근접질의의 처리는 질의 시간구간과 교차되는 이동객체의 MBR을 모두 구하여 각 MBR을 xy평면에 투영하여 기존의 공간데이터에 사용한 알고리즘을 그대로 이용하여 해결할 수 있다. 그러나, 질의 자체도 이동객체로 주어진 경우에는 xy평면에 투영된 이동객체의 MBR과의 거리로 최근접 질의를 처리할 수 없다. 따라서, 이동객체 사이의 거리를 구하는데 부가적인 작업이 필요하다.

그림1의 (나)와 같은 경우는 이동 최근접 질의를 나타낸 것이다. 이 경우 질의 시간 구간 내에 질의 객체도 움직이고 있다. 일반적으로 이동객체의 움직이는 궤적을 다각선 형태로 나타낼 수 있다. 따라서, 본 논문에서 이동객체의 최단거리는 각 다각선의 선분요소 사이의 최단거리로 정의 한다. 그러나, 그림 2의 a와 같이 일반 3차원 공간상의 거리를 구하는 것은 의미가 없다. 왜냐하면, 선분요소 사이에 가장 가까운 점이 서로 다른 시간인 t_1과 t_2에 해당한다면 이는 서로 근접해 있다고 말할 수 없기 때문이다.

따라서, b와 같이 서로 교차되는 시간 사이에 같은 시간에 대한 거리를 구하여 최근접 질의를 처리해야 한다.

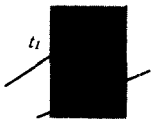


그림2. 이동 객체 사이의 거리

만약, 점객체가 일차적으로 이동한다면 t시간에서의 xy 평면상의 좌표는 시간 t에 대한 일차함수로 나타낼 수 있다.

$$x = f_x(t) = ax + b, y = f_y(t) = cy + d$$

따라서, t시간에 서로 움직이는 이동객체 사이의 거리는 다음과 같이 나타낸다.

$$d(t) = | f(t) - g(t) | = \sqrt{\{f(t) - g(t)\}^2}$$
$$= \sqrt{\{f_x(t) - g_x(t)\}^2 + \{f_y(t) - g_y(t)\}^2}$$

이와 같은 방법으로 이동객체 사이의 거리를 구하여 이동 최근접 질의를 처리한다. 이동 최근접 질의 처리의 단계는 다음과 같다.

이동질의객체의 각 선분 요소에 대해,

단계1: 3D R-tree의 노드에서 이동객체의 MBR 중 질의객체의 시간구간과 교차되는 노드를 찾는다.
단계2: 각 노드의 엔트리에 대해서 MINDIST를 이용한 최근접 탐색방법을 이용하여 시간구간내에 가장 가까운 이동객체를 찾고 그 거리를 구한다.
단계3: 지금까지 구한 최단거리와 비교하여 최단거리를 수정한다.
단계 4: 다음 선분요소를 가지고 단계1-3을 반복한다.

4. 이동객체의 사장공간(Dead Space)의 영향

시간에 따라 움직이는 이동객체의 3차원 MBR의 사장공간은 공간축인 x,y와 시간축인 t에 모두 영향을 받는다.

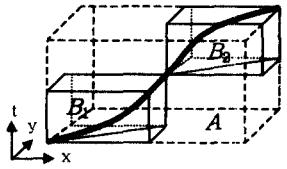


그림 3. 이동객체와 사장공간

그림 3과 같이 이동객체는 시간에 따라 위치가 변하게 됨으로써 A와 같은 이동객체의 3차원 MBR이 생기게 되고 이로 인해 사장공간이 늘어나게 된다. 이처럼 사장공간이 늘어나게 되면 불필요한 데이터도 접근하여 읽어옴으로써 최근접 질의 처리의 성능을 저하시킨다. 따라서, 그림 3의 B_1, B_2와 같이 이동객체의 궤도를 분할하여 MBR을 형성하는 것이 사장공간을 줄일 수 있

는 방법이다. 그러나, 사장 공간을 최소화하기 위하여 이동객체의 궤도를 너무 작게 분할하게 되면 하나의 이동객체가 여러 개로 나뉨으로써 노드에 저장되는 객체수가 증가하여 색인의 크기가 커지게 된다. 이는 질의 처리시 색인 자체에 접근하는 횟수가 많아짐으로써 성능을 저하시키게 된다. 따라서, 어느 정도 적당한 크기로 이동객체의 MBR을 분할하거나 병합시키는 것이 필요하다.

본 논문에서는 시간축에 의한 사장공간보다는 공간적인 x,y축에 의한 평균길이를 고려하여 공간적인 사장공간을 줄이는 방향으로 이동객체의 궤도를 분할 또는 병합하였다. 왜냐하면 최근접 질의는 공간적 사장공간이 크게 되면 가지치기가 제대로 되지 않아 많은 데이터를 계산해야 함으로 성능을 더욱 떨어뜨릴 수 있기 때문이다. 이동객체의 최근접 질의 성능은 색인의 크기가 줄어드는 시점과 공간적 사장공간의 크기가 작아지는 시점에서 좋은 성능을 나타내게 된다. 다음 장에서 MBR의 평균길이의 임계값에 대하여 공간적 사장공간이 질의 처리에 어떠한 영향을 미치는지 살펴본다.

5. 실험 및 결과

본 장에서는 앞에서 설명한 것과 같이 3D R-tree 노드의 사장공간에 대한 임계값에 따른 고정질의와 이동질의의 처리 성능을 알아본다. 실험을 위한 데이터는 객체 100개를 200개의 시간간격으로 임의 생성한 가상데이터와 같은 객체수로 생성한 GSTD[7]데이터이고, 최근접질의를 1000개를 생성하여 실험하였다. 이동객체의 분할과 병합을 하기 위한 방법은 노드의 평균 길이를 사용하였다. 그림 4와 그림 5는 각각 평균 MBR의 길이에 따른 고정최근접질의와 이동최근접질의를 처리한 결과이다.

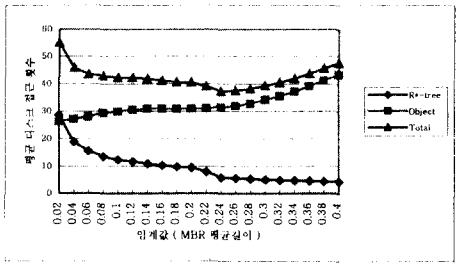


그림 4. MBR 평균길이에 따른 고정최근접질의 디스크접근횟수

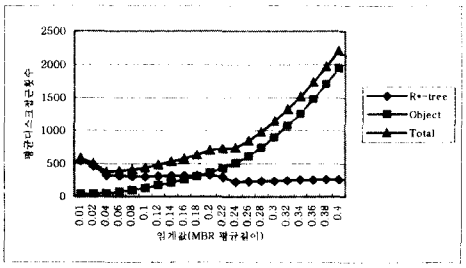


그림 5. MBR 평균길이에 따른 이동최근접질의 디스크접근횟수

두 경우, 이동객체의 MBR의 크기가 증가하면 3D R-tree의 노드 접근 수는 작아지는 반면에 사장공간이 증가함으로써 객체를 읽어오는 횟수가 증가하게 됨으로써 전체 평균 디스크 접근 횟수가 증가하게 되고, MBR의 크기가 너무 작으면 노드의 사장공간은 작지만 노드가 많아짐으로써 색인을 접근하여 읽어오는 노드 수는 증가한다. 그림 4의 고정최근접질의의 경우에는 약 0.25정도의 MBR크기에서 최적의 노드접근수를 보이고 있고 그림 5의 이동최근접질의의 경우에는 약 0.04에서 최적의 질의처리성능을 보여주고 있다. 따라서, 최근접 질의의 처리 성능은 이동객체의 MBR크기에 의한 사장공간과 객체 수에 의해 어느 일정한 값에서 최적의 성능을 보여줌을 실험을 통해서 알 수 있다.

6. 결론

본 논문에서는 3D R-tree를 이용하여 이동객체에 대한 최근접 질의 처리 방법을 제안하였고, 이동객체의 사장공간이 최근접 질의 처리에 어떤 영향을 미치는지 알아보았다. 실제 사장공간을 줄이기 위해서 이동객체를 여러 개로 분할하여 저장하게 되면 객체수가 증가하여 색인의 크기가 커지게 됨으로써 질의 처리 성능을 떨어뜨리게 된다. 반면에, 이동객체를 하나의 노드에 저장하면 사장공간이 늘어남으로써 질의 처리 성능을 떨어뜨리게 된다. 따라서, 이동객체의 움직임의 특성에 따른 적절한 분할 및 병합 정책이 필요하다는 것을 본 논문에서 보이고 있다.

7. 참조문헌

[1] Yannis Theodoridis, Michael Vazirgiannis and Timos Sellis: " Spatiao-temporal indexing for large multimedia applications" , Proc. IEEE Conf. on Multimedia Computing and System, pp.441~448, 1996

[2] Nick Roussopoulos, Stephen Kelley, and Frederic Vincent: " Nearest Neighbor Queries" , Proc. ACM-SIGMOD, pp 71~79, 1995

[3] Gisli R. Hjaltason and Hanan Samet: " Ranking in Spatial Databases" , Proc. SSD, pp.83~95, 1995

[4] Georage Kollios, Dimitiros Gunopulos and Vassilis J. Tsotras: " Nearest Neighbor Queries in a Mobil Environment" , Proc. Int. Workshop on Spatio-Temporal Database Management, pp.119~134, 1999

[5] Zhexuan Song and Nick Roussopoulos: " K-Nearest Neighbor Search for Moving Query Point" , Proc. SSTD, pp. 79~96, 2001

[6] King Lum Cheung and Ada Wai-chee Fu: " Enhanced Nearest Neighbour Search on the R-tree" , SIGMOD Record Vol.27(3), pp.16~21, 1998

[7] Yannis Theodoridis, Jefferson R.O. Silva and Mario A. Nascimento: " On the Generation of Spatiotemporal Datasets" , Proc. SSD, pp.147~164, 1999