이동체 데이터베이스를 위한 R-tree 기반 색인구조에서 궤적 클러스터를 사용한 분할 정책

김진곤[O] 전봉기[•] 홍봉희
부산대학교, 신라대학교[•]
{jgkim11[O], bhhong}@pusan.ac.kr, bgjun@silla.ac.kr[•]

Splitting policies using trajectory clusters in R-tree based index structures for moving objects databases

Jingon Kim[O] Bonggi Jun[•] Bonghee Hong
Pusan National University, Silla University[•]

요 약

이동체 데이터베이스를 위한 과거 궤적 색인으로 R-tree계열이 많이 사용된다. 그러나 R-tree계열의 색인은 공간 근접성만을 고려하였기 때문에 동일 궤적을 검색하기에는 많은 노드 접근이 필요하다. 이동체 색인의 검색에서 영역 질의와 궤적 질의는 공간 근접성과 궤적 연결성과 같이 상반된 특징으로 인하여 함께 고려되지 않았다. 이동체 색인에서 영역 질의의 성능개선을 위해서는 노드 간의 심한 중복과 사장 공간(Dead space)을 줄여야 하고, 궤적 질의의 성능 개선을 위해서는 이동체의 궤적 보존이 이루어져야 한다. 이와 같은 요구 조건을 만족하기 위해, 이 논문에서는 R-tree 기반의 색인 구조에서 새로운 분할 정책을 제안한다.

제안하는 색인 구조의 노드 분할 정책은 궤적 클러스터링을 위한 동일 궤적을 그룹화해서 분할하는 공간 축 분할 정책과 공간 활용도를 높이는 시간 축 분할 정책을 제안한다. 본 논문에서는 R-tree기반의 색인 구조에서 변경된 분할 정책을 구현하고, 실험 평가를 수행한다. 이 성능 평가를 통해서 검색성능이 우수함을 보인다.

1. 서 론

이동체는 시간에 따라 연속적으로 위치 정보가 변경되는 시공간 객체이다. 이동체의 과거 궤적은 3차원 점들을 연결한 선분들(Line Segments)로 표현되며, 이를 저장 관리하기 위해 시공간 색인이 사용된다.

이동체 색인에 있어서 질의 종류로는 크게 영역 질의, 타임스탬프(timestamp) 질의, 궤적 질의, 복합 질의로 나누어진다. 영역 질의는 주어진 시간 간격 동안에 공간 윈도우(spatial window)에 속하는 이동체들을 검색하는 질의이다. 타임스탬프 질의는 특정 시간에 주어진 공간 윈도우에 속하는 이동체들을 검색하는 질의이다. 궤적 질의는 특정한 이동체의 궤적을 검색하는 질의이다. 복합 질의는 " 특정 시간에 주어진 영역을 지나간 객체의 궤적을 검색하라" 와 같이 시공간 도메인의 영역 질의와 궤적 질의가 복합된 질의이다.

이동체 궤적을 3D R-tree에 저장하면 보편적으로 TB-tree보다 영역 질의에서는 우수한 성능을 보인다. 그러나, 노드 간의 중복이 많아 검색 성능이 저하되고, 궤적의 삽입 순서가 시간 축으로 증가하는 특징으로 인하여 공간 활용도가 낮다는 문제점이 있다. 3D R-tree는 궤적의 보존(Trajectory preservation)에 관한 개념을 고려하지 않기 때문에, 궤적 질의를 처리하기 위해서 점 질의(point query)를 이용하여 반복 검색하기 때문에 질의 처리 비용이 크다는 문제점이 있다. 이 논문에서는 R-tree의 색인 구조에서 한 노드 내에 동일 궤적을 최대한 많이 저장하기 위한 동일 궤적 그룹화에 따른 노드 분할 정책을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구를 기술하고, 3장에서 문제 정의를 설명하고, 4장에서 단말 노드의 공간 축 분할 정책, 시간 축 분할 정책을 제안한다. 5장에서는 성능평가 결과를 분석하고, 끝으로 결론 및 향후 연구를 기술한다.

2. 관련연구

이동체의 연속적인 이동을 표현하기 위해, 선분을 저장하는 색인 구조에 관한 기존 연구로서는 3D R-tree[1], STR-tree[2], TB-tree[2] 등이 있다. 3D R-tree는 2차원 공간 색인에 시간 축을 추가한 3차원 색인이다. 이동체의 궤적인 선분들을 3D R-tree에 저장하면 중복이 심하고, 사장 공간이 크다는 문제점이 있다. STR-tree는 R-tree의 변형으로서, 보존 파라미터(Preservation Parameter)를 사용하여 같은 객체의 궤적을 근접한 페이지에 저장하도록 유도한 색인이다. 하지만 STR-tree는 3D R-tree에 비해 영역 질의와 궤적 질의에서 성능 향상이 없다. TB-tree는 궤적 보전을 위해 단말 노드에 하나의 궤적만을 저장함으로써, 궤적 질의에서는 우수한 성능을 보인다. 하지만 단말 노드 간의 중복이 심하다는 문제점으로 인하여 영역 질의의 성능이 나쁘다.

노드 간의 중복을 줄여서 색인의 성능을 향상 시키는 기존의 공간 색인에 연구로서는 R*-tree[4], R+-tree[5], X-tree[6]가 있다. R*-tree는 영역(Area) 외에 중복(Overlap)과 가장자리(Perimeter)를 사용하여 노드 간의 중복을 줄이고자 하였다. 하지만 이동체와 같이 다차원 데이터 간의 심한 중복(high overlap)이 있는 경우에는 R*-tree의 분할 알고리즘은 노드 간의 중복을 줄일 수 없다. R+-tree는 절단 방법(clipping-based scheme)을 사용하는 색인으로서 노드 간의 중복을 허용하지 않는다. 하지만 R+-tree는 CPU 비용이 증가하고 절단으로 인한 색인의 크기가 증가하는 문제점이 있다. X-tree는 노드 분할 시에 노드 간의 중복 비율을 고려하여, 허용치를 초과하는 분할을 허용하지 않음으로써 노드 간의 중복을 회피하는 정책을 사용한다. 즉, 중복이 심한 노드를 분할할 때는 분할을 수행하지 않고, 슈퍼 노드(Super node)를 사용하여 노드의 최대 허용 엔트리 수를 증가시킨다. 하지만 X-tree는 슈퍼 노드로 인하여 노드 간의 중복을 줄였지만, 4차원 이하

의 색인에서 R-tree와 같은 검색 성능을 보인다.

기존의 클러스터링에 관한 연구는 정적인 데이터를 기준으로 수행되었다. 하지만 이동체의 위치 데이터는 삽입 연산을 수반하는 동적 데이터로서 cR-tree와 같이 클러스터링을 기반으로 하는 분할 정책을 사용하는 접근 방법이 적합하다. cR-tree[7]는 노드 분할 시 공간 근접성을 기반으로 하는 k-means 알고리즘을 사용하여 클러스터링을 하며, 이진 분할 및 다중 분할을 한다. 이 연구에서는 궤적 연결성을 기반으로 분할 노드의 엔트리들을 클러스터링하여 분할하는 정책을 제안한다.

3. 문제 정의

3D R-tree에서 복합 질의를 수행하기 위해 두 단계의 처리 과정이 필요하다. 첫 번째 단계는 특정한 영역에 대한 영역 질의(Range query)를 수행해야 하고, 두 번째 단계는 영역 질의 결과에 대한 이동체의 궤적을 점 질의(Point query)를 수행하여 동일 궤적을 검색해야 한다. 3D R-tree는 최소 영역 증가 정책(Least Area Enlargement Policy)과 중복 최소화 정책(Least Minimum Overlap Policy)에 따른 이동체 궤적을 삽입한다. 이러한 정책들은 이동체 궤적의 공간 근접성만으로 고려하였다. 즉, 동일 궤적이 서로 다른 단말 노드에 저장될 확률이 높기 때문에 연결되어진 궤적을 검색하기 위해서는 많은 노드 접근이 필요하다.

이러한 문제점을 해결하기 위해서는 우선 영역질의 성능개선이 필요하고, 그 다음으로 궤적 질의 성능 개선이 필요하다. 영역 질의에서는 노드간의 중복과 사장 공간을 최소화하여야 하고, 색인의 공간 활용도를 높이는 정책이 필요하다. 궤적 질의에서는 한 단말 노드에 동일 궤적이 최대한 많이 저장 될 수 있는 삽입 정책이 필요하다. 동일 궤적을 검색 하기 위해 점 질의를 수행 할 때, 한 단말 노드 내에 동일 궤적이 최대한 많이 저장됨으로써 노드의 방문을 줄일 수 있다.

4. 삽입 정책

4.1 하위 노드 선택(Choose Subtree)

단말 노드에 동일 궤적이 최대한 많이 저장하기 위해서는 우선 하위 노드 선택(ChooseSubtree)시 동일 궤적이 동일한 단말 노드에 삽입이 될 확률이 높아야 한다. 하나의 선분이 삽입이 이루어 질 때 기존의 3D R-tree의 경우에는 최소 영역 증가 정책(Least Area Enlargement Policy)를 이용하여 단말 노드를 선택한다. 3D R-tree의 이러한 정책은 하나의 단말 노드에 동일 궤적을 최대한 많이 삽입시키기에는 부적합하다. 따라서, 본 논문에서는 우선 3D R-tree의 최소 영역 증가 정책을 이용해서 선택된 단말 노드와 형제 노드들 간의 최소 중복율을 검사해서 임계값보다 클 경우에는 이전에 보고했던 선분이 저장되어진 노드를 찾아 삽입을 하였다. 본 논문에서 실험을 해본 결과 최소 중복율 임계값이 30%일 경우가 가장 좋은 것으로 나왔다.

4.2 공간 축 분할 정책

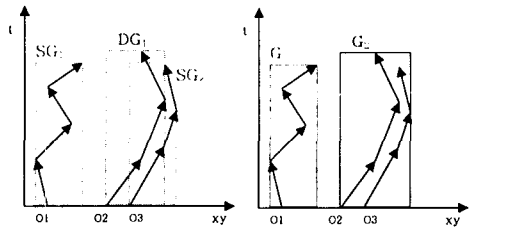
단말 노드의 공간 축 분할은 궤적을 보전하기 위한 분할 정책이다. 새로운 선분의 삽입에 따른 단말 노드의 오버플로우가 발생하면 두 개의 노드로 분할을 한다. 분할 과정은 다음과 같이 3단계 과정으로 이루어진다. 1)동일 궤적 그룹화 과정, 2)그룹들 중 시드 선택 과정, 3)나머지 그룹 분배 과정.

첫 단계로 동일 궤적끼리 그룹을 먼저 생성한다. 즉, n을 단말 노드 내에 저장되어진 이동체의 수라고 가정한다면, n만큼의 그룹을 생성한다. 두 번째 단계로, 그룹들 중에 공간적으로 가장 멀리 떨어져 있는 두 개의 그룹을 시드로 선택한다. 선택된 두 개의 그룹을 시드그룹(SG : Seed Group)이라 정의한다. 세 번째 단계로, 시드그룹을 제외한 나머지 그룹을 분배그룹(DG : Distribution Group)이라고 정의하고, 그룹의 공간 근접성에 따라 각 시드그룹에 분배를 한다. 그림 1(a)는 첫 번째 단계로 같은 궤적끼리 그룹핑한 것이고, 그림 1(b)는 시드그룹에 분배그룹을 분배해서 두 개의 노드로 분할 된 결과이다.

정의 1. 분할 노드 간의 중복율

궤적 그룹화 과정에서 분할되어 생성되는 두 개의 노드를 G_i, G_j 라 한다. 노드내의 궤적을 모두 포함하는 최소경계박스(MBB)를 그룹의 MBB라 하면, 분할 노드 간의 중복율은 다음과 같다.

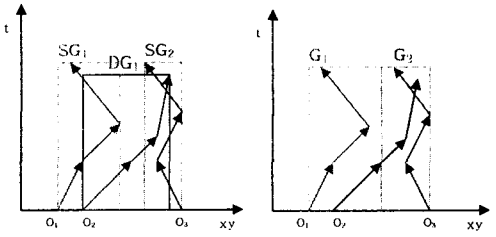
$$OverlapRate(G_i, G_j) \leftarrow area(G_i.MBB \cap G_j.MBB) / min(area(G_i.MBB), area(G_j.MBB))$$



(a) 궤적 그룹화 (b) 두 개의 노드로 분할된 결과

그림 1. 공간 축 분할

분배그룹(DG_i)을 각 시드그룹(SG)에 분배 할 때 분할 노드 간의 중복율이 임계값을 넘으면, 분할 노드 간의 심한 중복이 발생한 것이다. 이러한 경우에는 심한 중복이 되지 않도록 분배그룹을 궤적의 구성 요소인 선분으로 분리하여 분배한다. 그림 2(a)는 분배그룹이 각 시드그룹과 심한 중복이 발생한 경우이고, 그림 2(b)는 궤적을 분리하여 중복이 최소화 되도록 두 개의 노드로 분할한 결과이다.

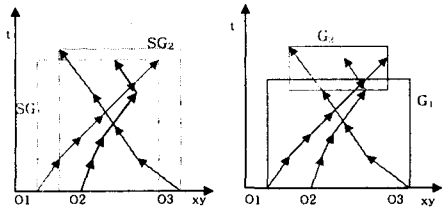


(a) 그룹간의 심한 중복의 예 (b) 궤적을 분리하여 분배하는 예

그림 2. 심한 중복 발생시 분배그룹의 궤적 분리

4.3 시간 축 분할 정책

공간 축 분할일 경우, 첫 번째 단계로서 동일 궤적끼리 그룹을 생성하고 다음 단계로서 시드그룹을 선택 한다. 시드그룹(SG) 간에 심한 중복이 발생할 수 있다. 이러한 경우에는 분배그룹(DG)를 분배를 하더라도 심한 중복이 발생하므로 시간 축으로 분할한다.



(a) 시드그룹 간의 심한 중복 발생 (b) UC 그룹의 예

그림 3. 시간 축 분할

이동체가 가장 최근에 보고한 위치를 포함하는 선분을 UC

선분(UC line segment)이라 한다. 시간 축 분할은 그림 3(b)와 같이 UC선분만을 포함하는 UC그룹과 나머지 선분을 포함하는 그룹으로 분할한다. 이동체의 위치 정보는 시간 축으로 증가하는 특징을 가지므로 UC그룹의 엔트리 수는 노드의 최소 엔트리 수(m)보다 작은 경우를 허용한다.

5. 실험

5.1 실험 데이터

본 논문에서 사용한 실험 데이터는 GSTD(Generate Spatio-Temporal Data) 알고리즘을 이용하여 이동체 데이터를 생성했다. GSTD 생성기를 통해서 여러 타입의 데이터를 생성 할 수 있지만, 본 논문에서는 사향 분포(Skewed distribution)인 데이터를 사용하였고, 각 이동체 수에 따라 1000번을 보고하는 데이터 셋을 사용하였다.

5.2 실험 평가

색인의 성능을 평가하기 위해 색인의 공간 활용도, 영역 질의 성능과 복합 질의 성능을 평가하였다. 각 질의의 성능은 노드 접근 횟수로 측정을 하고, 영역 질의의 크기는 각 도메인 크기의 5%, 10%이다. 또한 복합 질의의 크기는 Inner Range 1%에 대한 Outer Range 5%, 5%-10%로 실험을 하였다. 그리고, 시드그룹(SG) 간의 심한 중복율을 30%, 35%, 40%일 경우에 대해 실험을 달리 하였다.

5.2.1 공간활용도의 비교

그림 4에서 GSTD 알고리즘으로 생성한 사향분포 데이터 집합에서 이동체 수의 변화에 따른 공간 활용도 변화를 보여준다. 제안한 색인은 시간 축의 비균등 분할의 효과로 61% ~ 58%의 공간 활용도를 가진다.

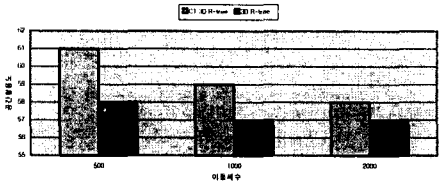


그림 4. 이동체 수에 따른 공간 활용도

5.2.2 영역 질의 성능 비교

그림 4와 5는 시드그룹 간의 중복 율에 따른 영역 질의에 대한 성능 비교한 것이다. 이동체수와 보고횟수 비율에 따라 조금의 차이는 있지만 3D R-tree와 비슷한 성능을 보이고 있다.

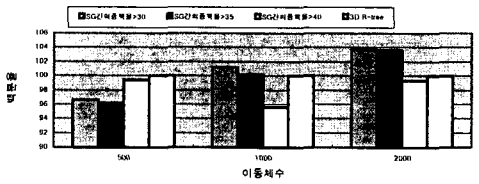


그림 4. 영역 질의(5%)

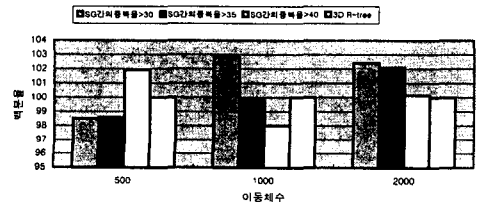


그림 5. 영역 질의(10%)

5.2.3 복합 질의 성능 비교

그림 6와 7는 시드그룹 간의 중복 율에 따른 복합 질의에 성능 비교이다. 시드그룹 간의 중복 율이 35%이상일 경우에 한 노드에 동일 궤적이 저장될 확률이 더 많기 때문에 복합 질의에서 더 좋은 성능을 보인다.

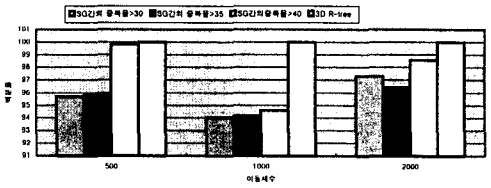


그림 6. Inner 1%-Outer 5%

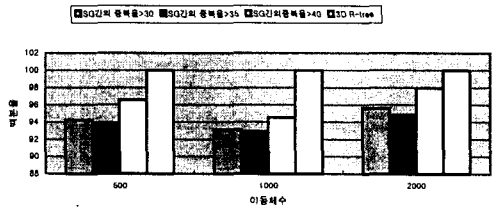


그림 7. Inner 5%-Outer 10%

6. 결론 및 향후 연구

이동체의 연속적인 움직임을 저장하는 이동체 색인에서의 문제점을 기술하고, 궤적 클러스터링 기반의 분할 기법을 제안하고 제안한 방법을 구현하여 성능평가를 실시하였다. 단말 노드 분할 시 시드그룹 간의 중복 비율에 따라 실험한 결과 3D R-tree 보다 영역 질의 성능은 비슷하나 복합 질의에서 성능이 우수하다는 것을 알 수 있었다.

향후 연구로써 더 많은 데이터 성능평가와 다양한 데이터로써 실험이 필요하며, 비단말 노드 분할 정책에 대한 연구가 필요하다.

7. 참고문헌

[1] Yannis Theodoridis, Michael Vazirgiannis, Timos Sellis, " Spatio-Temporal Indexing for Large Multimedia Applications" , In International Conf, on Multimedia Computing and Systems, pp. 441-448, 1996

[2] Dieter Pfoser, Christian S. Jensen, Yannis Theodoridis, " Novel Approaches to the Indexing of Moving Objects." , In Proc. of the 26th VLDB Conf, pp. 395-406, 2000.

[3] Antonm Guttman, " R-Trees: A Dynamic Index Structure for Spatial Searching" , In Proc. of ACM-SIGMOD Conf. on the Management of Data, pp. 47-57, 1984.

[4] N. Beckmann and H. P. Kriegel, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles", In Proc. ACM SIGMOD, p332-331, 1990.

[5] T. K. Sellis, N. Roussopoulos and C. Faloutsos, "The R+ -Tree: A Dynamic Index for Multi-Dimensional Objects", Proceedings of the 13th VLDB Conference, p507-518, 1987.

[6] S. Berchtold, D. A. Keim, H. P. Kriegel, " The X-tree : An Index Structure for High-Dimensional Data" , International Conference on Very Large Data Bases, p28-39, 1996.

[7] S. Brakatsoulas, D. Pfoser, and Y. Theodoridis. Revisiting R-tree construction principles. Technical report, Computer Technology Institute, Patras, Greece, 2002. http://dias.cti.gr/~pfoser/clustering.pdf.