도로 위에 존재하는 이동객체의 궤적에 대한 네트워크 기반의 색인 방법

김 경 숙[†] · 이 기 준[††]

요 약

최근 많은 연구들이 유클리디안 공간을 기반으로 대용량의 이동객체 궤적 데이터를 효율적으로 다루기 위해 진행되어 왔다. 그러나, 일반적으로 실제 응용분야에서는 이동객체가 움직일 수 있는 공간은 제한적이다. 예를 들어, 도로 위에 존재하는 자동차들은 서로 연결된 도로망을 통해서만 이동할 수 있다. 본 논문에서는 도로 위의 이동객체 궤적에 대하여 시공간 영역 질의를 처리할 수 있는 네트워크 기반의 색인방법을 제안한다. 제안된 방법은 질의 처리 과정에서 네트워크 상의 거리를 이용하기 위해서 도로 네트워크의 연결정보를 색인구조에 포함한다. 그리고, 이동객체의 위치 정보를 도로를 구성하는 도로 선분요소 정보를 이용하여 표현하고 이를 단위로 여러 개의 R-tree를 사용하여 이동객체 궤적들을 관리한다. 또한, 여러 개의 도로 선분요소 사이에 하나의 R-tree를 공유할 수 있는 구조를 가짐으로써, 큰 도로 네트워크 데이터에서도 적용할 수 있다. 우리는 실험을 통해서 네트워크 거리를 기반으로 시공간 영역질의를 처리하는 것은 기존의 유클리디안 거리 기반의 방법들보다 대략 30%의 성능향상을 보임을 보여준다.

키워드 : 도로기반 시공간 영역질의 처리, 색인기법, 궤적, 이동객체, 도로 네트워크

A Network-based Indexing Method for Trajectories of Moving Objects on Roads

Kyoung-Sook Kim[†] · Ki-Joune Li[††]

ABSTRACT

Recently many researchers have focused on management of historical trajectories of moving objects in Euclidean spaces due to numerous sizes of accumulated data over time. However, the movement of moving objects in real applications generally has some constraints, for example vehicles on roads can only travel along connected road networks. In this paper, we propose an indexing method for trajectories of moving objects on road networks in order to process the network-based spatiotemporal range query. Our method contains the connect information of road networks to use the network distance for query processing, deals with trajectories which are represented by road segments in road networks, and manages them using multiple R-trees assigned per each road segment. Furthermore, it has a structure to be able to share R-tree among several road segments in large road networks. Consequently, we show that our method takes about 30% less in node accesses for the network-based spatiotemporal range query processing than other methods based on the Euclidean distance by experiments.

Key Words : Network-Based Spatiotemporal Range Query Process, Access Method, Trajectory, Moving Object, Road Network

1. 서 론

무선 통신과 위치 측위 기술의 발달로 이동객체의 지리적 위치정보를 많은 응용분야에서 사용하게 되었다. 특히, 텔레매틱스 서비스는 도로 위에 이동 차량의 위치정보를 이용하여 운전자 및 이용자에게 교통 정보, 사고, 응급 상황, 최단 경로 검색 등의 서비스를 제공하고, 지능형 교통 관리시스템에서는 자동차의 움직임을 분석하여 교통정체를 관리하기도 한다. 이러한 응용분야에서 실시간으로 이동차량의 위치를 추적하는 것도 중요하지만 추적된 위치정보를 데이터베이스에 저장하고 이를 효율적으로 처리하기 위한 기반 기술 또한 필요하다. 이동객체 데이터베이스 시스템은 자동차와 같이 시간에 따라 자신의 위치정보가 연속적으로 변하는 이동객체를 다루기 위한 데이터베이스 시스템으로 90년대 후반부터 활발히 연구가 진행되어 왔다. 이동객체를 위한 데이터 모델링, 저장시스템, 질의 처리, 색인 기법 등 많은 연구들이 이동객체를 효율적으로 표현, 저장, 처리하기 위해

※ 본 논문은 한국전자통신연구원 "베이스스테이션 집중형 센서네트워크 데이터 처리 최적화 기법 연구(0501-2006-0030)"에 관한 위탁과제의 지원에 의해 수행됨
† 준 회 원 : 부산대학교 대학원 전자계산학과 박사과정
†† 정 회 원 : 부산대학교 전자전기정보컴퓨터 공학부 교수
논문접수 : 2006년 8월 25일, 심사완료 : 2006년 10월 30일

제안되었다. 이들 연구의 대부분은 이동객체가 자유롭게 움직일 수 있는 유클리디안 공간을 기반으로 연구되었다. 그러나, 실제 응용분야에서는 도로 네트워크 공간과 같이 움직임에 제약이 있는 공간도 존재한다.

일반적으로 도로 네트워크 공간은 유클리디안 공간과 비교하여 몇 가지 다른 특성을 가진다. 첫째는 객체의 위치정보를 표현하는 방법이 틀리다. 둘째는 도로 상의 두 객체의 거리는 유클리디안 공간 상의 거리계산과 다르다. 셋째는 공간의 차원이 다르게 해석될 수 있다. 기존의 연구들은 이러한 도로 공간의 특성을 고려하지 않음으로써, 실제 도로 네트워크 상에 움직이고 있는 자동차와 같은 이동객체를 효율적으로 다루는 데는 문제점을 가진다. 따라서, 우리는 이동객체의 움직임을 처리하는 과정에서 객체가 존재하는 공간적 특성도 함께 고려해야 한다.

본 논문에서는 도로 위에 존재하는 이동객체의 궤적에 대한 도로 네트워크 기반의 색인기법을 제안한다. 기존의 많은 연구에서는 이동객체의 위치정보를 (x, y, t) 3차원 공간 상의 좌표로 표현하고 질의 처리에 있어서도 유클리디안 공간의 특성을 이용하여 처리해왔다. 그러나 이러한 방법은 도로 네트워크에 존재하는 이동객체의 움직임을 처리하는 경우에는 비효율적이다. 우리는 이동객체의 움직임을 도로 정보를 이용하여 표현하고 도로 네트워크를 기반으로 시공간 영역질의를 정의한다. 그리고, 정의된 시공간 영역질의를 처리하기 위한 색인 방법을 제안한다.

논문의 구성은 다음과 같다. 2장에서 관련 연구를 정리하고, 도로 네트워크 공간 상에서 움직이는 이동객체의 궤적 표현과 도로 기반의 시공간 영역 질의를 3장에서 정의한다. 4장에서는 도로기반의 시공간 영역 질의를 효율적으로 처리하기 위한 네트워크 기반의 이동객체의 궤적 색인기법을 기술하고 5장에서 실험을 통해 제안된 색인기법의 성능을 분석한다. 마지막으로 6장에서 향후 연구와 함께 결론을 맺는다.

2. 관련 연구

이동객체 데이터베이스 관리시스템이란 시간의 흐름에 따라 객체의 위치나 모양, 크기 등의 기하학 정보가 연속적으로 변하는 객체를 저장하고 처리하기 위한 데이터베이스 관리시스템을 말한다. 이러한 이동객체 데이터베이스 관리시스템을 개발하기 위해서 이동객체를 위한 데이터 모델링에서부터 질의 처리기, 색인 기법, 그리고 데이터 생성기 부분까지 다양한 연구가 진행되어 왔다. 특히, 이동객체를 위한 색인기법은 연속적으로 움직이는 객체 정보의 갱신과 질의 처리의 성능 향상을 위해서 필수적인 요소로서, 많은 색인 기법들이 제시되었다.

이동객체의 색인기법은 크게 두 가지 형태로 분류해 볼 수 있다. 첫째는 이동객체의 현재 위치정보를 유지하고 이를 이용하여 가까운 미래에 대한 위치정보를 예측할 수 있는 색인기법이다. 이러한 색인 방법들은 실시간 위치기반 서비스를 위해서 질의를 처리하는 비용뿐만 아니라 시간에 따라 실시간으로 빈번하게 갱신되는 위치정보를 데이터베이스에 반영하기 위한 갱신비용 절감을 동시에 고려하고 있다. [1]에서 제안된 TPR-tree가 대표적인 현재위치 색인방법이다. 둘째는 과거에서 현재까지의 연속적으로 움직인 경로, 즉 이동객체의 궤적 정보를 처리하기 위한 색인기법이다. 이러한 궤적 색인방법은 누적된 대용량의 궤적 정보에서 사용자가 원하는 정보를 빨리 검색할 수 있는 방법을 중심으로 제안되었다.

본 논문에서는 두 가지의 색인 분류 중 과거 궤적 정보를 처리하기 위한 색인방법을 대상으로 한다. 기존에 제안된 많은 궤적 색인 방법들은 이동객체의 궤적 정보를 (x, y, t)의 연속된 3차원 좌표의 다각선으로 표현하고 이를 R-tree[2]를 이용하여 처리하고 있다. 예를 들어, 3D R-tree[3], HR-tree[4], 그리고 MV3R-tree[5] 는 궤적을 구성하는 최소경계사각형 정보를 이용하여 임의의 시간 점이나 구간에 대한 질의를 효율적으로 처리할 수 있는 대표적인 시공간 색인 방법들이다. 그러나 이 방법들은 과거 궤적에 대한 시간적 질의 처리에 중점을 두고 있어서 이동객체의 움직임을 해석하기 위한 궤적기반 또는 위상관계에 대한 질의 처리에는 좋은 성능을 기대하기 어렵다. 궤적 기반의 질의 처리 비용을 줄이기 위한 색인기법은 [6]에서 시공간 R-tree(STR-tree)나 궤적기반 R-tree(TB-tree)이 제안되었다. STR-tree와 TB-tree는 사장공간(dead spaces)으로 인한 성능 저하를 감소시키기 위해서 궤적 전체를 최소경계사각형으로 표현하지 않고 각 궤적 선분요소(trajectory segment)로 나누어 색인을 구성하였고 이동객체의 움직임을 해석할 수 있는 궤적의 정보를 색인 내에 유지하기 위해서 궤적 선분요소의 방향 값도 저장하였다. 특히, TB-tree는 색인을 구성할 때, 이동객체의 부분 또는 전체의 궤적에 대한 위상관계 질의나 궤적을 따라가며 처리해야 하는 복합질의를 효율적으로 처리할 수 있도록 같은 궤적에 속하는 선분요소들만 하나의 노드에 저장하고 각 노드에 저장된 이동객체의 궤적을 이중 연결리스트(double linked list)를 이용하여 전체의 궤적을 유지한다. 그 외에도 [7]에서 제안된 SETI는 시공간 및 시간 영역질의 처리뿐만 아니라 효과적인 궤적삽입을 위한 색인기법이다. SETI는 공간적인 영역을 분할하여 궤적의 선분요소를 삽입하고 시간 영역질의를 위해 각 분할영역에 시간구간 색인을 구성하여 해당 분할영역에 속하는 궤적 선분요소의 시간구간을 삽입한다. [8]에서는 궤적 선분요소의 시작시간과 끝시간을 이용하여 SETI의 성능을 좀더 향상시키는 SEB-tree를 제안하였다.

그러나, 위에서 언급한 색인 방법은 모두 유클리디안 공간에서 이동객체가 임의대로 움직일 수 있다고 전제하고 있기 때문에 움직임에 제약이 있는 이동객체를 처리하는 면에서는 효율적이지 못하다. 실제로 많은 응용분야의 이동객체들은 자신의 속한 공간에 따라 움직임에 제약들을 가지고 있다. 예를 들어, 자동차나 기차의 경우는 도로나 철도와 같이 일정한 공간 내에서만 움직일 수 있으며, 사람의 경우도 미리 정해진 영역 내에서만 움직일 수 있다. 최근에는 이동객체를 다루는 응용분야에서 도로 상에 존재하는 자동차를 관심에 두고 연구가 진행되고 있다. [9]에서는 도로 상에 존

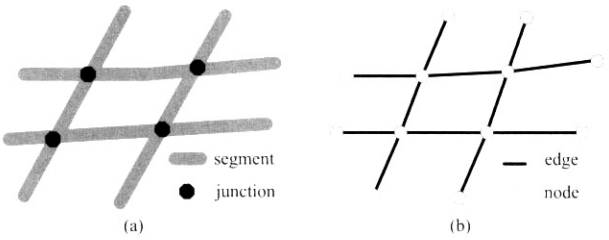
재하는 이동객체의 움직임의 모델을 제시하고 이를 위한 질의어를 정의하였고, [10]에서도 도로에 대한 2차원 모델과 그래프 모델을 기반으로 이동객체의 위치정보를 표현하고 처리하는 방법을 제안하고 있다. 그리고, [11,12]에서는 도로상의 이동객체가 가지는 움직임의 제약으로 인하여 최근접 질의가 기존의 유클리디안 공간에서의 질의 처리와 다르게 처리됨을 보여주고 있다.

특히, 색인방법에 있어서도 몇 가지 색인 방법들이 도로상에 존재하는 이동객체를 대상으로 연구되었다. Pfoser와 Jensen은 [13]에서 Hilbert 공간 채움 곡선(space-filling curve)와 두 개의 2차원 R-tree를 이용하여 도로 네트워크를 구성하는 도로 선분요소와 이동객체 궤적의 선분요소를 색인하는 방법을 제시하고 있다. 우선 도로 선분요소를 이용하여 상위의 2D-Rtree를 하나 구성하고 각 도로 선분요소에 Hilbert 값을 부여한 후 (x, y, t) 3차원 다각선 형태의 궤적을 (x, t) 2차원 형태로 차원을 줄여 나머지 2D-Rtree에 저장한다. 질의 처리는 먼저 상위의 2D-Rtree를 이용하여 해당 도로 선분요소를 찾은 후 각 도로의 Hilbert 값을 이용하여 하부의 2D-Rtree를 검색을 통해서 이루어진다. 이러한 이중적인 구조는 [14]에서 제안된 FNR-tree나 [15]에서 제안된 MON-tree에서도 나타난다. 두 색인 방법 모두 Pfoser가 제안한 색인 방법과 마찬가지로 상위에 도로 선분 또는 다각선 요소를 위한 2D-Rtree를 이용하고 하부에는 이동객체 궤적 선분을 처리하기 위해 R-tree를 사용한다. 그러나 FNR-tree는 여러 개의 1D-Rtree를 MON-tree는 여러 개의 2D-Rtree를 이용하여 상위의 2D-Rtree의 각 단말 노드의 요소들과 연결되어 이동객체 궤적 선분을 처리하는 방법이 하나의 2D-Rtree를 사용하는 Pfoser의 방법과는 다르다. 두 색인의 질의 처리 방법은 위의 방법과 비슷하게 상위 R-tree에서 질의에 해당하는 도로의 선분요소를 찾고 이후 각 도로선분에 연결된 하부 R-tree를 이용하여 궤적 선분들을 검색해 나간다. 이외에도 [16]에서는 도로 위에 움직이는 이동객체를 시공간 격자(space-time grid)공간을 이용하여 색인하는 방법을 보여주고 있다. 그러나 이러한 도로를 고려한 이동객체 색인방법들은 색인을 구성하는데 있어서는 도로의 요소를 고려하고 있지만 질의를 처리하는데 있어서는 기존의 유클리디안 공간 상의 거리를 기반으로 질의를 처리하고 있어 [17]에서 언급한 바와 같이 네트워크 공간에서는 적합하지 않은 질의 결과나 질의 처리의 비용을 증가시키는 문제를 야기시킬 수 있다. 따라서, 본 논문에서는 도로 네트워크의 특성을 이용하여 이동객체의 궤적을 표현하고 네트워크 거리를 기반으로 질의를 효율적으로 처리할 수 있는 색인기법을 제안한다.

3. 도로 네트워크 공간 상의 문제 정의

본 논문에서는 모든 이동객체가 도로 네트워크 상에 존재한다고 가정하여 도로 공간 상의 이동객체의 움직임을 도로 정보를 이용하여 표현한다. 그리고 유클리디안 거리 기반의 시공간 영역질의를 도로 네트워크 거리를 기반으로 재정의한다.

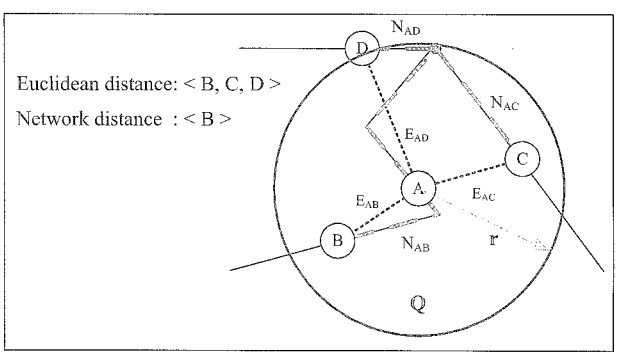
3.1 도로 네트워크



(그림 1) 도로 네트워크

도로 네트워크는 (그림 1)의 (a)과 같이 도로 선분요소(road segment)와 도로 교차점(road junction)들로 구성된다. 즉, 도로 네트워크는 공간적 요소뿐 만 아니라 각 도로 요소끼리 위상적 관계를 가지고 있다. 이러한 위상적 관계를 표현하기 위해서 (그림 1)의 (b)와 같이 그래프 구조를 이용하여 도로 선분요소는 그래프의 에지로, 도로의 교차점은 그래프의 노드로 도로 네트워크를 표현할 수 있다. 본 논문에서는 질의 처리 시 도로의 연결관계(connection relationship)를 이용하기 위해서 도로 네트워크를 그래프 형태로 나타낸다.

도로 네트워크 공간은 유클리디안 공간과는 몇 가지 다른 특성을 보인다. 첫째는 객체의 위치 좌표 표현을 다르게 할 수 있다. 유클리디안 공간 상에서 객체의 위치좌표는 (x, y) 형태로 표현하지만 도로 네트워크에서는 객체의 위치 좌표를 [18]에서처럼 도로의 정보를 이용하여 표현할 수 있다. 예를 들어, 객체가 위치한 특정 도로의 이름(name)이나 고유번호(identifier) 그리고 위치한 도로의 시작 위치에서부터의 상대적 위치(displacement)로 표현할 수 있다. 둘째는 공간의 차원이 다르다. 유클리디안 공간은 2차원 또는 3차원 이상의 공간으로 정의되는데 반하여 도로 네트워크 공간은 1차원에서 2차원 사이의 차원으로 정의된다[13, 15]. 셋째는 도로 상에 존재하는 두 객체 p와 q의 거리는 유클리디안 공간에서의 거리와 다르게 계산된다. 예를 들어, 2차원 유클리디안 공간 상의 두 객체의 거리는 두 객체의 직선거리 $L_2 = \sqrt{(p.x-q.x)^2+(p.y-q.y)^2}$ 로 계산하는데 반하여 도로 위에서의 두 객체 p와 q의 거리는 도로를 따라 움직일 수 있는 최단 경로(shortest path)의 길이로 계산이 된다. 이러한 공간 상의 다른 특징은 질의를 처리한 데 있어서 다른 결과를 도



(그림 2) 유클리디안 공간거리과 도로 네트워크 공간거리의 차이

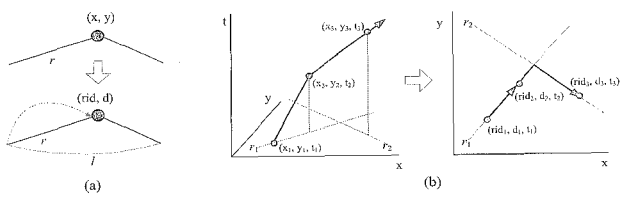
출할 수 있다. (그림 2)는 도로 상에 존재하는 이동객체들을 나타낸 것이다. 예를 들어, 주어진 질의가 "객체 A에서 거리 r 이내에 존재하는 객체들을 찾아라" 이라고 가정한다. (그림 2)에서 질의 처리시 유클리디안 공간 상의 거리를 고려한다면 객체 B, C, D가 모두 질의의 결과가 된다. 그러나, 실제 객체가 도로를 따라 움직일 수 있는 거리를 고려하게 된다면 객체 B만 질의의 결과가 된다. 이처럼 도로 상에 존재하는 이동객체를 처리하기 위해서는 도로의 연결정보가 중요하다.

3.2 도로 기반의 이동객체 궤적 표현과 시공간 영역 질의 정의

앞에서 언급한 바와 같이 본 논문에서는 이동객체의 궤적과 시공간 영역질의를 도로 네트워크 공간을 기반으로 정의한다. 우선, 색인의 대상이 되는 이동객체의 위치 좌표를 (그림 3)과 같이 (*rid, d*) 형태로 표현한다. 여기서, *rid*와 *d*는 각각 이동객체가 위치한 도로 선분요소의 고유번호와 도로의 시작점에서의 상대적 거리를 말한다. 따라서, 우리는 임의 시간 t에서의 객체의 좌표를 (*rid, d, t*)로 표현하고 시간에 따른 이동객체의 궤적은 다음과 같이 정의한다.

$$Trajectory = \{(rid, d, t)^* \mid rid \in \text{int}, d \in real(0 \le d \le l), t \in time\}$$

여기서, l은 이동객체가 위치한 rid 도로선분의 길이이다. 즉, 임의의 이동객체의 궤적은 (rid_1, d_1, t_1), (rid_2, d_2, t_2), …, (rid_n, d_n, t_n)의 순서열로 표현되어 진다. 그러나, 색인을 구성하는데 있어서 이동객체의 전체 궤적을 하나의 단말 노드에 유지하는 것은 과다한 사장공간(dead space)으로 인하여 색인의 성능 저하를 초래한다. 따라서, 본 논문에서는 다른 색인방법들과 마찬가지로 하나의 궤적을 여러 개의 $(rid_i, [d_i, d_{i+1}], [t_i, t_{i+1}])$형태의 궤적 선분요소(trajectory segment)나누어 색인을 구성하는데 사용한다. 궤적 선분요소는 각 도로 선분요소를 단위로 만들어 진다.



(그림 3) 도로 공간에서의 이동객체의 위치좌표와 궤적표현

두 번째로 도로 네트워크 공간 상에서의 시공간 영역질의를 정의한다. 궤적에 대한 시공간 영역질의는 "주어진 A 공간 안에 B 시간 동안 존재하는 궤적을 찾아라"와 같이 공간 영역 조건과 시간 영역 조건이 함께 주어진다. 따라서, 도로 네트워크 기반의 시공간 영역질의(network-based spatio-temporal range query)는 주어진 시간구간 동안 도로 상의 한 지점에서 주어진 r거리 이내에 존재하는 궤적들을 찾는 것으로 정의된다. 이 때, 주어진 질의점에서이동객체까지의 거리는 유클리디안 공간상의 거리가 아니라 도로 네트워크 상의 거리, 즉 연결된 도로를 이용하여 움직일 수 있는 최단 경로의 거리로 계산되어야 한다. 즉, 도로 상의 시공간 영역 질의를 정의하기 위해서 TS를 전체 궤적들의 집합, q를 시공간 영역질의라고 가정하자. 시공간 영역질의 q는 도로 상의 질의점을 (*rid, d*), 질의 시간 구간 *ti=[ts,te](ts<te)*, 그리고 네트워크 공간 상의 거리 r로 주어지면 다음과 같이 정의된다.

$$TS(q) = \{ts \mid \exists t \in T_I = [(q.ti \cap ts.ti) \neq \phi] \wedge ND_{q,ts}(t) \le q.r\},$$
$$\forall ts \in TS = \{ts_1, ts_2, ..., ts_m\}$$

여기서, ts는 이동객체의 궤적을 이루는 하나의 궤적선분을 말하고, T_I는 질의로 주어진 시간구간과 궤적선분의 시간구간의 교집합이고 $ND_{q,ts}$는 궤적 선분요소와 질의점 사이의 네트워크 공간거리를 말한다. 즉, 두 시간구간의 교집합에 존재하는 시간요소 t에서 질의점과 이동객체의 위치의 네트워크 상의 거리가 r보다 작을 경우 이 궤적 선분요소는 질의의 결과로 추출된다. t에서의 이동객체 위치는 궤적 선분요소에서 객체가 일정한 속도로 움직인다고 가정하여 선형보간법(linear interpolation)으로 계산된다. 그리고, 만약 질의시간과 궤적의 시간구간 사이에 교집합이 존재하지 않는 경우에는 질의점과 객체의 네트워크 거리를 계산할 수 없기 때문에 질의결과에서 제외된다.

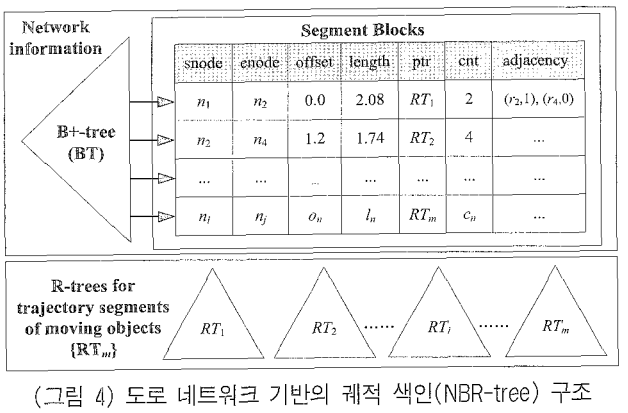
4. 도로 네트워크 기반의 궤적 색인방법

이 장에서는 도로의 경로를 따라 움직이는 이동객체의 궤적을 위하여 도로 네트워크 기반의 색인방법을 제시한다. 그리고, 이를 이용하여 도로 상의 거리를 기반으로 시공간 영역질의를 처리하는 방법을 설명한다.

4.1 자료 구조

우리는 본 연구에서 도로 네트워크 기반의 궤적 색인기법 NBR-tree(Network-based R-tree)를 제안한다. NBR-tree의 대략적인 구조는 (그림 4)와 같이 크게 2가지 부분으로 구성되어 있다. 하나는 도로 네트워크 정보를 검색하기 위한 B+-tree와 세그먼트 블록(Segment Blocks)이고 다른 하나는 이동객체의 궤적 선분요소 검색을 위한 여러 개의 R-tree들이다.

우선, B+-tree는 도로 선분요소(road segment)의 고유번호(rid)를 기반으로 만들어진다. B+-tree의 단말노드에는 각 도로 선분요소의 정보가 저장되어 있는 세그먼트 블록의 주



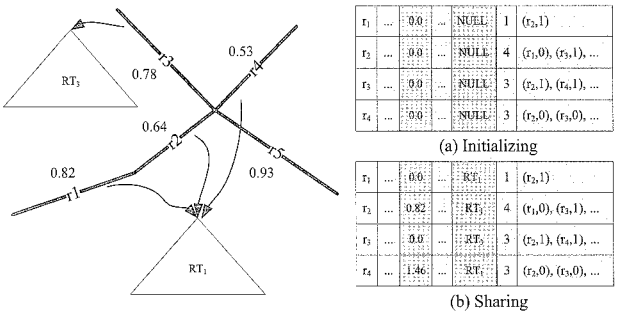
(그림 4) 도로 네트워크 기반의 궤적 색인(NBR-tree) 구조

소가 저장되어있다. 세그먼트 블록에 저장되어 있는 각 엔트리는 도로 선분요소의 양(시작/끝) 노드의 아이디(snode/enode), 시작노드의 상대위치(offset), 선분요소의 길이(length), R-tree 루트노드의 주소(ptr), 양노드에 연결된 이웃 도로 선분요소의 수(cnt), 그리고 이웃 도로선분요소들의 정보(adjacency)가 저장되어 있다. 이웃정보는 (adjEdge, type)의 리스트로 구성된다. adjEdge는 연결된 도로 선분요소의 고유번호를, type은 현재 도로 선분요소의 시작노드와 끝노드 중 어느 노드와 이웃 도로 선분요소가 연결되었는가를 말한다. type의 값이 1인 경우에는 끝노드와 연결되어 있으며 0인 경우에는 시작노드와 연결되어 있음을 말한다. 시작노드의 상대위치(offset) 정보은 초기에는 0.0으로 설정되고 다음 절에서 설명하는 도로 선분 병합 과정 거친 후 다시 계산된다. B+-tree와 세그먼트 블록의 도로 선분요소들은 [19]에서 보여진 방법과 비슷하게 클러스터링 되어 공간적 근접성이 있는 선분요소들이 같은노드 또는 블록에 저장되어 있다.

다음으로, 각 도로 선분요소에 할당된 R-tree는 시간(time)과 상대거리(displacement)의 이차원 R-tree이다. 즉, 궤적 선분요소 (*rid*, [d_1, d_2], [t_1, t_2])의 d와 t의 최소값과 최대값의 이차원 최소경계사각형(Minimum Bounding Rectangle)을 이용하여 R-tree를 구성한다. 단, 단말 노드에는 궤적 선분요소의 진행방향성을 추가한다. 진행방향성은 이동객체가 시작위치를 향하고 있는 경우는 (-)값을 끝위치를 향하고 있는 경우는 (+)값을 주어 이동객체의 움직이는 방향을 표현한다. 이러한 방향에 대한 정보는 궤적 선분의 최소경계사각형에서 임의시간에서의 이동객체 위치를 계산과정에서 사용된다.

4.2 도로 선분요소의 R-tree공유

(그림 4)의 NBR-tree 색인구조를 살펴보면 각 도로 선분요소마다 궤적 선분으로 위한 R-tree 루트의 주소를 가지고 있다. 이는 개별적으로 도로 선분요소마다 R-tree를 가질 수 있음을 의미한다. 그러나, 모든 도로 선분요소마다 R-tree를 할당하여 관리하는 것은 저장 공간 활용도나 질의 처리의 관점에서 보면 비효율적이다. 예를 들면, 만약 어떤 도로 선분요소는 지나가는 이동객체의 궤적선분이 몇 개 되지 않는다면 만들어진 R-tree는 거의 루트노드만 존재하거나 노드의 수가 몇 개 되지 않는 트리 형태가 되어 저장 공간 활용도가 떨어진다. 그리고 이러한 형태의 R-tree들을 질의 처리 시 매번 접근해 나가는 것은 질의 성능을 저하 시킬 수 있다. 따라서, 본 논문에서는 시작노드의 상대위치 정보를 이용하여 여러 도로 선분들이 하나의 R-tree를 공유할 수 있도록 하였다. 즉, 몇 개의 도로 선분요소를 지나는 이동객체 궤적 선분요소들을 같은 R-tree에서 관리함으로써, 전체 R-tree의 수를 줄일 뿐만 아니라 하나의 R-tree내에서도 어느 정도 적당한 궤적선분을 포함할 수 있도록 한다. 그러나, 서로 R-tree를 공유할 수 있는 도로 선분요소들은 서로 연결된 이웃 선분요소들로 제한하였다. 이는 도로 위에 존재하는 이동객체의 위치를 표현하기 위해서 도로의 선형 참조(linear reference)방법을 사용하기 때문이다. 따라서, 각 도로 선분요소들이 서로 연결성을 보장하는 선분요소에



(그림 5) 도로 선분요소들 사이의R-tree공유 예

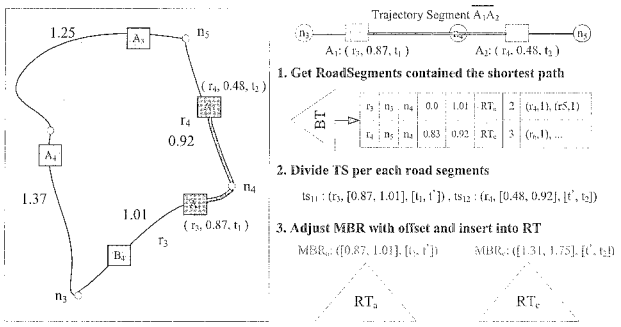
대해서만 R-tree를 공유할 수 있다. 뿐만 아니라 R-tree를 공유되는 도로 선분요소들 사이에는 사이클(cycle)을 존재하지 않는다. 이도 선형 참조를 깨는 요인이 되기 때문이다.

본 논문에서는 R-tree의 개수를 줄이기 위해서 연결개수와 이동경로 누적정보, 두 가지의 직관적인 경험적 방법을 이용하여 여러 도로 선분요소들이 하나의 R-tree를 공유하여 궤적 선분요소를 관리하도록 한다. 연결개수를 이용하는 경우에는 연결된 이웃 도로 선분의 개수가 1개만 존재하는 경우 서로 R-tree를 공유하는 방법이다. 만약 연결개수가 2개 이상이 존재하게 되면 서로 이웃된 도로 선분요소들은 독립적으로 R-tree를 가지게 된다. 그리고, 이동경로의 누적정보는 서로 여러 개의 이웃 도로 선분요소가 존재할 때, 객체가 많이 이동한 경로에 해당하는 도로 선분들끼리 R-tree를 공유하게 된다.

(그림 5)는 도로 선분요소 사이의 R-tree를 공유하는 예를 보여준다. (a)는 초기에 도로 선분요소 정보가 B+-tree를 통해 도로 선분요소 블록에 저장된 상태를 보여주고 (b)는 이동경로 누적정보를 이용하여 R-tree 공유 단계를 거친 후 도로 선분요소의 정보를 나타낸다. 각 도로 선분요소의 시작노드 상대위치와 R-tree 포인터 정보은 공유단계에서 정해진다.

4.3 궤적선분 삽입

도로 네트워크를 위한 B+-tree를 구성과 R-tree 공유 단계를 거치고 나면 이동객체의 궤적 선분요소들을 삽입한다. 앞 장에서 설명한 바와 같이 이동객체의 궤적은 도로 선분요소를 기반으로 각각의 궤적 선분요소(trajectory segment)로 나눠져서 각 해당 R-tree에 저장된다. 예를 들어, (그림 6)와 같이 이동객체의 연속된 두 좌표값 (r_3, *0.87*, t_1), (r_4, *0.48*, t_2) 이 입력된 경우, 두 좌표가 위치한 도로 선분요소의 고유번호를 비교한다. 만약 두 좌표가 위치한 도로 선분요소가 같다면 궤적 선분의 분할 과정 없이 하나의 궤적 선분요소로 도로 선분요소의 시작노드 상대위치 정보와 함께 최소경계사각형(Minimum Bounding Rectangle·MBR)을 계산한 후 해당 R-tree에 삽입된다. 그러나, (그림 6)의 경우처럼 두 좌표가 서로 다른 도로 선분요소에 위치한 경우에는 B+-tree를 통하여 도로 선분요소 r_3에서 r_4까지 최단 경로를 구성하는 도로 선분요소들의 정보들을 추출해온다. 그리고 최단 경로의 길이를 계산하여 두 좌표 사이의 네트워크 거리를 계산한 후, 이동객체가 두 지점 사이를 등속도로 움직



(그림 6) 궤적 선분요소 삽입 예

Algorithm 1: InsertTrajectorySegment(*moid*, *pos*1, *pos*2, *BT*, *RT*)

Input *moid* : identifier of moving object
*pos*1, *pos*2 : two successive positions (rid, d, t)
BT : B+-tree for road segments
RT : R-trees for trajectory segments

Begin Algorithm
IF *two positions exist the same road segment* THEN
rs = GetRoadSegment(*BT*, *pos*1.rid);
mbr = ComputeMBR(*rs*.offset, *pos*1, *pos*2); // ([d_1, d_2], [t_1, t_2])
InsertRTree(*RT*, *rs*.ptr, *moid*, *mbr*);
ELSE
rsList = GetShortestPath(*BT*, *pos*1.rid, *pos*2.rid);
nLength = ComputeTotalLength(*rsList*, *pos*1.d, *pos*2.d);
speed = *nLength* / (*pos*2.t-*pos*1.t);
tsList = SplitTrajectorySegment(*pos*1, *pos*2, *rsList*, *speed*);
FOR ts_i in *tsList* // *ts:* (rid, [d_1, d_2], [t_1, t_2])
rs = *rsList*.Get(ts_i.rid);
InsertRTree(*RT*, *rs*.ptr, *moid*, AdjustMBR(ts_i.mbr, *rs*.offset));
ENDIF
End Algorithm

(그림 7) 궤적 선분요소 삽입 알고리즘

였다는 가정하고 도로 선분요소가 바뀌는 점의 시간을 구하여 궤적선분을 분할해 나간다. 도로 선분요소에 따라 분할된 궤적선분 요소들은 최소경계사각형 정보를 재조정한 후 도로 선분요소의 R-tree 포인터 정보를 이용하여 해당 R-tree에 각각 삽입된다. (그림 7)은 궤적 선분요소 삽입 알고리즘을 도식화한 것이다.

4.4 도로 기반 시공간 영역 질의 처리

본 절에서는 제안된 NBR-tree를 이용하여 3장에서 정의된 도로 기반의 시공간 영역 질의를 어떻게 처리하는지 보여준다. (그림 8)은 도로 기반의 시공간 영역질의 처리를 위한 알고리즘을 도식화한 것이다. 도로 네트워크 거리를 기반한 시공간 영역 질의는 다음의 4단계로 이루어진다.

- **단계 1**: 주어진 질의점 (*rid, d*)에서 *rid*을 이용하여 B+-tree와 세그먼트 블록에서 해당 도로 선분요소 정보를 추출한 후 *d*를 중심으로 주어진 반지름 *r*에 포함되는 도로 선분구간을 계산한다. 계산된 선분구간은 도로선분의 고유번호, R-tree 루트노드 주소, 그리고 상대위치 정보와 함께 방문 도로 선분요소 리스트(Visited Road Segment List)에 삽입된다. 그리고 나서, 양 노드의 확장될 질의 반지름을 계산하여 각각 방문 노드 리스트(Visited Node List)에 저장하고 양 노드와 연결된 이웃 도로 선분요소 정보들을 다음 방문을 위한 큐(Queue)에 노드 정보와 함께 삽입한다.

Algorithm 2: ProcessRangeQueryOnNetworks (*query*, *radius*, *times*, *BT*, *RT*)

Input *query* : query point on a road segment (rid, d)
radius : query radius
times : query time interval [t_1, t_2]
BT : B+-tree for road segments
RT : R-trees for trajectory segments

Output *TS* : a set of trajectory segments

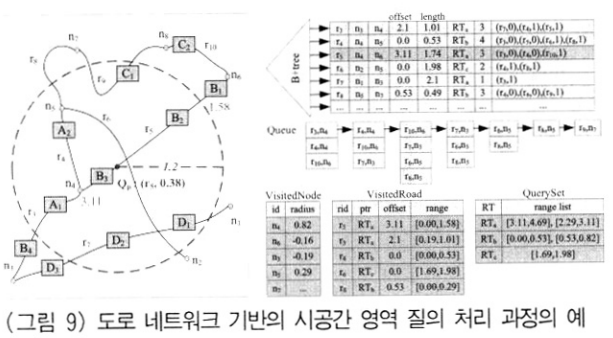
Begin Algorithm
queue = new Queue(); // for extending search spaces
VNList = new List(); // list for visited nodes
VRList = new List(); // list for visited road segments
rs = GetRoadSegment (*BT*, *query*.rid);
range = ComputeRange (*query*.d, *rs*.length, *radius*); // [d_1, d_2]
VRList.Add(*rs*.rid, *rs*.ptr, *rs*.offset, *range*);
VNList.Add (*rs*.snode, (*radius*-*query*.d));
VNList.Add (*rs*.enode, (*radius*-*rs*.length+*query*.d));
FOR *idx* = 1 to *rs*.cnt
IF *rs*.adjancy[*idx*] is adjacent with start node THEN
queue.EnQueue (*rs*.adjancy[*idx*], *rs*.snode);
ELSE *queue*.EnQueue (*rs*.adjancy[*idx*], *rs*.enode);
ENDIF
ENDFOR // step1 so far
WHILE !*queue*.Empty ()
branch = *queue*.DeQueue ();
node = *VNList*.Get (*branch*.nid);
IF *node*.radius > 0 THEN
rs = GetRoadSegment (*BT*, *branch*.rid);
IF *rs*.snode == *node*.id THEN
range = ComputeRange (0.0, *rs*.length, *node*.radius); // [0.0, d_2]
VNList.Add (*rs*.enode, *node*.radius-*rs*.length);
adjList = GetAdjacency (*rs*.adjancy, 1);
FOR each *adj* in *adjList*
queue.EnQueue (*adj*, *rs*.enode);
ENDFOR
ELSE
range = ComputeRange (*rs*.length, *rs*.length, *node*.radius); // [d_1, length]
VNList.Add (*rs*.snode, *node*.radius-*rs*.length);
adjList = GetAdjacency (*rs*.adjancy, 0);
FOR each *adj* in *adjList*
queue.EnQueue (*adj*, *rs*.snode);
ENDFOR
ENDIF
VRList.Add(*rs*.rid, *rs*.ptr, *rs*.offset, *range*);
ENDIF
ENDWHILE // step2 so far
FOR each *road* in *VRList*
querySet.Add (*road*.ptr, Adjust(*road*.range, *road*.offset));
ENDFOR // step3 so far
RETURN SearchRangeQueryRTree (*RT*, *querySet*, *times*); // step4
End Algorithm

(그림 8) 도로 기반의 시공간 영역 질의처리 알고리즘

- **단계 2**: 큐에 저장된 (도로 선분요소, 노드) 요소를 하나씩 꺼내면서 질의 반지름 범위 안에 도달할 수 있는 네트워크 상의 질의 영역을 넓이우선(Breadth-First) 방법으로 확장해 나간다. 우선, 큐에서 다음 방문할 요소의 도로 선분요소와 노드 정보를 추출한다. 노드의 고유번호를 이용하여 방문 노드 리스트에서 질의 반지름을 구하여 질의 반지름이 0보다 작은 경우는 도로 선분요소를 버리고 큐에서 다음 요소를 가져온다. 질의 반지름이 0보다 큰 경우에는 단계1과 마찬가지로 해당 도로 선분요소에서 질의 구간을 계산하여 방문 도로 선분요소 리스트에 추가시킨다. 그리고 나서, 시작노드와 끝노드 중 현재 연결된 노드가 아닌 노드만 확장될 질의반지름을 계산하여 방문 노드 리스트에 추가하고 연결된 이웃 도로 선분요소들을 큐에 삽입시킨다. 이러한 과정은 큐가 비워질 때까지 반복 수행한다.
- **단계 3**: 모든 탐색이 끝나면 방문 도로 선분요소 리스트에 저장된 질의 정보를 이용하여 질의 집합을 만들어 나간다. 이는 R-tree를 서로 공유하는 도로 선분요소들이 존재 할 수 있으므로 중복된 노드 접근을 막기 위해 같은 R-tree를 검색하는 질의들을 하나로 묶는 작업이다.

◦ **단계 4**: 모든 질의 집합이 구해지고 나면 해당 R-tree의 포인터 정보를 이용하여 R-tree를 검색해 나가면서 이동객체 궤적 선분요소들을 검색해 나간다. R-tree탐색과정은 기존의 R-tree 영역 질의처리와 동일하다.

(그림 9)는 "1P.M.에서 3P.M.사이에 질의점(r_5, 0.38)에서 1.2 km 이내에 있는 이동궤적을 검색하라"의 도로 기반 시공간 영역 질의를 처리하는 예를 보여준다. 우선, B+-tree와 도로 선분요소 블록에서 질의점이 위치한 도로 선분요소 r_5 정보를 읽어오고 r_5의 [0.0 km, 1.58 km]의 질의반지름 범위 내에 네트워크 상으로 움직일 수 있는 구간을 계산한다. 구간 정보는 r_5가 가리키고 있는 R-tree의 주소와 r_5의 시작노드인 n_4의 상대위치와 함께 방문 도로 선분요소 리스트(VisitedRoad)에 저장된다. 시작노드 n_4에서 확장될 수 있는 질의 반지름을 계산한다. 질의점의 위치가 0.38 km임으로 확장 질의 반지름 1.2 km에서 0.38 km을 뺀 0.82 km 이 n_4의 확장 질의 반지름이 된다. 구해진 n_4의 확장 질의 반지름은 노드 아이디와 함께 방문 노드 리스트(VisitedNode)에 저장하고 n_4와 연결된 이웃 연결 도로 선분요소 r_3과 r_4가 큐(Queue)에 삽입된다. 마찬가지로 끝노드 n_6의 경우도 남아있는 확장 반지름을 계산하고 연결된 이웃 도로 선분요소 정보를 큐에 삽입한다. 그러나 n_6의 경우에는 질의 반지름 1.2 km보다 멀리 존재하기 때문에 확장될 질의 반지름은 -0.16 km로 설정된다. 이렇게 질의점이 위치한 도로 선분요소에 대한 탐색이 끝나고 나면 큐에 저장된 요소들을 하나씩 꺼내서 단계2에서 설명한 것처럼 질의 영역에 포함되는 도로 선분요소들을 탐색해 나간다. 예를 들어, 큐에 서 요소 (r_3, n_4)를 제거하고 노드 n_4가 확장할 질의 반지름을 방문 노드 리스트에서 얻어온다. (그림 9)의 VisitedNode에서 n_4 질의 반지름이 0.82 km 임으로 r_3은 네트워크 상을 질의 반지름에 도달할 수 있는 영역을 포함하게 된다. 따라서, r_5의 경우처럼 질의 영역에 포함되는 구간을 계산하여 VisitedRoad에 해당 정보가 저장하고 r_3의 시작노드 n_3과 연결된 이웃 도로 선분요소들이 큐에 다시 삽입된다. 그리고 다시 (r_4, n_4)요소를 대상으로 이러한작업을 수행한다. 만약 (r_{10}, n_6)의 경우처럼 n_6의 반지름이 음의 값을 가지게 되는 경우에는 r_{10}은 그냥 큐에서 제거하고 큐에서 다음 원소를 얻어와서 탐색작업을 수행한다. 큐가 비워질 때까지 탐색작업을 반복하면 (그림 9)의 VisitedRoad와 같이 {r_3, r_4, r_5, r_6, r_8}의 5개의 도로 선분요소들이질의 영역에 포함된다. 그러나



(그림 9) 도로 네트워크 기반의 시공간 영역 질의 처리 과정의 예

기존에 제안된 색인 방법들 [12-14]은 유클리디안 기반으로 질의를 처리함으로 (그림 9)의 경우 {r_3, r_4, r_5, r_6, r_7, r_8, r_9}의 7개의 도로 선분요소들이 질의 영역에 포함되게 된다. 따라서, 본 논문에서 제안하는 네트워크 기반의 시공간 영역 질의처리 방법은 불필요한 도로 공간의 검색을 줄임으로써 효율적으로 시공간 영역 질의를 처리할 수 있다.

4.5 도로 네트워크 공간 상의 영역 질의 비용 모델

앞에서 설명한 바와 같이 도로 네트워크 상의 거리를 고려한 영역 질의는 일반적인 유클리디안 공간의 영역질의보다 검색해야 할 도로 공간을 줄일 수 있다. 이 절에서는 유클리디안 거리를 이용한 시공간 영역질의와 네트워크 거리상의 거리를 이용한 시공간 영역 질의의 처리 비용 모델을 비교해 본다. 도로 위의 임의 질의점에서 시간구간 T과 반지름 r을 가지는 시공간 영역질의 Q가 주어졌을 때, n개의 도로 선분요소로 구성된 도로 네트워크 S={s_1, s_2, …, s_n}에 대한 시공간 영역 질의 처리 비용은 다음과 같다.

$$C(S,Q) = C_{RS} + C_{RT} \qquad (1)$$

여기서, C_{RS}는 시공간 영역질의 Q안에 존재하는 도로 선분요소를 찾는 비용이고, C_{RT}는 각 도로 선분요소에 할당된 R-tree를 검색하는 비용의 합이다.

우리는 질의 비용 모델의 단순화를 위해서, 각각의 도로 선분요소 별로 R-tree를 할당하고 각 R-tree를 검색하는 비용은 같다고 가정한다. 그리고, 하나의 도로 선분요소를 찾는 비용을 A, 하나의R-tree를 검색하는 비용을 B라고 하면 식(1)은 다음과 같이 표현될 수 있다.

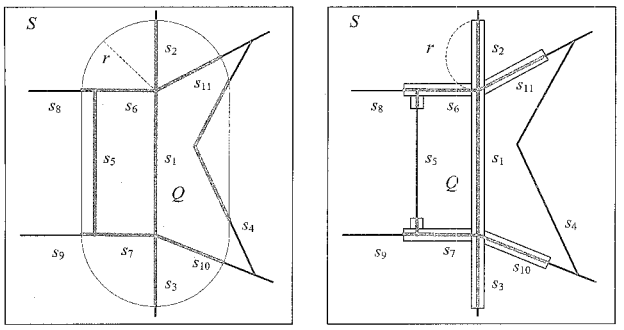
$$C(S,Q) = \sum_{i=1}^{n} P_I(s_i \cap Q \neq \phi) \cdot (A+B) \qquad (2)$$

여기서, $P_I(s_i \cap Q \neq \phi)$는 i-번째 도로 선분요소가 임의의 위치의 반지름이 r인 시공간 영역질의 Q와 교차될 확률을 말한다. 우리는 확률 P_I를 계산하기 위해서 S_i가 Q가 교차되는 사건의 지시확률변수(indicator random variable) X를 다음과 같이 정의하고 X의 기대값을 구한다.

$$X_{\text{intersect}} = I\{s \text{ intersect } Q\} = \begin{cases} 1 & : s \cap Q \neq \phi \\ 0 & : s \cap Q = \phi \end{cases}$$

$$E[X] = E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} P_I\{s_i \text{ intersect } Q\} = \sum_{i=1}^{n} \frac{L_i}{L}$$

여기서, L은 도로 네트워크를 구성하는 모든 도로 선분요소의 길이의 합이며 L_i는 질의 Q가 si 와 교차할 수 있는 도로의 길의 합이다. 그런데, L_i의 값은 질의 처리 시 유클리디안 거리를 사용하는 경우와 네트워크 상의 거리를 이용하는 경우에 따라 달라진다. 즉, (그림 10)의 (a)와 같이 유클리디안 거리를 기반으로 영역 질의를 처리하는 경우, 도로 선분요소 s가 반지름 r인 질의 영역과 교차될 수 있는 것은 질의점이 s에서 반지름 r만큼 버퍼링 된 영역 Q안에 존재하는



(a) Euclidean distance (b) Network distance

(그림 10) 임의 영역 질의에 대한 비용 비교

도로 선분요소 부분에 주어졌을 때이다. 따라서, 유클리디안 거리 기반에서 L_i는 버퍼링 영역 Q안에 존재하는 도로 부분들을 합을 구하여 계산된다. 그러나, 그래프 형태의 도로 네트워크 상의 거리를 기반으로 질의를 처리하는 경우에는 (b)와 같이 도로 선분요소 s와 연결된 이웃 도로 선분요소들을 거쳐 거리 r만큼 이동될 수 있는 경로들의 거리 합으로 L_i가 계산된다.

따라서, 우리는 식(2)를 다음과 같이 표현할 수 있다.

$$C(S,Q)=\sum_{i=1}^{n}\frac{L_i(r)}{L}\cdot(A+B)\cong n\cdot\frac{\bar{L}(r)}{L}\cdot(A+B) \qquad (3)$$

여기서, r은 질의 Q의 반지름이고 $\bar{L}(r)$은 반지름 r이 주어졌을 때 임의 도로 s가 질의와 교차될 수 있는 영역에 존재하는 평균 도로 선분요소들의 합이다. 그래프 형태의 도로 네트워크 상에서 우리는 차수(degree)를 이용하여 $\bar{L}(r)$를 계산할 수 있다. 일반적으로 그래프 이론에서 하나의 노드에 연결된 에지의 수를 노드의 차수(node degree)라 한다. 우리는 임의의 도로 선분요소의 연결된 평균 도로 선분요소의 수를 평균 에지의 차수 d라고 하고 평균 길이를 l이라 정의하면 다음의 식 (4)을 이용하여 도로 네트워크 상에서의 $\bar{L}(r)$를 계산할 수 있다.

$$\bar{L}(r)=l\cdot(1+\sum_{i=1}^{m-1}d_i)+d_m\cdot(r-ml) \qquad (4)$$

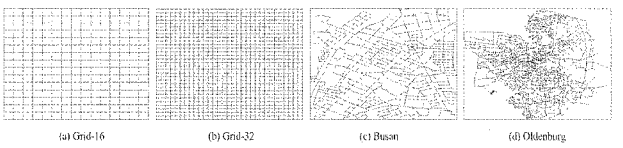
여기서, r은 시공간 영역질의의 반지름이고 m은 질의 반지름 내에서 탐색할 수 있는 깊이로, $m=\lfloor r/l \rfloor$ 이다. 그리고 d_i는 i-번째 깊이에서 평균 에지의 차수이다.

우리는 실험을 통해서 실제 유클리디안 거리 기반의 질의 처리 방법과 네트워크 거리 기반의 질의 처리 방법의 성능을 비교해본다.

5. 실험 평가

본 논문에서 제안된 네트워크 기반의 색인방법의 성능 평가를 위해 합성 및 실제 도로 데이터를 이용하여 실험을 수행하였다. 실험 결과에서는 네트워크 거리 기반의 질의 처리 방법이 유클리디안 거리 기반의 기존 방법보다 검색해야 할 도로 네트워크 공간 범위를 줄임으로써 기존의 방법들보다 좋은 성능을 보여준다.

5.1 실험 환경



(그림 11) 실험에 사용된 도로 네트워크 데이터

실험을 위해서 우리는 (그림 11)에서 보여지는 것과 같이 4가지 종류의 합성 및 실제 도로 네트워크 데이터를 사용하였다.

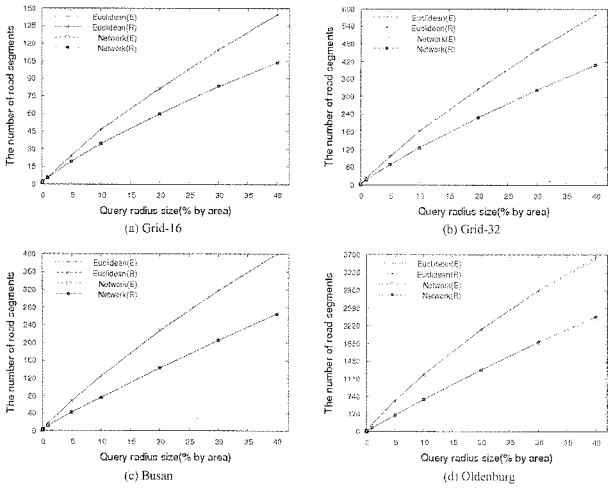
그리고 도로 네트워크를 순회하는 이동 궤적을 대량의 실제 데이터 획득의 어려움으로 인하여 Brinkhoff가 제시한 도로 기반 이동객체 생성기[20]를 이용하여 500 시간 단위 동안 대략 2,000,000개의 궤적 선분요소를 생성하였다. 생성된 궤적 선분요소는 질의처리 성능 비교를 위해서 본 논문에서 제안된 NBR-tree와 기존의 색인 방법 중 [13]에서 제안된 Hilbert-Mapping 방법과 [14]에서 보여준 MON-tree에 각각 삽입된다. 모든 색인 방법들의 노드 페이지 크기는 모두 4096 bytes로 하였으며 노드의 공간 활용도(storage utilization)는 70%로 정하였다. 시공간 영역 질의의 조건은 시간 조건은 전체 시간 범위의 25%로 일정하게 정하고 공간 영역 범위만0.01%, 0.1%, 1%, 5%, 10%, 20%, 30%, 그리고 40%로 변화시켰다. 이는 네트워크 거리 기반의 질의처리는 시간적 요인에는 영향을 끼치지 않기 때문이다. 실험에서 각 조건에 따라서 5000개의 임의 질의를 생성하여 성능평가에 사용하였다. 성능 평가에서는 각 색인 방법들에서 도로 선분요소를 검색하는 비용은 제외하고 질의 영역에 포함된 도로와 연결된 궤적 선분요소 색인을 위한 R-tree부분의 평균 노드 접근수만을 성능 비교의 고려 대상으로 하였다. 실험에 사용된 컴퓨터는 인텔 팬티엄 4 3.0GHz CPU와 2GB RAM를 가진 컴퓨터로서 윈도우(Windows XP) 운영체제를 사용하였고, 개발 언어는 자바(JAVA)로 구현하였다.

5.2 실험 결과

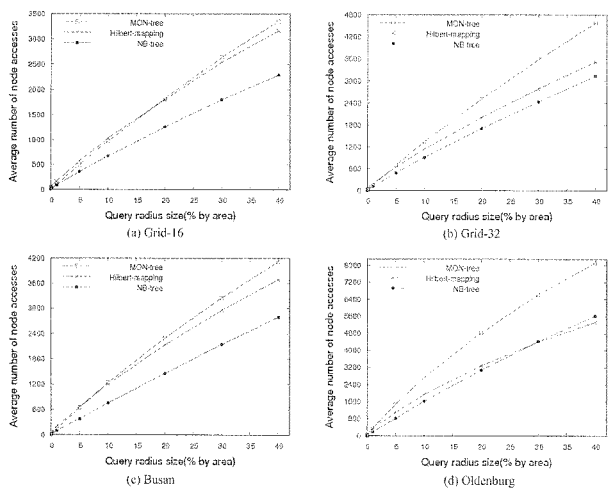
(그림 12)에서는 각 도로 네트워크 데이터에서 네트워크 거리가 유클리디안 거리 보다 어느 정도 도로의 검색공간을 줄일 수 있는지를 4.4절에서 설명한 질의 영역에 교차되는 도로 선분요소(road segment)의 성능 모델 개수와 실제 개수를 비교하여 보여준다.

실험 결과에서 알 수 있듯이 질의의 반지름이 작은 경우에는 유클리디안 거리나 네트워크상의 거리가 별 차이가 없는데 반하여 질의의 반지름이 커지는 경우에는 질의 영역에 교차되는 도로 선분요소의 개수는 현저한 차이를 보이게 된다. 대략 네트워크 거리 기반으로 영역질의를 수행 한 경우는 유클리디안 거리를 이용한 경우보다 30~40% 정도의 네트워크 검색 공간을 감소시키는 효과를 가짐을 알 수 있다. 이처럼 몇 개의 도로 선분요소가 질의에 교차되는 가에 따라 색인의 질의 처리 비용은 달라진다.

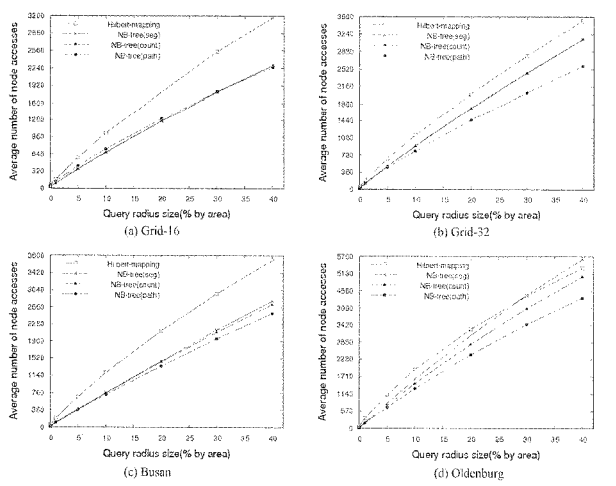
(그림 13)은 기존의 색인 방법들과 NBR-tree의 성능을 비교한 결과를 나타낸 것이다. 실험에서 MON-tree와 NB-tree는



(그림 12) 유클리디안 거리와 네트워크 상의 거리에 의한 도로 선분요소의 개수 비교



(그림 13) 유클리디안 거리와 네트워크 상의 거리 기반의 시공간 영역 질의 처리 성능 비교



(그림 14) R-tree공유방법을 이용한 시공간 영역 질의 처리 성능 비교

이동객체 궤적 선분 색인을 위해 각 도로 선분요소 별로 R-tree를 생성하였고 Hilbert-Mapping은 하나의 R-tree를 이용하였다. 실험에서는 질의 반지름이 작은 경우에는 여러 개의 R-tree를 사용하는 MON-tree나 NB-tree가 하나의 R-tree를 사용하는 Hilbert-Mapping방법보다 좋은 성능을 보이고 있다. 그러나 질의 반지름이 커짐에 따라 교차되는 도로 선분요소가 증가하고 이로 인하여 많은 R-tree를 검색함으로써 Hilbert-Mapping방법이 MON-tree보다 더 좋은 성능을 보이고, 특히 큰 도로 네트워크 데이터에서는 (그림 13)의 (d)와 같이 NB-tree보다도 좋은 성능을 보이고 있다. 하지만, 대체적으로 실험 결과에서는 NBR-tree의 네트워크 거리 기반의 질의 처리 방법이 MON-tree나 Hilbert-Mapping 방법의 유클리디안 거리 기반의 질의 처리 방법보다 좋은 성능을 보이고 있다.

(그림 14)에서는 NB-tree에서 도로 선분요소끼리 R-tree를 공유하지 않는 방법과 공유하는 방법을비교 실험한 결과를 보여준다. 4.2절에서 설명한 바와 같이 본 논문에서는 도로 연결개수와 이동경로를 이용하여 여러 도로 선분요소끼리 R-tree를 공유하여 이동객체 궤적선분을 저장하였다. 그래프에서 seg는 R-tree를 공유하지 않는 경우이고, count와 path는 각각 연결개수와 이동경로를 이용하여 R-tree를 공유한 경우를 나타낸다. 연결개수를 이용하여 R-tree를 공유하는 경우에는 합성 도로 네트워크 특성상 도로 선분요소마다 R-tree를 만드는 경우와 거의 동일한 개수의 R-tree를 사용하였지만 실제 도로 네트워크의 경우에는 40-50%정도의 R-tree 개수를 감소시킬 수 있었다. 그리고, 이동경로를 이용하여 R-tree를 공유하는 경우에는 가상 도로 네트워크에서는 원래 R-tree 개수의 10%정도를, 실제 도로 네트워크에서는 대략 15~20%정도로 R-tree개수를 감소시킬 수 있었다. 그리고 이렇게 R-tree를 공유하여 개수를 줄인 경우에는 작은 질의 반지름뿐 만 아니라 큰 질의 반지름에서도 NB-tree가 Hilbert-Mapping보다 평균 노드 접근수가 적었다. 특히 큰 네트워크 데이터에서는 대략 25%정도의 질의 처리 성능을 향상시킬 수 있었다.

6. 결론 및 향후 연구

도로 네트워크 공간은 유클리디안 공간과는 다르게 이동객체 움직임에 제약을 가진다. 본 논문에서는 도로 네트워크 공간에서 이동객체 궤적을 표현하고 이에 대한 시공간 영역 질의를 처리하기 도로 네트워크 기반의 궤적 색인 방법인 NBR-tree를 제안하였다. 제안된 색인방법은 크게 도로 네트워크를 구성하는 도로 선분요소들을 위한 B+-tree와 각 도로 선분요소를 지나는 이동객체 궤적 선분들을 저장하기 위한 여러 개 2D R-tree들로 구성되었다. 그리고, R-tree의 개수를 줄일 수 있도록 여러 도로 선분요소들이 하나의 R-tree를 공유할 수 있는 구조로 설계되었다.

이동객체 궤적은 여러 개의 도로 기반의 궤적 선분요소로 나누어지고 나누어진 궤적 선분요소들은 자신이 위치한 도로 선분요소와 연결된 2D R-tree에 시간과 상대거리의 이차원 최소경계사각형정보로 저장된다. 시공간 영역질의처리는 우선

도로 네트워크 상의 거리를 기반으로 B+-tree와 세그먼트 블록을 이용하여 질의 공간 영역에 포함되는 도로 선분요소들과 연결된 해당 R-tree의 주소에 대한 리스트를 만든다. 그리고 나서 찾아진 R-tree 주소들을 이용하여 각 R-tree에 저장된 이동객체 궤적선분들을 검색하여 질의를 처리하였다. 이러한 도로 네트워크 거리 기반의 색인 방법은 기존의 유클리디안 거리 기반의 색인 방법들보다 30~40% 정도의 네트워크 검색 공간을 감소시키고 이로 인해 질의를 효율적으로 처리할 수 있음을 실험을 통해서 보여주었다. 또한, 실험에서는 많은 R-tree의 개수로 인하여 발생하는 질의처리 성능 저하를 도로연결 개수 또는 이동경로와 같은 경험적 요소를 사용하여 줄일 수 있음을 보여주었다.

향후 연구 과제로는 색인에서 지원하는 질의 형태를 k-최근접 질의까지 확장하고 R-tree의 개수와 색인의 성능의 관계에 대한 수학적 성능 비용 모델을 마련하기 위한 연구가 필요하다. 뿐만 아니라 적절한 R-tree 개수를 결정하기 위한 요소를 결정하기 위한 연구도 함께 필요하다.

참 고 문 헌

[1] S. Saltenis, C. S. Jensen, S. T. Leutenegger, M.A. Lopez, "Indexing the Positions of Continuously Moving Objects," In SIGMOD, pp.331-342, 2000.

[2] A. Guttman, "R-trees a dynamic index structure for spatial searching," in Proc ACM SIGMOD Int Conf on anagement of Data, pp.47 - 57, 1984.

[3] Y. Theodoridis, M. Vazirgiannis, and T. Sellis, "Spatio-Temporal Indexing for Large Multi- media Applications," in Proc. IEEE Conf. On Multimedia Computing and System, pp.441 - 448, 1996.

[4] M. Nascimento and J. Silva, "Towards historical R-trees," in Proc. ACM-SAC, pp.235 - 240, 1998.

[5] Y. Tao and D. Papadias, "MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries," in Proc. VLDB, pp.431 - 440, 2001.

[6] D. Pfoser, C. S. Jensen, and Y. Theodoridis, "Novel Approaches in Query Processing for Moving Object Trajectories," in Proc. VLDB, pp.395 - 406, 2000.

[7] V. P. Chakka, A. Everspaugh, and J. M. Patel, "Indexing Large Trajectory Data Sets with SETI," in Proc. CIDR, pp.164 - 175, 2003.

[8] Z. Song and N. Roussopoulos, SEB-tree: "An approach to Index continuously moving objects," in Proc. IEEE Conf. MDM, pp.340 - 344, 2003.

[9] M. Vazirgiannis and O. Wolfson, "A Spatiotemporal Model and Language for Moving Objects on Road Networks," in Proc. SSTD, pp.25 - 35, 2001.

[10] L. Speicys, C. S. Jensen, and A. Kligys, "Computational Data Modeling for Network-Constrained Moving Objects," in Proc. ACM-GIS, pp.118 - 125, 2003.

[11] C. Shahabi, M. R. Kolahdouzan, and M. Sharifzadeh, "A Road Network Embedding Technique for K-Nearest Neighbor Search in Moving Object Databases," in Proc. ACM GIS, pp.94 - 100, 2002.

[12] C. S. Jensen, J. Kolar, T. B. Pedersen, and I. Timko, "Nearest Neighbor Queries in Road Networks," in Proc. ACM GIS, pp.1 - 8, 2003.

[13] D. Pfoser and C. S. Jensen, "Indexing of Network-Constrained Moving Objects," in Proc. ACM-GIS, pp.25 - 32, 2003.

[14] E. Frentzos, "Indexing objects moving on Fixed networks," in Proc. SSTD, pp.289 - 305, 2003.

[15] V. de Almeida and R. Guting, "Indexing the Trajectories of Moving Objects in Networks," in Proc. SSDBM, pp.115 - 118, 2004.

[16] H. D. Chon and D. Agrawal and A. E. Abbadi, "Query Processing for Moving Objects with Space-Time Grid Storage Model," in Proc. IEEE Conf. MDM, pp.121 - 130, 2002.

[17] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query Processing in Spatial Network Databases," in Proc. VLDB, pp.802 - 813, 2003.

[18] P. Scarponcini, "Generalized Model for Linear Referencing in Transportation," GeoInformatica, vol.6, no.1, pp.35 - 55, 2002.

[19] S. Shekhar and D.-R. Liu, "CCAM: A Connectivity-Clustered Access Method for Networks and Network Computations," TKDE, vol.9, no.1, pp.102 - 119, 1997.

[20] T. Brinkhoff, "A framework for generating network-based moving objects," GeoInfomatica, vol.6, no.2, pp.153 - 180, 2002.

김 경 숙

e-mail : ksookim@pusan.ac.kr

1999년 부산대학교 전자계산학과(학사)

2001년 부산대학교 전자계산학과(석사)

2001년~현재 부산대학교 대학원 전자계산학과 박사과정

관심분야 : 공간 및 시공간 데이터베이스, 지리정보시스템, Telematics 및 Ubiquitous Computing 등

이 기 준

e-mail : lik@pusan.ac.kr

1984년 서울대학교 계산통계학과(학사)

1986년 서울대학교 계산통계학과 (전자계산학 전공)(석사)

1992년 프랑스 응용과학원 전자계산학과 전산학박사

1993년~현재 부산대학교 전자전기정보컴퓨터 공학부 교수

관심분야 : 공간 및 시공간 데이터베이스, 지리정보시스템, Telematics 및 Ubiquitous Computing 등