이동 객체 색인에 대한 신기술 동향

한국과학기술원 복경수*
충북대학교 유재수**

1. 서 론

90년대 중반 이후 이동 통신 기술의 발전과 무선 통신 장비의 발전으로 사용자의 이동성에 기반한 다양한 서비스가 제공되고 있다. 특히, 사용자의 휴대용 단말기 사용이 점차 일반화되면서 무선망을 통해 사용자의 이동성을 지원할 수 있어 무선 인터넷 서비스가 점차 증가되고 있다. 또한, 이동 중인 사용자의 위치를 수 m정도의 오차 범위에 측위할 수 있는 인식 기술의 계속적인 발전으로 사용자의 단말기 위치를 활용하여 위치을 인식하고 다양한 정보를 제공받을 수 있다. 이와 함께, 사용자의 위치에 따라 다양한 서비스를 제공하기 위해 위치 기반 서비스(LBS: Location Based Service), 텔레메틱스 서비스(Telematics Service), 지능형 교통 시스템(ITS:Intelligent Transport System) 등이 각광받고 있다[1]. 일반적으로 위치 기반 서비스는 핸드폰, PDA와 같은 무선 단말기에 탑재된 GPS 장비나 이동 통신망을 통해 사용자의 위치를 파악하여 공공 안전 서비스, 위치추적 서비스, 교통안내 서비스, 정보 제공 서비스 등으로 제공한다. 텔레매틱스 서비스는 양방향 무선 네트워크와 GPS가 설치된 단말기를 통해 교통정보, 원격차량 진단, 모바일 전자상거래(M-Commerce)와 같은 각종 정보 서비스를 제공한다[2,3].

유비쿼터스 사회는 유비쿼터스 컴퓨팅과 유비쿼터스 네트워크를 기반으로 물리공간을 지능화함과 동시에 물리공간에 펼쳐진 각종 사물들을 네트워크로 연결하는 것을 의미한다. 이러한 유비쿼터스 사회는 이동성을 전제로 하고 있으며 유비쿼터스 사회가 구체화되면서 위치 인식 기술을 기반으로 이동 중인 사용자나 자동차와 같은 이동 객체(Moving Object)를 활용하기 위한 필요성이 증가되고 있다. 이동 객체는 시간의 변화에 따라 공간적인 위치 및 모양이 연속적으로 변화는 객체로 시계열 데이터(time series data) 또는 시공간 데이터(spatio-temporal data)의 특수한 형태이다. 이러한 이동 객체는 위치 변화를 서버로 전송하고 이를 통해 이동 객체의 위치를 추적할 수 있다.

이동 객체에 대한 활용이 점차 증가되면서 이동 객체를 효과적으로 저장, 관리하기 위한 이동 객체 데이터베이스(MOD: Moving Object Database)에 대한 관심이 진행되고 있다. 이동 객체 데이터베이스는 시간에 따라 연속적인 위치 변화를 갖는 객체를 효과적으로 저장하고 관리하기 위한 데이터베이스이다[4]. 전통적인 데이터베이스 시스템은 한 번 삽입이 수행되면 갱신이 자주 일어나지 않는 정적인 객체들을 저장하고 관리한다. 따라서, 이동 객체와 같이 계속적인 위치를 변화를 갖는 객체에 대해서는 많은 갱신 비용을 초래하여 성능이 현저히 저하되는 문제점이 있다[5]. 이동 객체 데이터베이스는 계속적인 위치 변화를 갖는 이동 객체를 효과적으로 갱신하고 이를 통해 다양한 서비스를 제공하기 위해 이동 객체 모델링, 이동 객체 질의 처리, 이동 객체 불확실 위치 관리, 이동 객체 색인 구조 등과 같이 다양한 분야에서 연구들이 진행되고 있다.

대용량의 이동 객체에 대한 서비스를 제공하기 위해서는 시간의 변화에 따른 공간적인 위치를 빠르게 색인하고 다양한 유형의 검색을 처리할 수 있는 색인 구조가 필수적이다[5,6]. 이를 위한 한가지 방안으로 기존에 제안된 전통적인 다차원 색인 구조에 시간 차원을 추가하여 색인을 구성할 수 있다. 그러나 전통적인 다차원 색인 구조는 색인을 구성할 객체들에 대한 동적 변화 특성을 고려하지 않고 있으며 삽입이나 갱신과 같은 색인 구조의 동적 변화보다는 검색에 초점을 맞추어 구성된다. 이동 객체는 시간의 변화에 따라 연속적인 위치 변화를 갖기 때문에 새로운 객체의 삽입은 물론 기존에 삽입된 객체에 대한 갱신이 매우 빈번히 발생한다[6]. 따라서, 이러한 이동 객체의 계속적인 위치 변화에 따른 삽입이나 갱신을 보다 효과적으로 처리하기 위한 색인 구조에 대한 연구가 필요하다[7]. 이동 객체에 대한 색인 구조는 제공되는 검색 유

* 정 회 원
** 종신회원

형에 따라 현재 위치 검색을 위한 색인 구조, 과거에서 현재까지의 위치를 검색하기 위한 색인 구조, 미래 위치를 검색하기 위한 색인 구조로 구분되어 연구되고 있다.

위치 기반 서비스, 텔레메틱스 서비스, 지능형 교통 시스템은 제공되는 서비스 환경의 차이로 서로 독립적인 서비스로 인식되어 왔다. 유비쿼터스 사회의 실현이 구체화되고 이동 통신 기술과 복합 출현으로 서비스간 차이와 경계는 점차적으로 희미해지고 있다. 이와 함께, 도로망에서 이동하는 차량을 대상으로 하는 색인 구조에 많은 연구들이 진행되고 있다. 또한, 이동 객체의 연속적인 위치 변화에 따른 갱신을 효과적으로 수행하기 위한 연구가 계속적으로 진행되고 있다.

본 고에서는 이동 객체에 대한 서비스를 제공하기 위해 최근에 연구되는 색인 기법에 대한 기술 동향에 대해 기술한다. 본 고에서는 도로망에서 이동하는 제약적 이동 객체에 대한 색인 기법과 이동 객체의 갱신을 효과적으로 수행하기 위한 색인 기법에 대해 기술한다. 도로망에서 이동하는 객체에 대한 색인 기법은 제공되는 검색 특성에 따라 궤적 색인 기법과 현재 및 미래 위치 위치 색인 기법에 대해 기술하고 이동 객체의 갱신을 효과적으로 지원하는 위한 색인 기법은 디스크 기반 색인 구조와 메모리 기반 색인 구조에 대해 기술한다.

2. 도로망을 이용한 이동 객체 색인 기법

최근 위치 기반 서비스의 확대와 더불어 텔레메틱스 기술의 발전으로 도로망에서 이동하는 자동차에 대한 위치를 추적하여 이에 적합한 서비스를 제공하기 위한 연구들이 진행되고 있다. 도로망에서 이동하는 객체들은 기존 유클리드 공간에서 사용되는 거리 함수를 이용하여 유사도를 판별할 수 없다. 이에 따라 도로망에서 이동하는 객체를 색인하기 위한 연구들이 진행되었다. 도로망에서 이동하는 객체들은 도로의 제약적인 특성에 의해 이동 범위가 결정되기 때문에 이를 이용하여 색인을 구성할 경우 갱신 비용 및 검색 성능을 향상시킬 수 있는 색인 구조를 개발할 수 있다. 본 절에서는 도로망에서 이동하는 객체들을 색인하기 위해 연구된 궤적 색인 기법과 현재 및 미래 위치 색인 기법에 대해 기술한다.

2.1 궤적 색인

유클리드 공간에서 이동 객체에 대한 궤적 색인은 객체의 갱신된 위치를 라인 세그먼트로 표현하여 저장한다. 그러나 도로망에서는 이동하는 객체들의 궤적은 도로 내에 존재하기 때문에 도로망의 특성을 이용할 경우 보다 효과적인 색인을 구성할 수 있다. 이에 따라, E. Frentzos는 도로망에서 이동하는 객체의 궤적을 색인하기 위해 2DR-트리와 1DR-트리를 결합한 FNR(Fixted Network R)-트리를 제안하였다[8]. FNR-트리는 라인 세그먼트로 구성되는 도로망을 2DR-트리로 구성하고 R-트리의 단말 노드에는 객체들이 도로망에서 존재하는 시간 구간을 색인하는 1DR-트리로 구성한다.

그림 1은 FNR-트리의 구조를 나타낸 것이다. FNR- 트리의 상위에 존재하는 2DR-트리의 중간 노드는 기존 R-트리와 유사하지만 단말 노드는 도로의 지리적 특성을 표현하고 하위에 존재하는 1DR-트리를 접근하기 위해 구조로 변경되었다. 2DR-트리의 단말 노드는 기존 R-트리와 달리 도로의 지리적 특성을 표현하기 위해 라인 세그먼트를 포함하는 MBB(Minimum Bounding Dox), MBB 내에서 라인 세그먼트의 지역적 특성을 나타내는 방향 정보, 그리고 하위에 존재하는 1DR-트리를 접근하기 위한 포인터로 구성되어 있다.

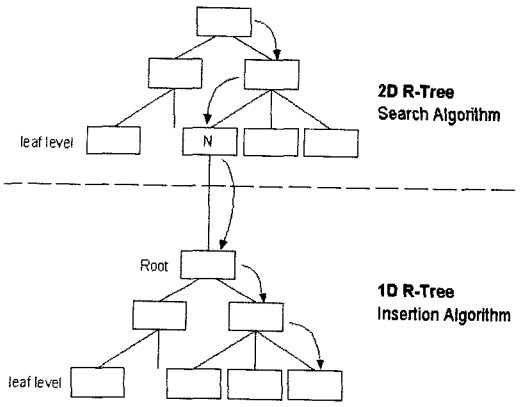
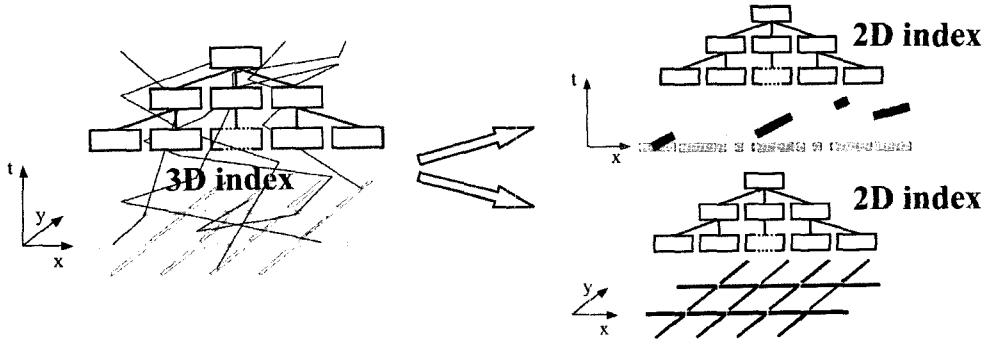


그림 1 FNR-트리 구조

FNR-트리의 하위에 존재하는 1DR-트리는 2차원의 단말 노드의 영역 MBB에 포함되는 라인 세그먼트 내에서 이동하는 객체들을 표현한다. 1DR-트리의 단말 노드는 객체의 식별자, 객체가 포함된 라인 세그먼트의 식별자, 라인 세그먼트 내에서 객체의 이동 방향, 시간 을 표현한다. 1DR-트리의 중간 노드에는 자식 노드에 포함된 객체들의 이동 시간을 색인하여 표현하고 있다. FNR-트리에서 객체 삽입은 먼저 기존 R-트리의 탐색 연산을 이용하여 2DR-트리를 탐색하고 1DR-트리에 객체의 삽입한다.

이동 객체의 궤적을 색인하기 전통적인 TB-트리나 3DR-트리 등은 객체의 이동 궤적을 색인하기 위해 객체가 이동하는 2차원 공간 정보와 1차원의 시간적인 정보를 이용하여 색인을 구성하기 때문에 3차원의 라



(a) 3차원의 궤적 색인 (b) 변환된 색인

그림 2 힐버트 값에 변환된 색인 구조

인 세그먼트에 의해 색인을 구성한다[6]. 그러나 3차원의 정보를 이용하여 색인을 구성할 경우 대량의 이동 객체들의 궤적을 표현하기 위해 색인 구조의 크기가 증가되는 문제점이 있다. 이러한 문제점을 해결하기 위해 D. Pfoser는 도로망에 존재하는 모든 에지들을 라인 세그먼트로 저장하여 2DR-트리를 구성하고 실제 도로망에서 이동 객체의 궤적을 힐버트를 이용하여 2DR-트리를 색인하는 방법을 제안하였다[9]. D. Pfoser는 이동 객체의 궤적을 2차원으로 구성하기 위해 도로망에 존재하는 2차원의 모든 에지(edge)들은 에지의 중간 값에 대한 힐버트 값으로 정렬되어 저장한다.

그림 2는 힐버트 값에 의해 표현된 이동 객체의 궤적을 색인한 예이다. D. Pfoser는 도로망에서 이동하는 객체의 궤적을 색인하기 위해 3차원의 궤적에 대해 두 개의 2차원의 색인 구조로 변환하여 저장한다. D. Pfoser는 실제 도로망을 색인하는 2차원의 색인 구조를 저장하고 힐버트 값에 의해 변환된 객체의 궤적을 2차원의 색인 구조에 저장한다. 그림 2의 (a)는 일반적인 3차원의 라인 세그먼트에 대한 궤적 색인 구조을 나타내며 (b)는 힐버트 값에 의해 변환된 색인 구조를 나타낸 것이다. D. Pfoser는 이동 객체에 대한 범위 질의를 수행하기 위해서는 먼저 도로망을 색인한 2차원의 색인 구조에서 시간 차원을 제거하고 검색을 수행하여 공간적인 위치에서 검색 조건을 만족하는 도로망의 에지들을 판별한다. 도로망의 에지를 판별하면 실제 힐버트 값에 의해 변환된 궤적을 검색한다.

V.T. Almeida는 FNR-트리에서 이동 객체를 삽입하거나 검색을 수행하기 위해 상위 2차원의 R-트리를 탐색하는 문제점을 해결하기 위한 궤적 색인 기법을 제안하였다[10]. MON-트리는 FNR-트리와 유사하게 상위 2DR-트리에는 도로망을 색인하고 하위 1DR-트리에는 도로에서 이동하는 객체들을 색인한다. MON-트리의 상위 2DR-트리는 FNR-트리와 달리 도로망을 색인하기 위해 도로망을 구성하는 에지 또는 루트(route)가 될 수 있다. 루트는 폴리라인으로 표현된 에지들의 모임이다. 그림 3은 MON-트리의 구조를 나타낸 것이다. MON-트리는 이동 객체의 궤적을 삽입할 때 상위 2DR-트리를 탐색하는 비용을 제거하기 위해 해쉬 테이블을 이용하여 이동 객체의 궤적을 삽입할 노드를 직접 접근할 수 있다. 이러한 해쉬 테이블은 FNR-트리보다 트리를 접근해야 하는 횟수를 줄일 수 있게 되어 삽입성능을 향상시킬 수 있다.

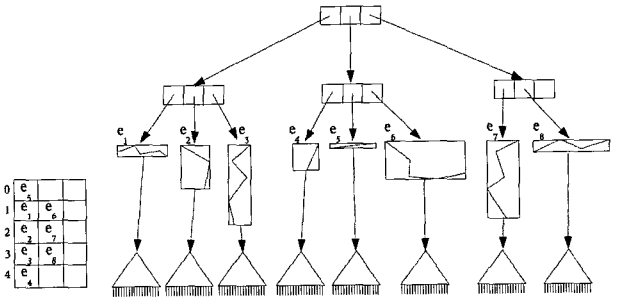


그림 3 MON-트리 구조

2.2 현재 및 미래 위치 색인

K. S. Kim은 도로망에서 이동하는 이동 객체의 갱신을 효과적으로 수행하면서 현재 위치을 색인하기 위한 IMORS(Indexing Moving Objects on Road Sectors) 기법을 제안하였다[11]. IMORS는 이동 객체의 갱신과 독립적으로 유지될 수 있는 도로망을 표현하기 위한 2차원의 R-트리와 실제 이동 객체들을 저장하기 위한 도로 섹터 블록(road sector block)으로 구성된다. 도로망을 표현하는 2차원의 R-트리는 도로 섹터에 의해 구성되며 도로 섹터는 폴리라인으로 표현된 라인 세그먼트들을 나타낸다. 2차원의 R-트리의 단말 노드는 도로 섹터와 도로망에서 이동하는 실제 객체들을 저장한 도로 섹터 블록을 접근하기 위한 포인터를 저장한다. 그림 4는 도로망에서 이동하는 객체의 현재를 색인한 IMORS의 구조를 나타낸 것이다.

그림 4에서 보는 것과 같이 데이터 블록에는 이동 객체에 대한 상세한 정보가 기록되어 있고 도로 섹터 블록에는 이동 객체의 위치만을 저장하고 이를 접근하기 위한 양방향 포인터를 유지한다.

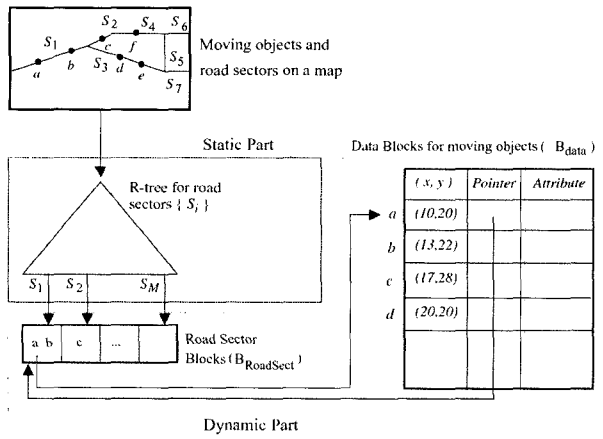
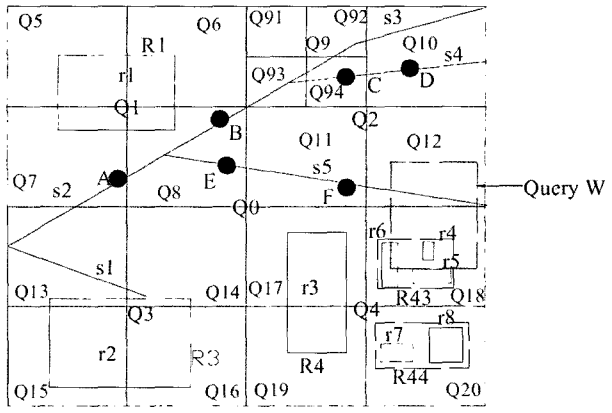


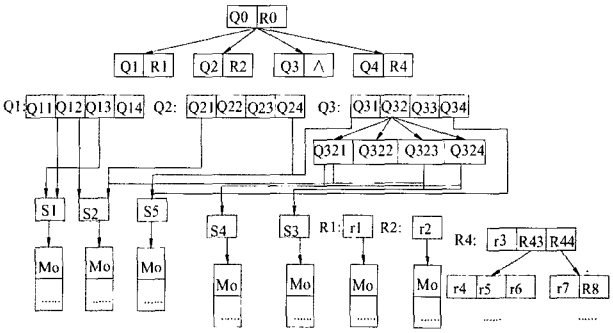
그림 4 IMORS의 구조

J. Guo는 도로망에서 이동하는 이동 객체의 현재 및 미래 위치을 색인하고 이동 객체의 위치 변화를 빠르게 갱신하기 위해 PMR 사분트리(quadtree), 사분트리, R^*-트리를 결합한 PQR-트리를 제안하였다[12]. PQR-트리는 사분트리와 R^*-트리를 이용하여 QR-트리를 생성하고 도로망에서 빠른 이동 변화를 갖는 객체들은 도로 세그먼트를 분할한 도로 섹터 내에 저장한다. QR-트리는 전체 영역에 대한 계층적인 구조를 생성하기 위한 색인 구조로 기존 사분트리를 변형하여 $\langle Q_i, R_i \rangle$로 구성되어 있다. Q_i는 전체 영역에 대해 PRM 사분트리의 분할 정책에 의해 분할된 영역을 나타내고 R_i는 분할된 각 사분트리의 영역에 대해 특정 위치에서 건물 또는 사무실과 같은 특정 영역에서 이동하는 객체들을 포함하는 영역을 나타낸다. PQR-트리에서는 PMR 사분트리의 분할 영역을 이용하여 분할된 영역 내에 임계치 이상의 도로 세그먼트가 포함되어 있을 경우 사분 트리를 분할하고 분할된 영역 내에 포함된 R_i 영역에 분할되지 않을 경우 다시 분할을 수행한다. 그림 5은 PQR-트리의 구조를 나타낸 것이다. 그림 5의 (a)와 같이 분할된 영역이 존재한다고 할 때 생성된 PQR-트리는 그림 5의 (b)와 같다. PQR-트리는 그림 5의 (b)에서 보는 것과 같이 QR-트리의 단말 노드는 도로망을 분할한 도로 섹터들을 접근하기 위한 포인터를 갖는다. 도로 섹터들은 해당 도로 섹터에서 이동하는 객체들을 동적 배열 형태로 유지한다.

R-트리 기반 색인 구조는 동일한 노드에서 서로 다른 속도와 방향으로 이동하는 객체들을 색인할 경우 MBR 영역을 계속적으로 확장시키기 위해 많은 갱신



(a) 분할 영역



(b) PQR-트리 구성 예

그림 5 PQR-트리 구조

비용을 소요할 뿐만 아니라 MBR 영역들 사이에 겹침이 발생하여 검색 성능이 저하되는 문제가 있다. 이러한 문제점을 해결하기 위해 J. Chen은 도로망에서 이동하는 객체의 갱신을 효과적으로 수행하기 위해 도로망에서 유사한 이동 변화를 갖는 객체들을 그룹으로 관리하기 위한 AU(Adaptive Unit)이라는 개념을 이용한 색인 구조를 제안하였다[13]. AU는 도로망을 나타내기 위해 사용되는 에지에 대해 동일한 에지에서 동일한 방향으로 특정 임계치 내에서 유사한 속도를 갖는 객체들을 그룹화한 것이다. J. Chen은 도로망을 색인하기 위한 2차원의 R-트리와 AU 리스트로 구성되어 있다. 그림 6는 AU에 의해 도로망에서 이동하는 객체들을 색인으로 구성한 예를 나타낸 것이다. AU 색인 기법은 도로망을 색인한 2차원 R-트리에서 AU 리스트를 직접 접근할 수 있도록 메모리에 AU의 요약 정보를 저장하는 직접 접근 테이블(Direct Access Table)을 사용한다. 2차원 R-트리의 단말 노드들은 도로 세그먼트들을 표현하기 위해 사용되는 에지와 AU를 접근하기 위한 직접 접근 테이블에 대한 포인터를 유지한다. 동일한 에지에 대해 다수의 AU들이 존재할 수 있기 때문에 질의 성능을 향상시키기 위해 동일한 에지에 대한 AU는 동일한 디스크 페이지에 기록된다.

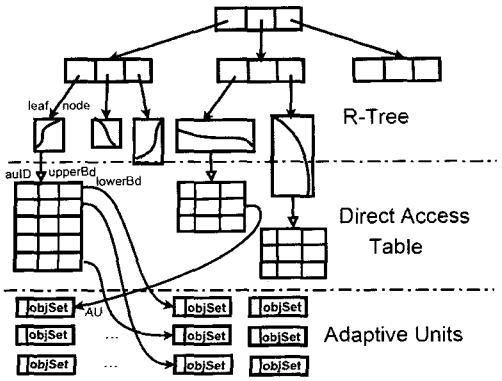


그림 6 AU에 의한 색인 구조

3. 위치 갱신에 효과적인 색인 기법

이동 객체의 현재 및 미래 위치를 검색하기 위한 색인 구조는 이동 객체의 연속적인 위치 변화에 따라 갱신 작업을 수행하기 때문에 많은 갱신 비용을 필요로 할 뿐만 아니라 이로 인해 검색 성능을 저하시킨다. 이동 객체의 현재 및 미래 위치를 검색하기 위해 R-트리 기반 색인 구조는 계속적인 삽입과 삭제 과정을 반복하면서 갱신 과정을 수행한다. 이러한 과정에서 분할된 영역들 사이에 겹침이 증가하거나 객체의 위치 변화로 인해 분할된 영역을 재조정하기 위한 작업을 수행한다. 따라서, 분할된 영역들 사이에 선별력이 저하되어 검색 성능이 저하되는 문제점이 있다. 본 절에서는 계속적인 이동 객체의 위치 변화에 따른 갱신을 효과적으로 수행하기 위해 제안된 색인 구조에 대해 기술한다.

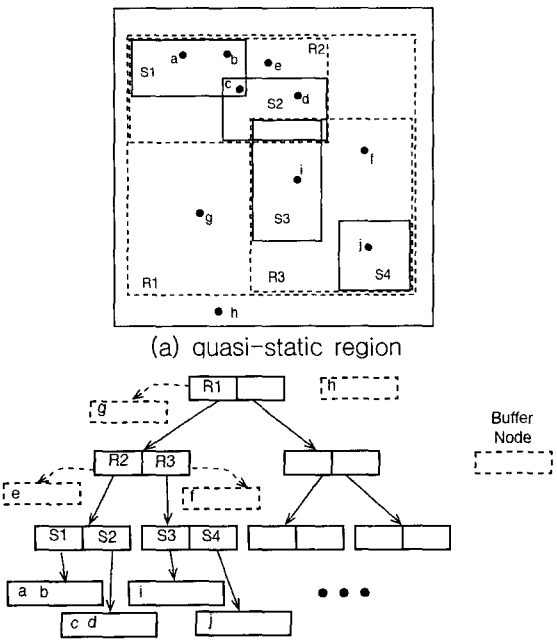
3.1 디스크 기반 색인 기법

이동 객체의 위치를 색인하기 위한 디스크 기반 색인 구조는 객체의 계속적인 위치 변화에 따른 갱신을 수행하기 위해 디스크 접근 회수를 감소를 시키는 것이 매우 중요하다. 이러한 문제점을 해결하기 위해 LUR-트리[14], 상향식 갱신(bottom up update)[7], Q+R-트리[15] 등이 제안되었다. LUR-트리와 상향식 갱신 기법은 메모리에 디스크를 직접 접근할 수 있는 보조 색인 구조를 생성하여 갱신 과정에서 색인 구조 전체를 탐색하는 비용을 감소시켰다. 이에 반해, Q+R-트리는 객체의 이동 특성에 따라 사분 트리에는 갱신을 효과적으로 처리하기 위해 빠르게 이동하는 객체를 저장하고 특정 지역에서 느리게 이동하는 객체들은 R*-트리에 저장하는 기법을 사용하였다. 최근 기존에 제안된 기법들을 보다 향상시키기 위해 많은 연구들이 진행되고 있다.

R. Cheng은 이동 객체의 위치를 색인하기 위한 기존 R-트리 기반 색인 구조의 갱신 및 질의 성능을 향상시키기 위한 CT(Change Tolerant)-R-트리를 제안하였다[16]. CT-R-트리는 R-트리를 기반으로 이동 객체의 과거 이동 패턴을 분석하여 이동 객체의 갱신이 빈번히 발생하는 영역(quasi-static region)을 생성하여 색인을 구성한다. CT-R-트리는 색인을 구성하기 위해 다음과 같은 네 단계를 수행한다.

- 이동 객체의 이동 패턴을 분석하여 quasi-static region을 판단
- quasi-static region들 사이의 관계를 표현하기 위한 update graph를 생성
- update graph를 이용하여 노드에 저장할 quasi-static region들을 합병
- 합병된 quasi-static region들을 이용하여 R-트리 기반 색인 구조를 생성

CT-R-트리는 각 중간 노드에는 노드 버퍼(node buffer)를 연결 리스트로 유지하여 객체의 삽입이나 갱신으로 객체의 위치가 quasi-static region 내에 포함되어 있지 않을 경우 객체의 위치를 포함하는 중간 노드의 노드 버퍼에 위치를 저장한다. 그림 7은 CT-R-트리에서 객체의 위치를 저장한 예를 나타낸 것이다. 그림 7에서 S_i는 quasi-static region을 나타낸 것이고 a에서 j까지는 이동 객체를 나타낸 것이다. 그림 7의 (a)에서 이동 객체 e, f, g, h는 quasi-static region에 포함되지 않는 객체들로 이러한 객체들은 그림 7의 (b)와 같이 객체의 위치를 포함하는 중간 노드를 탐색하여 노드 버퍼에 저장한다.



(a) quasi-static region

(b) CT-R-트리의 구성 예

그림 7 CT-R-트리 구조

CT-R-트리는 LUR-트리와 유사하게 이동 객체에 대한 갱신을 빠르게 처리하기 위해 객체의 현재 위치를 저장하고 있는 노드를 직접 접근할 수 있도록 해쉬 테이블을 메모리에 유지한다. 또한, 객체의 삽입으로 특정 중간 노드에서 연결 리스트로 구성된 노드 버퍼가 특정 임계치 이상의 노드로 구성된다면 CT-R-트리는 새로운 R-트리를 생성하여 생성된 R-트리를 전체를 기존 색인 구조에 벌크 삽입한다.

LUR-트리, 상향식 갱신(bottom up update), Q+R-트리, CT-R-트리 등은 기존 R-트리 기반 색인을 변형한 색인 구조이다. 그러나 이러한 색인 구조들은 객체의 삽입이나 갱신으로 전체적인 R-트리가 갖는 문제점을 해결하지 못한다. 즉, 객체의 삽입이나 갱신으로 MBR을 재조정하거나 분할로 인해 색인 구조를 재구성하는 작업을 수행해야 한다. X. Xiong과 X. Wang은 R-트리 기반 색인 구조에서 발생하는 문제점을 해결하기 위해 그리드 기반 색인 구조를 디스크 기반 색인 구조로 변형한 GTree와 LUGrid 기법을 제안하였다.

X. Wang에 의해 제안된 GTree는 객체의 삽입이나 갱신으로 인한 노드의 재조정 작업을 감소시키기 위해 그리드를 동일한 크기로 분할하여 노드에 저장하는 기법을 제안하였다[17]또한, 갱신 과정에서 접근해야 할 디스크 수를 감소시키기 위해 median-down이라는 갱신 기법을 제안하였다. GTree는 노드의 팬아웃 수만큼 동일한 크기의 그리드를 분할하여 노드에 저장하고 분할된 각 그리드는 더 이상 분할이 필요 없을 때까지 계속적으로 분할하여 자식 노드를 생성한다. GTree에서는 갱신 시간 및 노드의 재구성 시간을 감소시키기 위해 직접 연결(direct link)과 형제 연결(sibling link)이라는 개념을 사용한다. 직접 연결은 LUR-트리의 보조 색인 구조와 동일하게 갱신 과정에서 단말 노드를 직접 접근하기 위한 목적으로 사용한다. 형제 연결은 X-트리에서 슈퍼 노드와 유사한 개념으로 단말 노드에서 불필요한 분할을 방지하기 위한 목적으로 사용한다. 그림 8는 GTree의 구조를 나타낸 것이다. 그림 8에서 보는 것과 같이 GTree는 높이 균형 트리가 아니며 더 이상 그리드에 대한 분할이 필요 없을 경우 중간 노드에 실제 객체를 저장하는 단말 노드를 저장할 수 있다. 또한, 중간 노드에는 자식 노드에 대한 영역 정보를 저장할 필요없이 수학적 연산을 통해 자식 노드와의 관계를 계산할 수 있다.

GTree에서는 이동 객체의 대한 갱신을 효과적으로 처리하기 위해 median-down이라는 갱신 기법을 제안하였다. GTree는 갱신 과정에서 접근해야 할 조상

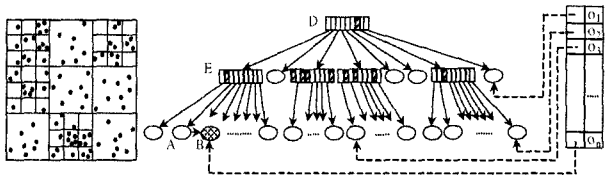


그림 8 GTree 구조

노드들을 직접 접근하기 위해 단말 노드에 조상 노드에 대한 포인터를 유지한다. 갱신을 수행하기 위해서는 객체의 기존 위치가 저장된 단말 노드와 객체의 새로운 위치를 삽입할 단말 노드의 공통적인 최하위 조상 노드(LCAN : Lowest Common Ancestor Node)를 수학적인 수식에 의해 계산하고 단말 노드에 저장된 조상 노드에 대한 포인터를 이용하여 접근할 부모 노드를 순차적으로 접근하지 않고 갱신이 필요한 LCAN을 직접 접근한다. 또한, X. Xiong은 지연 삽입과 지연 삭제를 수행하기 위해 그리드 기반 색인 구조를 이용한 이동 객체의 갱신을 효과적으로 수행하기 위한 RUGrid를 제안하였다[18]. RUGrid는 갱신 과정에서는 접근해야 할 디스크 수를 감소시키기 위해 디스크에 존재하는 하나의 노드에 해당하는 그리드를 메모리에 유지한다.

3.2 메모리 기반 색인 기법

이동 객체의 위치를 색인하기 위한 가장 일반적인 메모리 기반 색인 기법은 전체 영역을 그리드로 분할하고 해쉬 값에 의해 접근하는 해쉬 기법이다. 기존 해쉬 기법은 고정된 크기의 그리드를 이용하여 이동 객체에 대한 해쉬 값을 할당하기 때문에 그리드의 크기에 따라 갱신 성능 및 검색 성능에 많은 영향을 미친다. 즉, 그리드 크기가 증가할수록 갱신 성능은 향상되지만 검색 성능은 저하되고 그리드 크기가 감소될수록 검색 성능은 향상되지만 갱신 성능은 저하되는 문제점이 있다. 이러한 문제점을 해결하기 위해 D. Kwon은 적응적 다중 레벨 해쉬 기법(adaptive multi-level hashing)을 제안하였다[19]. 다중 레벨 해쉬 기법은 이동 객체의 특성을 고려하여 두개의 계층으로 구성된 해쉬 구조를 생성한다. 상위 해쉬는 빠르게 이동하는 객체에 대한 갱신을 효과적으로 처리하기 위해 넓은 그리드를 생성하고 하위 해쉬는 특정 지역에서 느리게 이동하는 객체에 대한 검색 성능을 처리하기 위해 좁은 그리드를 생성한다. 그림 9는 적응적 다중 레벨 해쉬 기법을 나타낸 것이다. 그림 9에서 보는 것과 같이 두 개의 계층 구조로 구성되어 있어 객체의 이동성에 따라 빠르게 이동하는 객체는 Coarse Grid에 저장하고 특정 지역에서 이동 변화가 느린 객체는 Fine

Grid에 저장된다. 적응적 다중 레벨 해쉬 기법은 객체의 이동성에 따라 상승(escalation)과 하강(deescalation)을 수행하여 Finde Grid와 Coarse Grid를 서로 이주하여 저장될 수 있다.

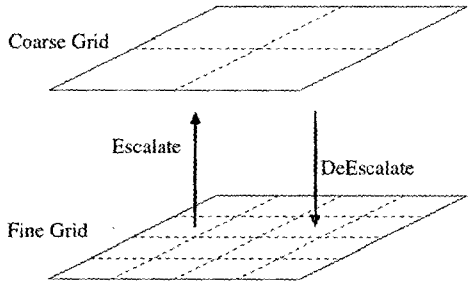


그림 9 적응적 다중 레벨 해쉬

객체의 갱신 과정에서 객체의 새로운 위치가 기존 그리드에 존재할 경우에 일반적인 해쉬 기법을 사용한다. 그러나 객체의 새로운 위치가 기존 Grid을 벗어났을 경우에는 상승 단계나 하강 단계를 수행한다. 상승 단계는 Find Grid에 존재하는 객체가 기존 그리드를 벗어났을 경우 빠르게 이동하는 객체로 판별되어 Coarse Grid에 저장하는 것을 의미한다. 이에 반해 하강 단계는 Coarse Grid에 존재하는 객체에 대해 그리드를 벗어난 객체가 실제 느리게 이동하는 객체로 판별되어 Fine Grid에 저장하는 것을 의미한다. 그러나 Coarse Grid에 존재하는 객체가 그리드를 벗어났을 때 객체가 실제 빠르게 이동하는 객체인지 느리게 이동하는 객체인지를 판별하는 것은 매우 많은 비용을 소요한다. 따라서 적응적 다중 레벨 해쉬 기법은 Coarse Grid에 존재하는 객체가 그리드를 벗어났을 경우 임시 리스트(temporary list)에 저장하고 해당 그리드에 대한 검색을 수행할 때 객체의 이동성을 판별하여 하강 단계를 수행할지 판별한다.

B. Cui는 이동 객체의 위치를 색인하기 위해 객체의 이동성에 따라 활성 객체(active object)와 비활성 객체(inactive object)로 분류하고 색인을 구성하는 IMPACT라는 기법을 제안하였다[20]. IMPACT는 상대적으로 빠른 이동을 하면서 많은 갱신 비용을 소요하는 객체들을 활성 객체라 정의하고 이러한 객체들은 메모리 기반 색인 구조에 유지하고 그렇지 않은 객체들은 디스크 기반 색인 구조에 유지된다. 활성 객체를 저장하기 위한 메모리 기반 색인 구조는 그리드 기반 해쉬 기법을 사용한다. 메모리 기반 색인 구조에서 각 그리드는 실제 객체에 대한 모든 정보를 유지하는 것이 아니라 객체가 저장된 위치에 대한 포인터만을 유지한다. 디스크 기반 색인 구조는 TPR*-트리로 구성되며 메모리에는 OLRU 기반의 버퍼을 유지하여 디스크에 색인 구조를 직접 접근할 수 있도록 한다. 그림 10은 IMPACT의 구조를 나타낸 것이다. 메모리는 활성 객체를 유지하기 위한 메모기 기반 색인 구조와 자주 접근되는 디스크 기반 색인 구조의 노드를 저장하기 위한 버퍼를 유지한다.

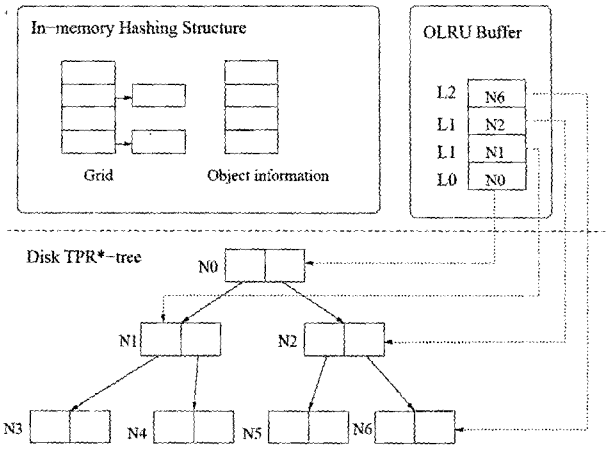


그림 10 IMPACT 구조

IMPACT에서는 갱신된 객체의 이동 속도에 따라 메모리 기반 색인 구조와 디스크 기반 색인 구조를 이주하면서 저장된다. 일반적으로 메모리에는 속도 임계치 이상이되는 활성 객체들을 유지한다. 메모리에 객체를 저장할 여유 공간이 있을 경우에는 비활성 객체를 유지할 수 있다. 그러나 이러한 비활성 객체는 메모리에 여유 공간이 없을 경우 디스크에 저장되어야 한다. 이러한 처리를 효과적으로 수행하기 위해 IMPACT는 메모리에 유지되는 비활성 객체의 식별자를 큐로 관리한다. 메모리에 여유 공간이 없을 때 디스크에 저장된 비활성 객체의 속도 임계치 이상으로 이동하면서 큐에 저장된 비활성 객체를 디스크에 기록한다.

4. 결 론

유비쿼터스 사회에 대한 실현이 구체화되면서 객체의 이동성에 기반한 서비스들에 관심이 집중되고 있다. 이와 함께, 이동하는 객체의 위치를 계속적으로 추적하여 이에 대한 서비스를 제공하기 위한 이동 객체에 대한 연구가 매우 활발하게 진행되고 있다. 본 고에서는 대용량의 이동 객체에 대한 서비스를 제공하기 위해 필요한 색인 기법의 기술 동향을 도로망에서 이동 객체 색인 기법과 이동 객체의 갱신을 효과적으로 수행하기 위한 색인 기법 관점에서 기술하였다. 현재까지 이동 객체를 색인하기 위한 많은 연구들이 진행되어 왔지만 아직까지 명확한 해법이 제시되지 못하고 있는

상황이다. 향후 서비스가 확대될 뿐만 아니라 서로 다른 서비스들의 융합으로 인해 이동 객체에 대한 색인 기법의 연구들은 계속적으로 진행되어야 하며 이와 함께 색인 구조를 통해 질의 처리 기법에 대한 연구가 계속적으로 진행되어야 할 것이다.

참고문헌

[1] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang, "Moving Objects Databases : Issues and Solutions", Proc. International Conference on Scientific and Statistical Database Management, pp.111-122, 1998.

[2] 이성호, 민경욱, 김재철, 김주완, 박종형, "위치기반서비스 기술 동향", 전자통신동향분석 제30권 제3호, pp.33-42, 2005.

[3] 안선일, 장병준, 이윤덕, "텔레메틱스 동향 및 기술개발 방향", 한국정보과학회지 제23권 제2호, pp77-82, 2005.

[4] G. Trajcevski, O. Wolfson, B. Xu, and P. Nelson, "Real-Time Traffic Updates in Moving Objects Databases", Proc. the 13th International Workshop on Database and Expert Systems Applications, pp.698-704, 2002.

[5] B. C. Ooi, K. L. Tan, and C. Yu, "Frequent Update and Efficient Retrieval: an Oxymoron on Moving Object Indexes?", Proc. International Conference on Web Information Systems Engineering Workshops, pp.3-12, 2002.

[6] M. F. Mokbel, T. M. Ghanem, and W. G. Aref, "Spatio-Temporal Access Methods", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, Vol.26, No.2, pp.40-49, 2003.

[7] M. L. Lee, W. Hsu, C. S. Jensen, B. Cui, and K. L. Teo, "Supporting Frequent Updates in R-Trees : A Bottom-Up Approach", Proc. International Conference on Very Large Data Bases, pp.608-619, 2003.

[8] E. Frentzos, "Indexing Objects Moving on Fixed Networks", Proc. International Symposium on Spatial and Temporal Databases, pp.289-305, 2003.

[9] D. Pfoser and C. S. Jensen, "Indexing of Network Constrained Moving Objects, Proc. ACM International Symposium on Advances in Geographic Information Systems, pp.25-32, 2003.

[10] V. T. Almeida and R. H. Guting, "Indexing the Trajectories of Moving Objects in Networks", GeoInformatica, Vol.9, No.1, pp.33-60, 2005.

[11] K. S. Kim, S. W. Kim, T. W. Kim, and K. J. Li, "Fast Indexing and Updating Method for Moving Objects on Road Networks", Proc. International Conference on Web Information Systems Engineering Workshops, pp.34-42, 2003.

[12] J. Guo, W. Guo and D. Zhou, "Indexing of Constrained Moving Objects for Current and Near Future Positions in GIS", Proc. International Multi-Symposium of Computer and Computational Sciences, Vol.2, pp.504-509, 2006.

[13] J. Chen, X. Meng, Y. Guo, and Z. Xiao, "Update-efficient Indexing of Moving Objects in Road Networks", Proc. Workshop on Spatio-Temporal Database Management, 2006.

[14] D. Kwon, S. Lee, and S. Lee, "Indexing the Current Positions of Moving Objects Using the Lazy Update R-tree", Proc. International Conference on Mobile Data Management, pp.113-120, 2002.

[15] Y. Xia and S. Prabhakar, "Q+Rtree : Efficient Indexing for Moving Object Database," Proc. International Conference on Database Systems for Advanced Applications, pp.175-182, 2003.

[16] R. Cheng, Y. Xia, S. Prabhakar, and R. Shah, "Change Tolerant Indexing for Constantly Evolving Data", Proc. International Conference on Data Engineering, pp.391-402, 2005.

[17] X. Wang, Q. Zhang, and W. Sun, "GTree : An Efficient Grid-Based Index for Moving Objects", Proc. International Conference on Database Systems for Advanced Applications, pp.914-919, 2005.

[18] X. Xiong, M. F. Mokbel, and W. G. Aref, "LUGrid: Update-tolerant Grid-based Indexing for Moving Objects", Proc. IEEE International Conference of Mobile Data Management, pp.13, 2006.
[19] D. Kwon, Sangjun Lee, W. Choi, and S. Lee, "Adaptive Multi-level Hashing for Moving Objects", Proc. International Conference on Database Systems for Advanced Applications, pp.920-925, 2005.
[20] B. Cui, D. Lin, and K. L. Tan, "Towards Optimal Utilization of Main Memory for Moving Object Indexing", Proc. International Conference on Database Systems for Advanced Applications, pp.600-611, 2005.

복 경 수

1998 충북대학교 수학과(이학사)
2000 충북대학교 정보통신공학과(공학석사)
2005 충북대학교 정보통신공학과(공학박사)
2005. 3~현재 한국과학기술원 전산학과 Postdoc
관심분야 : 자료 저장 시스템, 이동 객체 데이터베이스, 시공간 색인 구조, 센서 네트워크 및 RFID
E-mail : ksbok@dbserver.kaist.ac.kr

유 재 수

1989 전북대학교 컴퓨터공학과(공학사)
1991 한국과학기술원 전산학과(공학석사)
1995 한국과학기술원 전산학과(공학박사)
1995~1996. 8 목포대학교 전산통계학과 전임강사
1996. 8~현재 충북대학교 전기전자컴퓨터공학부 교수
관심분야 : 데이터베이스 시스템, 정보검색, 멀티미디어 데이터베이스, 분산 객체 컴퓨팅 등
E-mail : yjs@chungbuk.ac.kr