이동 객체 데이타베이스에서 KDB-tree의 동적 분할 정책

(The Dynamic Split Policy of the KDB-Tree in Moving Objects Databases)

임 덕 성 † 이 창 헌 †† 홍 봉 희 †††
(Duksung Lim) (Changheun Lee) (Bonghee Hong)

요 약 시간의 흐름에 따라 누적되는 대용량의 과거 위치를 관리하는 이동 객체 데이타베이스에서 이동 객체의 과거 위치를 효율적으로 검색하기 위해서는 이동 객체의 특성을 고려한 색인 구조가 필요하다. 그러나, 영역 질의 성능이 우수한 다차원 색인인 KDB-tree를 이동 객체 데이타베이스에 적용할 경우 시간 도메인이 증가하는 이동 객체 데이타베이스의 특성으로 인해 공간 도메인 가중 분할이 발생한다. 공간 도메인 가중 분할은 하나의 노드가 차지하는 MBR의 공간 영역이 분할 횟수에 반비례하게 감소되어 시공간 영역 질의 처리시 색인의 검색 비용을 증가시키는 문제가 있다.

이 논문에서는 이동 객체 데이타베이스에서 시공간 영역 질의를 효율적으로 처리하기 위한 KDB-tree의 동적 분할 정책을 제안한다. 동적 분할 정책은 공간 우선 분할 방법을 적용하는 분할 도메인 선정 방법과 포인터 페이지에서 공간 활용도를 최대화 시킬 수 있는 최근 시간 분할 정책, 영역 페이지에서 적용되는 최후 시간 분할 정책으로 구성된다. 제안한 동적 분할 정책의 성능을 평가하기 위해 3DR-tree, MV3R-tree, KDB-tree와의 성능을 비교한다. 영역 질의를 위한 성능 평가에서 동적 분할 정책을 적용한 MKDB-tree는 기존 색인에 비해 평균 30%이상의 노드 접근 회수를 감소시킨다.

키워드 : KDB-tree, 이동 객체 데이타베이스, 이동 객체 색인, 동적 분할

Abstract Moving object databases manage a large amount of past location data which are accumulated as the time goes. To retrieve fast the past location of moving objects, we need index structures which consider features of moving objects. The KDB-tree has a good performance in processing range queries. Although we use the KDB-tree as an index structure for moving object databases, there has an over-split problem in the spatial domain since the feature of moving object databases is to increase the time domain. Because the over-split problem reduces spatial regions in the MBR of nodes inverse proportion to the number of splits, there has a problem that the cost for processing spatial-temporal range queries is increased.

In this paper, we propose the dynamic split strategy of the KDB-tree to process efficiently the spatial-temporal range queries. The dynamic split strategy uses the space priority splitting method for choosing the split domain, the recent time splitting policy for splitting a point page to maximize the space utilization, and the last division policy for splitting a region page. We compare the performance of proposed dynamic split strategy with the 3DR-tree, the MV3R-tree, and the KDB-tree. In our performance study for range queries, the number of node access in the MKDB-tree is average 30% less than compared index structures.

Key words : KDB-tree, Moving Objects Databases, Moving Objects Index, Dynamic Split

· 본 연구는 교육인적자원부 지방연구중심대학육성사업(차세대물류IT기술연구사업단)에서 지원함

† 정 회 원 : 영진전문대학 컴퓨터정보계열 교수
junsung@yjc.ac.kr
†† 학생회원 : 부산정보과학고등학교 교사
angel71@pusan.ac.kr
††† 종신회원 : 부산대학교 공과대학 컴퓨터공학과 교수
bhhong@pusan.ac.kr

논문접수 : 2003년 4월 17일
심사완료 : 2006년 5월 9일

1. 서 론

최근 무선 이동통신 기술의 발달로 휴대폰과 PDA (Personal Digital Assistants)등의 무선 이동기기(이동전화, PDA등)가 보편화되면서 GPS(Global Positioning System)기술을 사용한 시공간에서의 위치 기반 서비스

(LBS : Location Based Service)의 요구가 증대되고 있다. 휴대폰, PDA를 소지한 개인이나 차량이 GPS수신기를 사용하여 일정 시간 간격으로 그들의 위치를 서버에 전송하는 것을 가정할 때, GPS수신기를 소지한 이동 물체를 이동 객체라고 정의한다. 이동 객체는 자신의 위치정보를 비 주기적인 간격으로 서버에 전송하고, 시간에 따라 지속적으로 이동하는 특성이 있다. 변경된 위치데이타를 보고하는 다수의 이동 객체들이 전송한 방대한 양의 데이타들을 저장하고 효율적으로 검색하기 위한 많은 이동 객체 색인 방법들이 연구되고 있다[1,2].

이동 객체의 대표적인 질의로서 특정 시간이나 특정 지역 안에 있는 이동 객체를 검색하기 위한 시공간 영역 질의가 있다. 시공간 영역 질의를 처리하기 위해 이 논문에서는 이동 객체의 위치데이타를 시공간 상에서의 점으로 모델링하고, 두 점사이의 이동 객체 검색은 연속된 두 점 사이의 선형 보간법을 이용하여 색인 하는 방법을 기반으로 한다[2,3]. 점으로 모델링된 데이타를 효율적으로 검색하기 위해서는 SAM(Spatial Access Method)보다는 PAM(Point Access Method)을 사용하여 색인 하는 방법이 효율적이다[4,5].

대표적인 PAM 에는 Grid File, BANG File, KDB-tree등이 있고 그 중에서 KDB-tree는 데이타의 분포를 고려하여 분할하고, 간단한 스키마를 가지고 있어서 대용량 데이타베이스 설계를 위한 다차원 공간색인에 성능이 좋은 것으로 연구된바 있다[6]. 그러나 KDB-tree는 이동 객체가 시공간에서 계속적으로 위치를 변경하며 시간 도메인의 값이 증가한다는 특성을 반영하고 있지 않다. 따라서 이동 객체의 계속적인 위치 보고로 인해 오버플로우가 발생했을 때 이동 객체의 위치데이타 검색 성능이 저하되는 다음과 같은 3가지의 문제점이 있다.

첫째, KDB-tree는 분할 도메인 선정방법으로 순환적 분할 도메인 선정 방법을 사용하는데 이 방법은 이동 객체가 이동하는 3차원 시공간이 공간 도메인과 시간 도메인으로 구성되어 있으므로 공간 도메인으로 가중 분할하는 문제를 발생시킨다. 즉, 고정된 공간 도메인에 대해 분할이 누적 될 경우 하나의 노드가 차지하는 공간 영역의 크기가 분할 수에 반비례하게 감소된다. 그 결과 시공간 영역 질의 처리시 색인의 검색 비용을 증가시키는 문제가 있다.

둘째, KDB-tree의 포인트페이지 분할 방식인 균등 분할 방식에서 시간 도메인으로 분할이 될 경우, 공간활용도를 저하시키는 문제를 발생시킨다. 시공간에서 균등 분할 방식은 공간 도메인으로 분할 할 때는 적합하지만, 시간 도메인으로 분할 할 때는 분할 후 과거 영역이 되는 포인트페이지의 공간활용도가 저하되는 문제를 발생시킨다. 그림 1에서 포인트페이지 P1과 P2는 공간 도메인으로 균등 분할을 하고, 포인트페이지 P3과 P4는 시간 도메인으로 균등 분할을 하게 된다. 포인트페이지 P3의 경우, 분할 이후에 과거 영역이 되므로 추가 삽입이 발생하지 않는 이동 객체 데이타의 특성에도 불구하고, 균등 분할을 하게 됨으로써 포인트페이지 P3의 공간활용도가 떨어지게 된다.

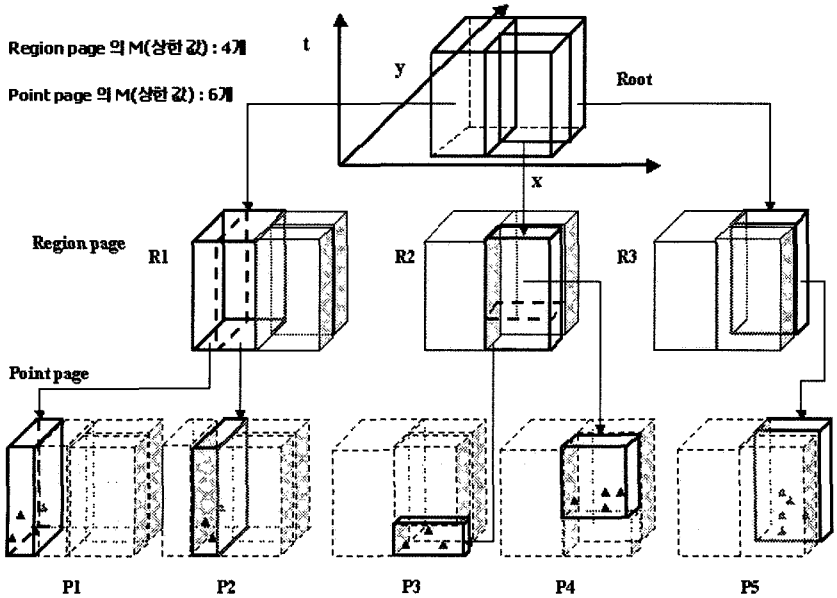


그림 1 3차원 KDB-tree의 예

마지막으로, KDB-tree의 영역페이지 분할 정책에 있어서 시공간의 이동 객체를 색인하기 위해서는 새로운 영역페이지 분할 정책이 필요하다. 기존의 FS(Forced Splitting)분할의 연쇄적 분할 전파 문제와 심각한 공간 활용도 저하 문제를 해결한 FD(First Division) 분할 정책[6]의 경우 시간 도메인으로의 분할 시 포인트 페이지 분할의 문제와 같이 영역페이지의 공간활용도를 저하시키고 색인의 깊이를 증가 시키게 된다. 따라서 시공간에서의 이동 객체 데이타의 특성을 고려한 KDB-tree의 분할 정책이 필요하다.

이 논문에서는 이동 객체의 과거 위치 검색을 효율적으로 처리하기 위해 시공간에서 이동 객체의 각 도메인 특성을 고려한 분할 방법을 제안한다. 이를 위해 분할 도메인 선정 방법을 순환적 방법, 시간 도메인 가중 분할 방법, 긴 도메인 분할 방법, 공간 우선 분할 방법으로 분류하고, 시공간에서 이동 객체 색인에 적합한 공간 우선 분할 기법과 최근 시간 분할 기법을 제안한다.

이 논문의 구성은 다음과 같다. 2장에서는 관련연구를 기술하고, 3장에서는 이 논문에서 적용하는 이동 객체 모델과 질의 모델을 기술한다. 4장에서는 이동 객체의 특성으로 인한 시공간에서 발생하는 KDB-tree의 분할 문제점을 설명한다. 5장에서는 시간 도메인의 특성을 고려한 동적 분할 정책을 제안하고, 6장에서는 동적 분할 정책을 적용한 MKDB-tree와 기존 색인과의 성능을 비교한다. 마지막으로 7장에서는 결론 및 향후 연구에 대해서 기술한다.

2. 관련 연구

이동 객체 데이타베이스에서 이동 객체의 위치 검색을 위한 색인 구조는 다차원 색인 구조를 기반으로 한다. 다차원 색인은 크게 PAM과 SAM으로 구분된다. PAM 방식은 점으로 구성된 데이타베이스에서 검색을 효율적으로 처리하기 위해 연구되어 왔다. 점 데이타는 각 차원(1차원, 2차원, …)상에서 공간적 영역을 점유하지 않는 한 점으로 표현이 가능한 반면 SAM 방식은 공간적 영역을 점유하는 데이타(예를 들어 2차원에서 사각형, 3차원에서 구)를 효율적으로 검색하기 위해 사용되는 데이타 접근 방식이다[4]. 이 논문에서는 이동 객체의 과거 위치를 점으로 모델링하여 점으로 구성된 데이타베이스를 기반으로 하기 때문에 PAM방식의 색인 방법을 사용한다.

KDB-tree[7]는 KD-tree와 B-tree의 특성을 결합한 것으로 노드들의 계층이 균형을 이룬 색인이다. 단말 노드에 점(Point)을 저장하기 때문에 노드당 엔트리의 개수가 MBR을 저장하는 색인들보다 많아서 생성되는 색인의 크기가 작고, 검색이 빠르다는 장점을 가지고 있다. 그러나 KDB-tree의 분할 문제에 대한 연구[8]에서 영역페이지 분할 정책으로서 강제 분할 정책을 사용하기 때문에 연쇄적인 분할 전파를 야기하고, 포인트페이지의 공간활용도를 저하시키는 문제를 지적했다. 이 문제는 다른 공간 분할 구조의 색인에서도 동일한 문제를 발생시키는데 Buddy-tree와 LSD-tree에서는 이와 같은 강제적이고 연쇄적인 분할 전파를 발생시키는 방법과 달리 영역페이지가 처음 분할되었던 분할 요소에 의해서 분할되는 분할 정책을 사용한다[9,10]. [6]에서는 KDB-tree의 영역페이지 분할 정책이 강제적으로 하위 영역에 전파된다고 하여 FS분할(Forced Splitting)과 영역페이지가 처음으로 분할되었던 분할 축에 의해 영역페이지가 분할 하는 방식을 FD분할(First Division Splitting)이라 정의하고, 두 분할 정책에 관한 성능 비교를 하였다. 성능 비교의 결과 FD분할이 강제로 영역페이지의 하위 영역까지 분할이 전파되는 문제가 발생하는 FS분할 보다 포인트페이지의 공간활용도를 높이고, 검색 성능도 향상 시킴을 보였다.

KDB-tree를 이동 객체 DB에 적용한 연구로서 [2]에서는 이동 객체의 보고 주기를 일정하게 함으로써 연속적인 정보를 유지 가능하게 하는 모델과, 이동 객체 색인을 위한 현재 질의와 미래 질의 처리 방법에 관해 연구를 하였다. [3]에서는 단말 노드에 MBR을 저장함으로써 노드 엔트리 수가 적은 문제를 해결하기 위해 Line객체를 점으로 모델링해서 저장하는 방법과 영역질의 시 질의 영역 내에 있으나 검색되지 않는 문제를 해결하기 위해 버퍼(Buffer)질의 방법을 제안하였다.

MV3R-tree[5]는 MVB-tree (multi-version B-tree)의 개념을 이용한 것으로, MVR-tree와 단말 노드들로 구성되는 보조적인 3D R-tree를 결합한 이중 구조를 가진다. MVR-tree는 타임스탬프 질의를 처리할 때 사용되고, 3D R-tree는 영역 질의를 처리할 때 사용된다. MV3R-tree는 타임스탬프 질의와 영역 질의를 모두 효율적으로 다룰 수 있지만, 3D R-tree에 비해 영역 질의 성능 향상은 없지만, MV3R-tree는 HR-tree에 비해 저장공간을 적게 필요로 하는 장점을 가진다.

이동 객체의 생성에 대한 연구로써 GSTD(Generate Spatio-Temporal Data)알고리즘[11]은 정의된 분포에 따라 이동 객체의 위치 보고 간격, 중점 이동 간격, 크기 변경 간격을 달리하여 이동 객체 데이타 집합을 생성한다. 네트웍 기반 이동 객체 생성 알고리즘은 네트웍 데이타를 기반으로 이동 객체의 최대속도, 출발지점, 도착지점, 네트웍 상의 최대 이동 객체 개수, 라우팅 정보들을 인자로 설정하여 이동 객체 데이타 집합을 생성한다.

3. 이동 객체 모델 및 시공간 질의 모델

이동 객체 데이타베이스에서 이동 객체들은 주기적으로 위치정보와 응용에 따른 추가 정보를 보고하고, 전송된 정보는 데이타베이스에 저장된다. 이 논문에서 이동 객체의 위치는 보고시점 t에서 시공간 위치정보(x,y,t)로서 모델링된다. 이동 객체는 일정 시간 간격(주기 T)내에 평균적으로 주기당 보고 횟수($T_{sampling}$)만큼의 위치를 보고한다. 이와 같은 이동 객체 모델링에서 이동 객체의 과거 위치를 시공간의 한 점으로 저장하기 때문에 점과 점사이의 이동 경로의 표현은 선형 보간법에 의해 표현된다고 가정한다.

이동 객체의 과거 위치를 검색하기 위한 색인 구조는 시공간 위치 정보를 이용하여 색인을 구성한다. 색인을 구성하기 위한 이동 객체 데이타는 시공간 위치정보와 데이타의 저장 위치를 가리키는 레코드 ID(point id)로 구성된다.

이동 객체의 과거 위치 검색을 위한 대표적인 질의로서 시공간 영역 질의가 있다. 시공간 영역 질의는 "시간 간격 I동안 R 영역 내의 이동 객체를 검색하라"와 같이 시간 간격(Time Interval)과 공간 영역(Spatial Region)이 혼합된 질의 모델이다. 이와 같은 질의 모델에서 연속된 두 점 사이의 질의가 올 경우 검색이 불가능한 경우가 있다.

그림 2는 특정 시각의 이동 객체 a, b, c, d의 위치와 연속된 다음 위치의 두 점과 질의 영역 R을 xy평면에 투영하여 나타내고 있다. 이동 객체 b의 경우 점이 질의 영역 R에 포함되므로 검색된다. 그리고, 이동 객체 a와 d의 경우 검색 대상이 되지 않는다. 그러나 이동 객체 c의 이동 경로는 질의 영역 R에 포함되지만 두 점은 포함되지 않는다. 따라서, 질의 결과의 일관성을 위하여 이동 객체 c의 검색 방법이 필요하다. 이 논문에서는 이동 객체의 최대 이동 범위($V_{max} \times T_{interval}$)를 이용하여 질의 영역을 버퍼링 시킨 버퍼 질의 R'을 이용한다. 버퍼 질의는 이동 객체의 위치를 점으로 표현할 경우 연속된 두 점 사이의 위치 검색시 검색이 되지 않는 단점을 보완한 방법으로 최대 이동 범위만큼 질의 영역을 버퍼링 시켜 질의 영역을 확대한 형태의 기법으로 query

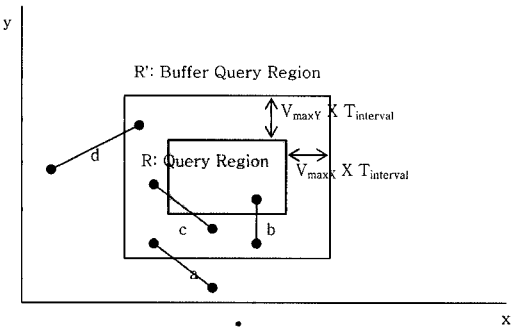


그림 2 버퍼 질의

window extension[3]에 기반한다. 버퍼 질의를 이용할 경우 검색 비용은 증가하는 단점을 가지지만, 질의 결과는 이동 객체의 이동 경로를 선분으로 저장하는 R-tree 기반 색인 구조와 동일하다.

4. 이동 객체의 특성으로 인한 KDB-tree 분할 정책의 문제점

4.1 분할 도메인 선정 방법의 문제점

이 절에서는 3 차원 시공간의 x, y, t도메인에서 이동 객체의 위치 정보가 삽입되어 분할이 발생하는 경우 분할 도메인 선정시 시간 도메인의 선형적인 증가를 고려한 분할 도메인 선정 방법을 분류하고 기존 방법에서의 문제점을 분석한다.

4.1.1 순환적 선정 방식

KDB-tree의 분할 도메인 선정방법은 순환적으로 분할 도메인을 선정하는 것이다. 이동 객체의 경우에는 3 차원 도메인을 대상으로 하므로 x, y, t도메인이 순서적으로 순환하면서 분할 도메인을 선정한다. 그림 3에서 보듯이 포인트페이지의 상한 값 M = 4일 때, ②에서 x 도메인으로 분할이 되고, ③에서 y 도메인으로 분할 되고, ④에서 t 도메인으로 분할되고 나면 다시 ⑤에서 x 도메인으로 분할되는 순환적인 방법이다. 순환적 분할 도메인 선정 방법은 모든 도메인에서 데이타의 분포를 포인트 페이지마다 균일하게 유지하지만, 시간 도메인이

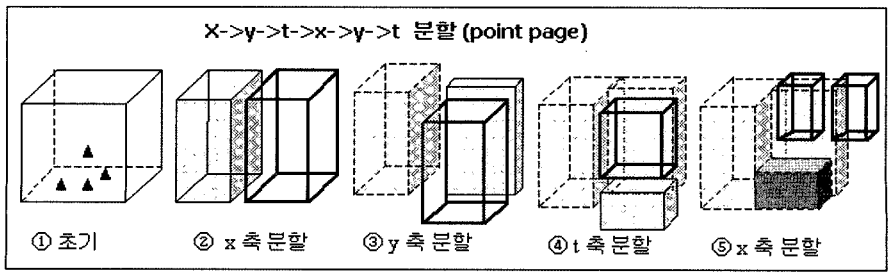


그림 3 순환적 분할방법

증가하는 시공간에서 이 방법은 공간으로 가중 분할 하는 결과가 된다. KDB-tree와 같은 공간 분할 방식의 색인은 공간의 범위가 한정되어 있다. 한정된 범위의 공간 도메인으로 가중 분할을 하게 되면 계속적으로 증가하는 시간 도메인에 비해 공간으로 조밀하게 분할되어 시간 질의 시 검색 성능이 낮아지는 단점이 있다.

4.1.2 시간 도메인 가중 분할

시간 도메인 가중 분할 방법은 x, y도메인 보다 t도메인으로 일정 비율에 따라 더 많이 분할하는 것을 의미한다. 그림 4에서 ④까지는' 순환적 분할 방법과 같고 ④이후로, 계속적으로 t 도메인으로 가중 분할을 하는 방법이다. 이 방법을 사용하면 공간 도메인보다 시간 도메인으로 더 많이 분할이 된다. t도메인으로의 분할이 많으므로 t도메인 분할 이후에 데이타의 입력이 발생하지 않는 과거 영역의 포인트페이지 생성 비율이 증가한다. 이것은 영역페이지 분할에서도 연계되어 문제를 발생시킬 수 있다. 기존의 영역페이지 분할 방식 중에서 FD 분할의 성능이 우수한 것으로 연구되어 있지만, 시간 도메인으로 가중 분할하게 되면 첫 번째 분할 도메인이 시간축으로 선정될 확률이 증가함에 따라 영역페이지의 분할이 시간 도메인으로 많이 발생하게 된다. 그 결과 과거의 영역이 되는 영역페이지를 많이 생성하게 되어 색인의 공간활용도를 저하시키게 되어 색인의 크기가 커지는 문제점을 발생시킨다.

4.1.3 긴 도메인 분할 방법

긴 도메인 분할 방법은 고정된 순서로 분할 도메인을 선정하는 것이 아니라 포인트페이지 분할 도메인을 선정할 때 분할이 발생한 포인트페이지를 구성하는 도메인 중에서 가장 긴 도메인을 선정해서 분할하는 방법이다. 이 방법은 고정된 순서에 의해 분할하는 것이 아니므로 포인트페이지를 구성하는 도메인들 간의 길이를 균등하게 한다. 그림 5에서 영역페이지 R1의 하위에 있는 포인트페이지 P1에 데이타 D16이 입력된 경우 y도메인으로 분할하여 효율적으로 공간을 분할한다. 그러나 긴 도메인 선정 방법은 시공간이 고정된 경우에는 우수한 성능을 보이지만, 시간 도메인이 증가할 경우 공간

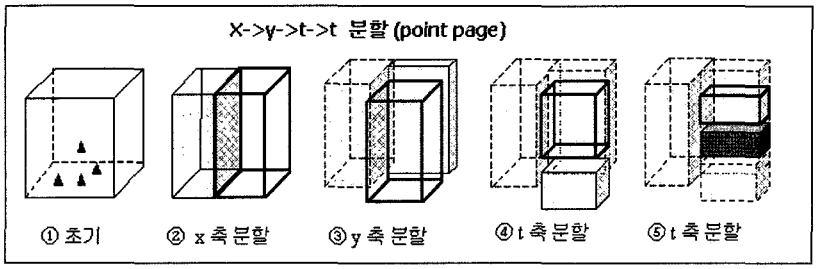
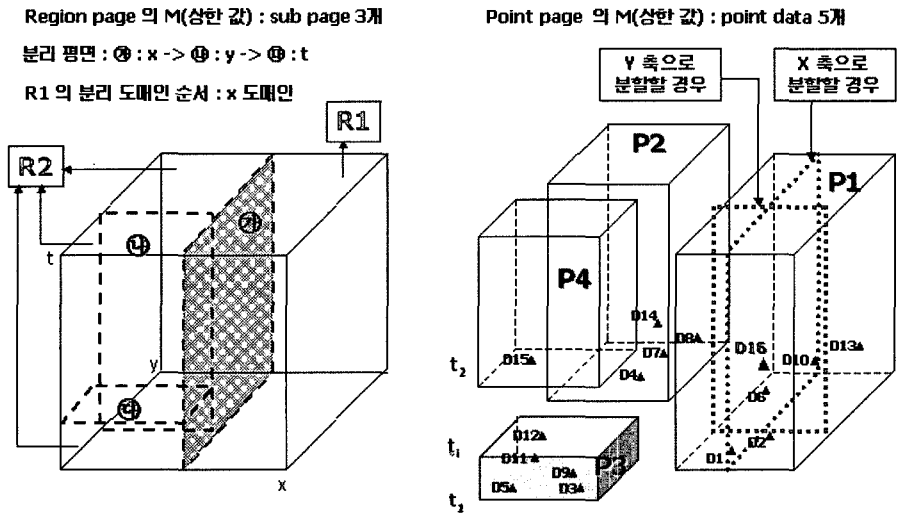


그림 4 시간 도메인 가중 분할방법



(a) 영역페이지 (b) 영역페이지의 하위에 있는 포인트페이지 들

그림 5 긴 도메인 선정 방법

분할의 누적으로 인해 발생하는 공간 영역의 세분화를 초래한다. 따라서 매우 조밀하게 분할된 공간 영역이 선정될 경우 타임 슬라이스 질의와 같이 특정 시간에 대한 영역 질의의 성능이 낮아지는 단점이 있다.

4.1.4 공간 우선 분할 방법

공간 우선 분할 방법이란 포인트페이지의 분할 시 공간 도메인 x, y에 대한 분할을 수행한 후 시간 도메인 t에 대한 분할을 하는 방법이다. 이 방법은 도메인 내의 이동 객체들의 초기 분포를 분석한 후 시간 도메인으로 분할하는 것으로 주기적으로 동일한 패턴을 가지는 이동 객체에 대한 분할 방법으로서 효율적이다. 예를 들어 도심의 교통은 1일을 주기로 반복된 패턴을 보인다. 이 경우 하루 동안의 입력된 데이타를 공간적으로 최적 분할하고 이후로 들어오는 데이타의 경우 공간 영역에 대한 시간 도메인 분할만을 수행하므로 영역 질의와 영역을 포함하는 시간 질의에 대한 검색을 효율적으로 할 수 있다. 그러나 공간 분할 후 시간 분할 방법의 경우 2가지 문제점이 있다. 첫째, 데이타의 입력에 따라 분할 시점을 공간 도메인 분할에서 시간 도메인 분할로 전환되는 전환점을 찾는 것이 필요하다. 둘째, 시간 도메인으로 분할 할 경우 기존의 균등 분할 방식으로 분할할 경우 과거 영역이 되는 포인트페이지의 공간활용도가 낮아지는 단점이 있다.

4.2 포인트페이지 분할의 문제점

이동 객체 데이타의 계속적인 삽입으로 인해 포인트페이지에 오버플로우가 발생되면 KDB-tree의 포인트페이지 분할은 포인트페이지 분할 도메인을 순환적으로 계산한다. 포인트페이지 분할 도메인이 선정되면 선택된 분할 도메인을 분할 축으로 데이타의 최대값과 최소값의 중간 값을 분할 점으로 선정하여 포인트페이지를 분할한다[7].

그림 6에서 보듯이 x 도메인(또는 y 도메인)으로 포인트페이지가 분할이 발생하면 두 개의 포인트페이지 ①, ②번을 생성한다. 이때 ①, ②번의 포인트페이지는 계속적으로 데이타가 삽입된다. 그러나 t 도메인으로 분할이 이루어져 생성된 ③, ④번의 포인트페이지에서 ③번 포인트페이지는 t 도메인의 특성상 t 도메인으로 분할된 시점 이후에는 과거 영역의 포인트페이지가 되어 더 이상 데이타의 삽입이 생기지 않게 된다. 이것은 삽입되는 이동 객체의 위치 데이타들이 시간 도메인에 대해 증가된 값을 가지고 삽입되기 때문이다.

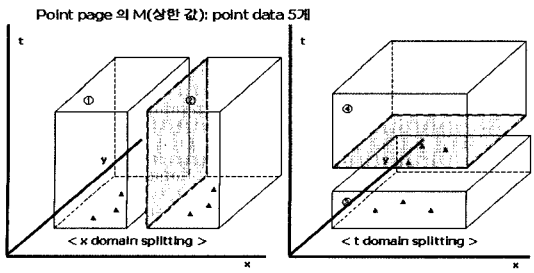


그림 6 포인트페이지의 분할

따라서 기존의 포인트페이지 분할 방식을 사용할 경우 이동 객체의 특성상 t도메인으로 분할 발생시 그림 6의 포인트페이지 ③은 더 이상 데이타가 입력되지 않으므로 공간활용도가 저하된다. 이것은 검색 시에 디스크 접근 빈도수를 증가시키는 원인이 되고 결국 색인의 탐색 성능을 저하시키게 된다.

4.3 영역페이지 분할의 문제점

4.3.1 FS 분할 (Forced Splitting)의 문제점

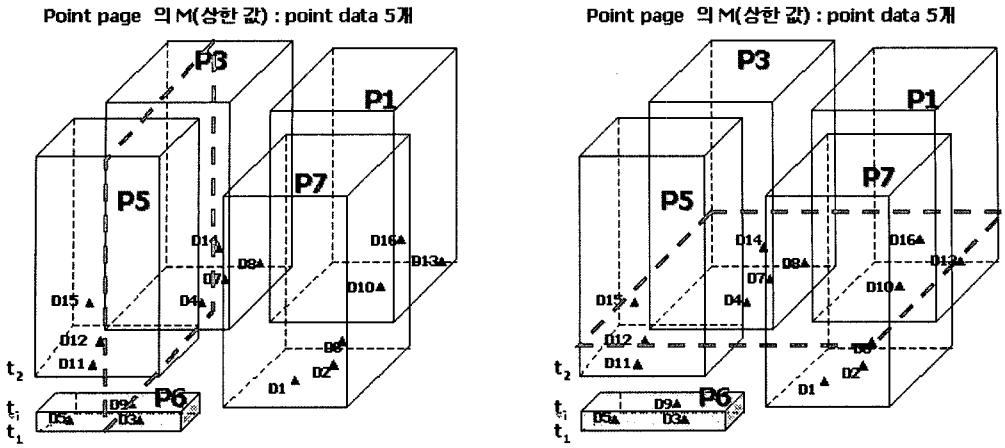
FS분할은 기존의 연구[8]에서 문제점이었던 강제적 분할 전파 문제로 인해 포인트페이지를 과도하게 분할한다. 영역페이지가 강제적으로 하위의 포인트페이지에 분할을 전파하게 되면 포인트페이지들의 공간활용도를 저하시킨다. 그림 7(a)는 영역페이지의 분할이 공간 도메인 x로 발생했을 때 강제적으로 하위의 포인트페이지들이 분할되는 예를 보이고 있다. 포인트페이지 P3, P5, P6이 강제 분할되고 있고 특히 포인트페이지 P6의 경우는 과거에 t도메인으로 분할이 발생하여 더 이상 데이타가 입력되지 않는 포인트페이지인데 강제 분할에 의해 더욱 공간활용도가 낮아진다.

그림 7(b)는 영역페이지의 분할이 시간 도메인 t로 발생 했을 경우 강제적인 분할 전파에 의해 포인트페이지 P1, P3, P5, P7이 분할된다. FS분할을 사용할 경우 강제로 분할 시점이 아닌 다수의 포인트페이지가 분할되어 색인의 크기를 증가시키고 또한 분할 도메인이 t도메인으로 분할이 발생한 포인트페이지 중에서 분할 시점보다 과거의 영역이 되는 포인트페이지들은 색인의 공간활용도를 저하시키게 된다.

4.3.2 FD 분할(First Division Splitting)의 문제점

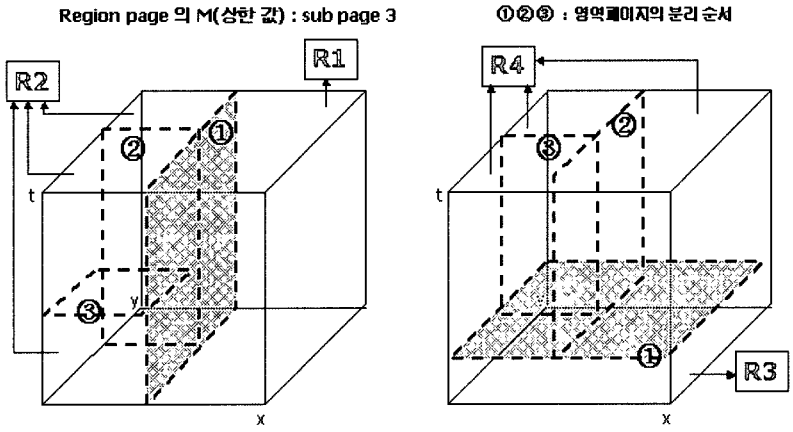
FD 분할은 영역 페이지를 분할할 때 하위 영역으로 강제적인 분할 전파가 일어나는 것을 제거 함으로서 불필요한 포인트페이지의 분할을 감소시켜서 다차원 색인에서 성능이 좋은 것으로 연구되었다[6]. 시공간에서 이동 객체의 위치 데이타는 t도메인 값의 증가에 선형적인 분포를 갖는 특징으로 인해 영역페이지의 분할 정책으로 FD분할을 사용할 경우 처음으로 분리가 발생한 도메인에 의해 분할 성능이 달라진다.

그림 8(a)에서 영역페이지는 처음 영역 분할이 x 도메인에서 발생한다. 세 번째 영역 분할(분할 평면: ③)로 인해서 영역페이지의 하위 영역이 4개가 되어 오버플로우가 발생한다. FD분할을 사용하면 영역페이지의 분할이 처음 분리되었던 분리 축(또는 평면)으로 영역페이지



(a) 공간도메인 x로 강제 분할된 예 (b) 시간도메인 t로 강제 분할된 예

그림 7 FS분할로 인한 포인트페이지들의 강제 분할



(a) 첫번째 분리 x 도메인 (b) 첫번째 분리가 t 도메인

그림 8 영역페이지의 FD 분할 예

의 분할이 일어난다. 그림 8(a)의 영역페이지 경우 ①번 분할평면(x 도메인)으로 분할하여 두 개의 영역페이지 R1과 R2를 생성한다. 그림 8(b)의 영역페이지는 처음 영역 분할이 t도메인으로 발생했기 때문에 오버플로우가 발생하면 영역페이지는 ①번 분리 평면(t도메인)으로 분할하여 두 개의 영역페이지 R3, R4를 생성한다. 그림 8(b)의 영역페이지의 경우에서 t도메인의 특성상 t도메인 값의 상승 방향으로 데이타가 입력 된다고 했을 때 t 도메인으로 분할된 두 개의 영역페이지 R3, R4중에서 영역페이지 R3의 하위에 있는 포인트페이지에는 더 이상 데이타가 삽입되지 않게 되는 문제가 발생한다.

5. 동적 분할 정책

5.1 분할 도메인 선정 방법

5.1.1 공간 우선 분할 방법

공간 우선 분할 방법의 경우 공간 도메인으로 우선적으로 분할 한 뒤, 시간 도메인으로 분할하는 기법이다. 공간에 대해 우선적으로 분할하는 이 기법은 이동 객체의 움직임이 주기적인 패턴을 가지는 경우 효율적이다. 즉 초기 이동 객체의 이동 패턴으로 공간 분할을 최적화 한 뒤 동일 패턴으로 입력되는 데이타를 시간 축으로 분할한다.

공간우선 분할 기법에서 가장 중요한 것은 공간 분할에서 시간 분할로 전환되는 전환점을 찾는 것이다. 이동 객체가 주기적 패턴을 가질 경우 주기 T동안 이동한 위치를 공간 평면으로 투영한 뒤 공간 평면에서 분할할 경우 분할된 영역이 공간적으로 너무 작아지는 단점이 있다. 이를 해결하기 위해 이 논문에서는 한 주기 동안

보고된 위치의 평균 공간 영역을 정의하고 평균 공간 영역의 비율에 따라 분할 시점을 선정하는 방법을 사용한다.

표 1 전환시점 산출 요소

N_{MO}	이동 객체의 개수
M,m	포인트페이지의 최대/최소 엔트리 개수
T	이동 객체의 응집확산 주기
$T_{sampling}$	한 주기 동안의 이동 객체의 평균 보고 회수
N_{TPP}	주기 동안의 총 포인트 페이지 수
$Area(U_{Spatial})$	전체 공간 도메인의 면적
MA_{pp}	포인트페이지의 평균 면적의 크기
DRD	밀집 지역 밀도

식 (1)에서는 먼저 이동 객체의 개수(N_{MO}), 주기 당 평균 보고횟수($T_{sampling}$)를 이용하여 주기 T동안 이동 객체가 보고하는 위치 정보의 개수를 산출한다. 산출된 위치 정보가 모든 포인트페이지에 균등하게 배분될 경우 각 페이지는 최대 M개의 위치 정보 저장이 가능하다. 페이지당 저장되는 위치 정보의 개수를 추출하기 위해 최소 엔트리 수(m)를 이용할 경우에는 기존 분할 방법에 의한 포인트 페이지의 영역만이 계산되어 시간에 따른 이동 객체의 특성을 고려한 분할 정책에서 높은 공간 활용도를 달성하는 방법과 차이가 있다. 따라서 공간 활용도가 최대한 확장되는 시점에서의 포인트 페이지의 평균 영역을 계산하기 위해 최대 엔트리 수(M)를 이용하여 주기 동안 총 포인트 페이지 수 N_{TPP}를 산출한다. 포인트 페이지의 평균 영역의 크기(MA_{PP})를 구하기 위해 전체 공간 영역($Area(U_{spatial})$)을 주기 당 총 페이지의 개수(N_{TPP})로 나누어 식 (2)와 같이 산출한다.

$$N_{TPP}=\frac{(N_{MO}*T_{sampling})*T}{M} \quad (1)$$

$$MA_{PP}=\frac{Area(U_{spatial})}{N_{TPP}} \quad (2)$$

빈번한 위치 보고를 하는 지역을 밀집 지역이라 할 때, 밀집 지역에서는 잦은 분할이 발생하기 때문에 색인의 구성에 큰 영향을 준다. 밀집 지역은 포인트페이지의 영역이 MA_{PP}보다 작은 포인트페이지라 할 수 있다. 따라서 밀집 지역 밀도(Dense Region Density)는 식 (3)과 같이 전체 공간 영역에서 밀집 지역의 밀도로 정의한다.

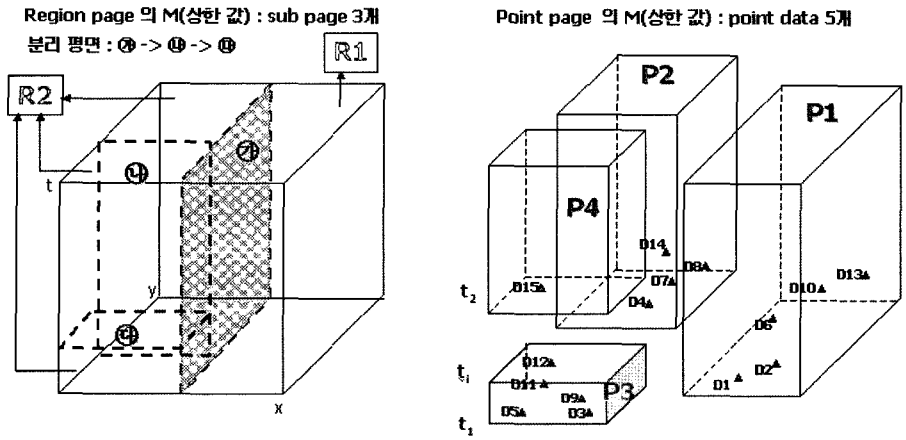
$$DRD=\frac{\sum Area(pp_i)}{Area(U_{spatial})}*100, \{pp_i \mid Area(pp_i) \le MA_{pp}\} \quad (3)$$

공간 우선 분할 기법에서 공간 분할에서 시간 분할로의 전환 시점은 DRD로 선정된다. 즉 밀집 지역 밀도가 높을 경우 평균 영역 이하로 분할되는 포인트 페이지의 개수가 많아지고, 밀집 지역 밀도가 낮은 경우 공간으로 분할하는 회수가 줄어들게 된다. 따라서 분할 시점은 데이타 세트의 DRD에 따라 달라지게 된다. 효율적인 검색을 위해 5.1의 실험을 통해 밀집 지역 분포에 따라 분할 시점을 선정한다.

5.2 포인트페이지 분할 정책

5.2.1 최근 시간 분할(Recent Time Division)정책

포인트페이지에서 시간 도메인 t로 분할 할 경우 균등 분할 방법을 적용하면 분할된 아래 영역(t 도메인을 기준으로)에는 추가적인 데이타의 삽입이 발생하지 않아 공간활용도가 떨어져 색인의 크기를 증가시키는 단점이 있다. 이 문제를 해결하는 방법으로 이 논문에서는 포인



(a) 영역페이지 (b) 영역페이지의 하위에 있는 포인트페이지 들

그림 9 최근시간 분할 기법의 예

트페이지 분할시 시공간 도메인 중에서 t도메인으로 분할 할 때 가장 최근 시간을 가진 데이타와 나머지 데이타들로 분할하는 최근 시간 분할 기법을 제안한다.

최근 시간 분할 기법이란 3차원 시공간 도메인에서 분할 도메인이 시간 도메인 t로 선정될 경우 기존 KDB-tree의 포인트페이지 분할 방법으로 분할하지 않고 그림 9와 같이 분할 이후 생기는 결과 포인트페이지들 중에서 t도메인의 분할 시점보다 과거의 시점이 되는 영역의 포인트페이지 P3에는 최근 시각 데이타를 제외한 모든 데이타를 저장하고 미래의 시점이 되는 영역의 포인트페이지 P4에는 마지막 보고된 최근 데이타 1개를 저장하는 분할 방법을 의미한다.

따라서 분할 시각인 ti이후에 입력되는 데이타는 $[t_1, t_i)$의 시간 영역을 가진 포인트페이지 P3에는 삽입되지 않는다. 포인트페이지 분할을 할 때 포인트페이지 P3에 포인트페이지의 상한 값과 같은 수인 5개의 데이타를 넣어서 분할하므로 공간 활용도를 향상시킨다. 이럴 경우 $[t_i, t_2)$ 영역인 포인트페이지 P4에는 M-1개인 4개의 데이타가 계속적으로 입력이 발생하더라도 분할이 발생하지 않아서 최대한 분할을 연장시키는 효과를 가져온다. 즉 최근 시간 분할은 분할 되는 두 영역의 특징을 반영하여 분할을 함으로써 과거의 영역에는 상한 값의 데이타를 넣고 미래의 영역에는 앞으로 계속적으로 입력되는 특징을 반영하여 최소한의 데이타를 넣어 분할을 하게 된다. 이것을 실제 데이타의 입력에 따른 트리로 구성하면 그림 10과 같다.

그림 10과 같이 KDB-tree의 포인트페이지 분할 시 이동 객체의 특성을 고려한 최근 시간 분할 기법은 포인트페이지의 공간활용도를 향상시키고 분할 노드의 수를 감소시켜서 전체 색인의 탐색시간을 향상 시킨다.

5.3 영역페이지 분할 정책

5.3.1 최후 분할(Last Division) 정책

LD분할이란 영역 페이지를 분할 할 때 마지막으로 분할된 분할 축(평면)으로 영역 페이지를 분할하는 방법이다. 영역 페이지의 분할 정책에서 분할될 영역 페이지가 공간 축으로 분할된 엔트리를 가질 경우 FD분할이 효율적이지만, 시간 축 엔트리로 구성될 경우 FD분할은 성능 저하를 초래한다. 이 논문에서는 영역 페이지 분할 시 공간 우선 분할 기법에서 공간 도메인으로 분할 할 경우 기존의 FD 분할을 사용하고 시간 도메인으로 분할 할 경우에는 LD분할을 사용한다.

그림 11에서 포인트페이지 P3이 분할된 분할 시점부터 시간 도메인 분할을 시작한다고 했을 때 계속적으로 시간 도메인 t로 분할이 발생할 것이다. 포인트페이지

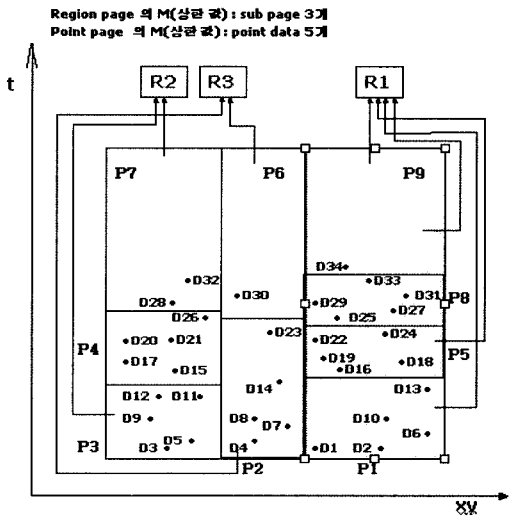


그림 11 영역페이지의 분할

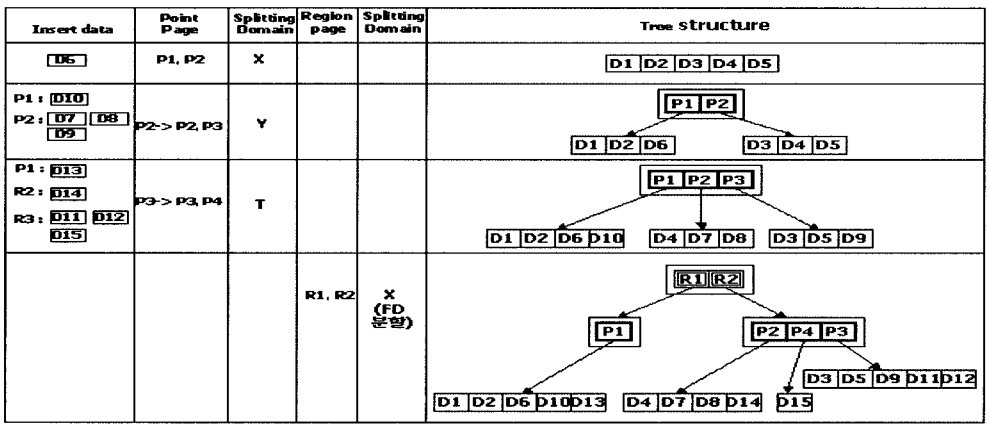


그림 10 최근시간 기법을 사용한 색인의 구성

분할에 의해 포인트페이지 P9가 새로 생성되면, 영역페이지 R1에 오버플로우가 발생된다. 이 시점에서 기존의 FD 분할을 사용하면 영역페이지 R1의 첫번째 분할 축이 P1과 P5의 경계가 된다. 이럴 경우 과거의 영역이 되는 영역페이지에 포인트페이지 P1만을 삽입하게 되므로 노드의 공간활용도를 저하시키게 된다. 반대로 영역페이지 R1의 분할 시점에서 LD분할을 사용하게 되면 과거의 영역이 되는 영역페이지에 P1, P5, P8을 삽입하고 새로운 영역페이지에 마지막 분할에 의해 발생한 포인트페이지 P9를 삽입하게 된다. 즉 시간 분할 이후의 영역페이지 분할에서는 첫번째 분리된 분리 평면 보다는 마지막으로 분리된 분리 평면으로 분할하는 것이 데이타의 삽입에 의해 계속적으로 시간 도메인 t로 분할이 발생한다고 했을 때 더 효과적이다.

그림 12 및 그림 13에서는 시간 도메인 t로 분할을 계속할 때 LD 분할에 의해 생성되는 트리를 나타낸다. 그림 12에서 영역페이지 분할 전 트리의 경우 포인트페이지 P1, P2, P3, P4, P5는 포인트페이지의 최근 시간 분할을 한 상태이고 현재 포인트페이지 P6, P7, P8로만 데이타가 입력되고 있다. 포인트페이지 P8에 한 개의 데이타가 입력이 될 경우 분할이 발생한다. P8의 분할은 영역페이지 R1의 분할을 유도한다. 이때 FD분할을 사용하면 그림 13(a)의 트리 처럼 영역페이지 R1은 포인트페이지 P1만을 하위 영역으로 갖고 영역페이지 R4는 포인트페이지P5, P8, P9를 하위 영역으로 가지고 분할이 발생한다. 그러나 LD분할을 사용하면 그림 13(b)의 트리와 같이 영역페이지 R1은 포인트페이지 P1, P5, P8을 가지고 영역페이지 R4는 포인트페이지 P9만을 가지고 분할하게 된다. LD 분할을 사용할 경우 영역페이지 R1의 경우 과거 영역의 영역페이지가 되면서 공간활용도가 높은 포인트페이지들을 가지고 분할을 한 것을 볼 수 있다.

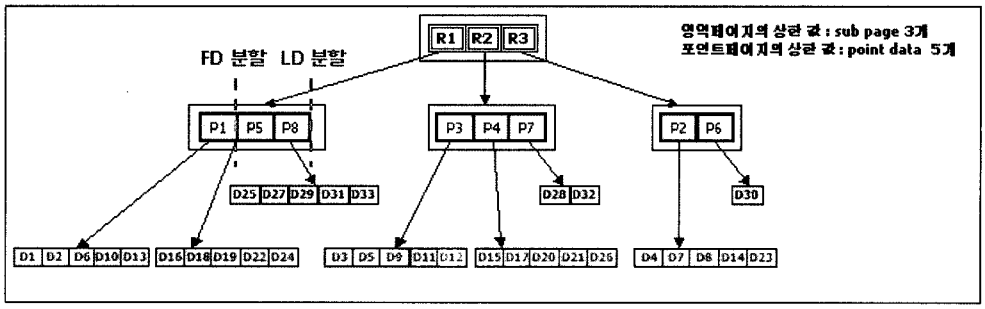


그림 12 영역페이지 분할 전의 트리

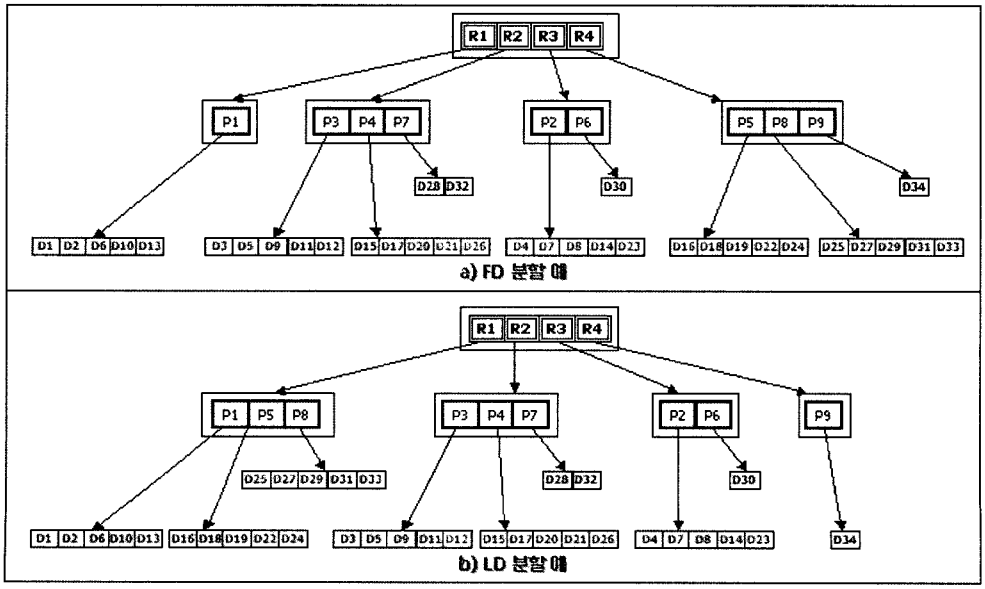


그림 13 FD 분할과 LD 분할 의 예

6. 성능 평가

이 논문에서는 3차원 시공간을 대상으로 한 이동 객체 데이타를 색인 하는데 있어서 MV3R-tree, 3D-Rtree, 기존의 분할 정책을 사용하는 KDB-tree와 이 논문에서 제안하는 분할 정책을 사용하는 변경된 KDB-tree(MKDB-tree)의 색인 구축 성능과 질의 처리 성능을 평가한다. 실험 환경으로는 1.7GMHZ의 CPU와 512Mbyte의 메모리를 가진 Linux 환경의 Personal Computer를 사용한다.

색인의 성능 평가에서 사용될 데이타의 집합은 GSTD 알고리즘[11]을 기반으로 생성한다. KDB-tree와의 성능 비교를 위하여 데이타셀은 0과 1사이의 정규화된 데이타로 변환한다. 데이타 집합은 객체 수에 따라 1000개, 2000개, 5000개, 10000개이고, 200회의 위치를 보고하는 가정으로 생성된다. 초기 생성시 분포는 가우시안(평균:0.5, 분산:0.1)이고 위치 이동시 시간축에 따른 위치 보고 간격과 중심 이동분포도 가우시안 분포에 따른다.

6.1 분할 시점 선정

공간 우선 분할 기법에서 분할시점을 선정하기 위해 밀집 지역 밀도에 따른 성능을 분석한다. 그림 14는 이동 객체의 수(1000, 2000, 5000, 10000)에 따른 4가지 데이타셀에 대한 밀집 지역 밀도(DRD)에 따른 영역 질의 성능을 비교한 그래프이다. 그리고 노드의 크기를 512, 1024, 2048Byte로 달리하면서 실험을 하였다. 그 결과 밀집 지역 밀도가 작을수록 좋은 성능을 보이고 있다. 그러나, 밀집 지역 밀도가 전체 영역에 대한 평균 면적의 비율 이하일 경우에는 동일한 성능을 나타내고, 추가적으로 평균 면적의 크기를 감소시킨 실험에서는 평균 면적의 크기와 성능이 반비례하였기 때문에 이후 실험에서는 분할 시점을 밀집 지역 밀도가 0.1%로 선정한다.

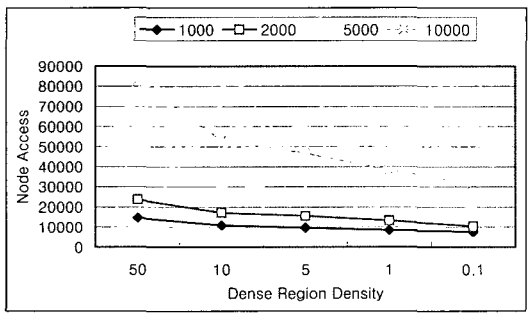


그림 14 밀집 지역 밀도에 따른 성능 비교

6.2 삽입 성능 비교

이 실험에서는 이동 객체의 데이타의 삽입에 따른 색인 성능 비교 평가를 위해 구축된 색인의 공간활용도 및 생성된 노드 수를 비교한다. 노드 크기는 1024byte이고, 4가지 색인의 비단말노드(MBB:24byte, RRN:4byte)의 엔트리 수는 36, 단말 노드의 경우 3DR-tree는 MBB, Line ID로 구성되므로 엔트리수가 36, KDB-tree, MKDB-tree, MV3R-tree는 Point, Point ID로 구성되므로 엔트리수가 64이다.

6.2.1 공간활용도

그림 15에서 MKDB-tree의 공간활용도가 다른 3개의 색인에 비해서 월등함을 볼 수 있다. 기존의 분할 정책을 사용하는 KDB-tree에 비해 20%~30%의 성능 향상이 된 결과를 보이는데 이것은 시간 도메인의 특성을 고려하여 시간 분할 시에 균등 분할을 하지 않고 최근 시간 분할을 수행함으로써 이동 객체와 시간 도메인의 특성을 반영해준 결과라고 볼 수 있다.

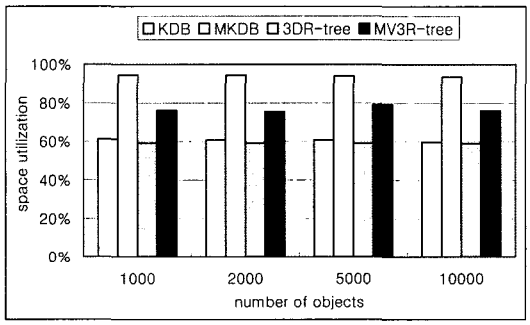


그림 15 이동 객체의 개수에 따른 공간활용도의 비교

6.2.2 생성 노드 수

생성된 노드의 경우는 MKDB-tree에서 노드 생성의 개수가 가장 적은 것을 볼 수 있다. 3D-Rtree의 경우 단말 노드의 엔트리 개수는 KDB-tree의 절반 수준이기 때문에 생성된 노드의 개수의 상대적으로 높다. 또한 기존 분할 정책을 사용하는 KDB-tree는 이동 객체의 특성을 고려하지 않기 때문에 공간 활용도가 MKDB-tree에 비해 상대적으로 낮아 생성되는 노드의 수가 30%이상 많게 된다.

6.3 검색 성능

시공간 영역 질의(Spatio-temporal range Query)의 성능을 비교하기 위해 3가지 종류의 질의 영역을 이용하여 실험한다. 시공간 영역 질의는 시간 차원을 공간의 다른 차원으로 간주하여 처리한다[11,12]. 따라서 실험에서 질의 영역은 데이타 세트의 전체 시공간 도메인을 정규화하여 전체 시공간 영역의 5%, 10%, 20% 크기의 질의 집합을 구성한다. 각 질의 집합의 개수는 1000개이다. 검색 성능은 각 질의 집합의 검색 시 노드 접근 회수를 누적하여 비교한다. 검색 질의의 경우 3DR-tree에

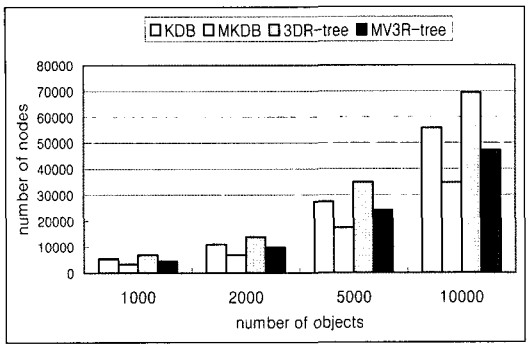


그림 16 이동 객체의 개수에 따른 생성된 노드 수의 비교

서 검색한 결과와 동일한 결과를 얻기 위해 데이타세트 생성시 이동 객체의 최대 이동 범위를 이용하여 버퍼 질의 모델을 이용한다.

그림 17, 그림 18, 그림 19에서는 질의 크기에 따른 검색 성능을 보이고 있다. 영역 질의 결과 단말 노드에 MBB를 저장하는 3DR-tree의 검색 성능은 점을 저장하는 KDB-tree에 비해 작은 크기의 데이타 세트에서는 색인의 높이의 차이로 인해 검색 성능이 3배 이상 차이를 나타내지만, 데이타 세트 크기가 증가 함에 따라 KDB-tree의 검색 성능은 3DR-tree에 비해 평균 30%의 성능 향상을 나타낸다. 이것은 색인의 높이가 차이가 적은 데이타 세트일 경우 버퍼 질의로 인한 노드 접근 회수가 증가하기 때문에 성능 향상 비율이 낮아진다. 그러나 동일한 KDB-tree에서 분할 정책을 달리할 경우 MKDB-tree는 KDB-tree에 비해 38%~73%정도의 성능 향상을 나타낸다. 특히 질의의 영역이 커질 경우 높은 공간 활용도로 인해 모든 데이타셀에서 성능 향상 비율이 유사함을 알 수 있다. 또한 시공간 질의에 우수한 MV3R-tree의 경우에도 KDB-tree와 3DR-tree에 비해 우수한 성능을 보이고 있지만, MKDB-tree는 이동 객체 데이타의 시간적 특성을 이용한 최근 시간 분할 기법의 적용으로 MV3R-tree보다 28~35% 향상된 성능을 보이고 있다.

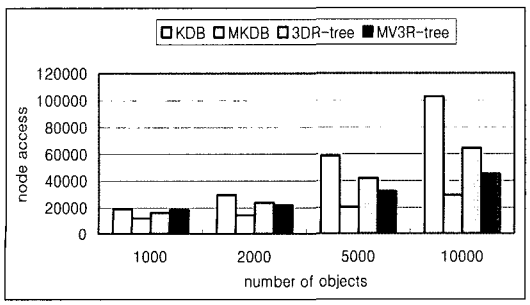


그림 17 데이타 세트별 5% 영역 질의 결과

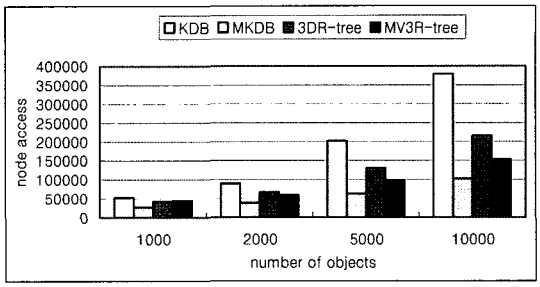


그림 18 데이타 세트별 10% 영역 질의 결과

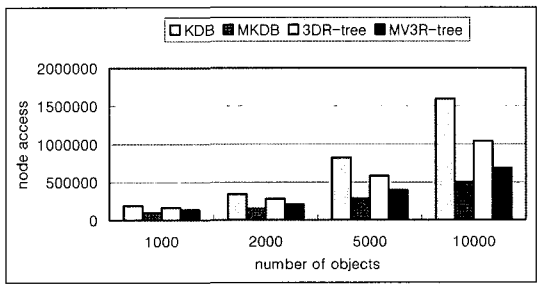


그림 19 데이타 세트 별 20% 영역 질의 결과

7. 결 론

이 논문에서는 시공간에서의 이동 객체의 과거 위치 데이타를 검색하기 위해 KDB-tree를 사용할 경우 이동 객체의 특성으로 인하여 발생하는 KDB-tree의 문제점을 기술하고, 이를 해결하여 색인의 검색 성능을 향상시키기 위해 변경된 분할 정책들을 제안하였다. 우선 포인트페이지 분할 방식의 문제를 해결하기 위해 포인트페이지의 분할 도메인 선정 방법을 분류하고 시공간에서 이동 객체 색인에 효과적인 공간 우선 분할 기법과 분할 시점 선정 방법을 제안하였다. 그리고 포인트페이지의 분할 시 공간 도메인인 x, y도메인으로 분할 할 경우에는 기존의 방법을 사용하고 시간 도메인으로 분할을 할 경우에는 분할로 인해 생성되는 결과 포인트페이지를 시간 도메인 값이 가장 큰 데이타와 나머지 데이타들로 분할하여 과거의 영역이 되는 포인트페이지의 엔트리 수가 M이 되게 하는 최근 시간 분할 기법을 사용한다. 영역페이지 분할 정책으로 공간 분할 시에는 FD분할을 사용하고 시간 분할을 할 때는 이 논문에서 제안한 LD분할을 사용하여 시공간에서의 시간 도메인을 고려한 영역페이지 분할 정책을 제안하였다.

시공간에서의 이동 객체 색인을 위해 시간 도메인을 고려한 KDB-tree의 분할 정책들을 구현하고 성능평가 실험을 실시하여 시간 도메인을 고려한 분할 정책을 사용한 KDB-tree의 성능이 기존의 KDB-tree의 성능보다 우수한 것을 알 수 있었다. 이 논문에서 제시한 분할

정책을 사용한 KDB-tree의 경우 색인의 구축비용이나 영역질의에 대한 검색 비용에서 향상된 성능을 보였으며 특히 이동 객체를 위한 시공간 색인인 MV3R-tree와 SAM 방식인 3D-Rtree와 Buffer질의를 사용한 영역질의에 대한 검색 성능 비교에서도 우수한 것을 볼 수 있었다.

참 고 문 헌

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

[2] G.Kollios, D. Gunopulos, and V. J. Tsotras, "On Indexing Mobile Objects," Int. conf. PODS, pp. 261-272, 1999.

[3] E.Stefanakis, Y.Theodoridis, Timos Sellis, "Point Representation of Spatial Objects and Query Window Extension: A New Technique for Spatial Access Methods," International Journal of Geographical Information Science vol.11, no.6, pp. 529-554, 1997.

[4] Volker Gaede, Oliver Günther, "Multidimensional Access Methods," ACM Computer Survey vol. 30, no. 2, pp. 170-231, 1998.

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

[6] Ratko Orlandic, Byunggu Yu, "Implementing KDB trees to Support High-Dimensional Data," Int. conf. IDEAS, pp. 58-67, 2001.

[7] J.T. Robinson, "The K-D-B Tree: A Search Structure For Large Multidimensional Dynamic Indexes," Int. Conf. ACM SIGMOD, pp. 10-18, 1981.

[8] M. Freeston, "A General Solution of the N-dimensional B tree Problem," Int. Conf. ACM SIGMOD, pp. 80-91, 1995.

[9] A. Henrich, H.-S. Six and P. Widmayer, "The LSD Tree: Spatial Access to Multidimensional Point and Non-Point Objects," Int. conf. VLDB, pp. 45-53, 1989.

[10] B. Seeger and H.P. Kriegel, "The Buddy tree: An Efficient and Robust Access Method for Spatial Data Base Systems," Int. conf. VLDB, pp. 590-601, 1990.

[11] Y.Theodoridis, J.R.O Silva, and M.A Nascimento, "On the Generation of Spatio-Temporal Datasets," Int. conf. SSD, LNCS 1651, Springer, pp. 147-164, 1999.

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

임 덕 성

정보과학회논문지 : 데이타베이스
제 33 권 제 2 호 참조

이 창 헌

2001년 동의대학교 컴퓨터공학과 졸업(학사). 2003년 부산대학교 컴퓨터공학과 졸업(공학석사). 2003년~현재 부산정보과학고등학교 교사. 관심분야는 시공간 데이타베이스, 이동체 데이타베이스

홍 봉 희

정보과학회논문지 : 데이타베이스
제 33 권 제 1 호 참조