TPR-tree의 성능 예측을 위한 비용 모델
(A Cost Model for the Performance Prediction of the TPR-tree)

최 용 진 † 정 진 완 ††
(Yong-Jin Choi) (Chin-Wan Chung)

요 약 최근에 움직이는 객체의 미래 위치를 위한 TPR-tree가 제안되었으며, TPR-tree를 이용한 많은 연구들이 제안되었다. 그러나, TPR-tree가 시공간 데이타베이스에서 널리 사용됨에도 불구하고, TPR-tree를 위한 비용 모델은 제안되지 않았다. R-tree와 같은 공간 색인을 위한 비용 모델들은 움직이는 객체들의 미래 위치를 전혀 고려하지 않기 때문에, TPR-tree에 대한 시공간 질의를 위한 디스크 액세스 수를 정확하게 예측하지 못한다. 본 논문에서는 움직이는 객체들의 미래 위치를 고려한 TPR-tree를 위한 비용 모델을 처음으로 제안한다. 다양한 실험 결과, 제안된 TPR-tree의 비용 모델은 디스크 액세스 수를 정확하게 예측한다.

키워드 : 시공간 데이타베이스, 움직이는 객체, TPR-tree, 비용 모델

Abstract Recently, the TPR-tree has been proposed to support spatio-temporal queries for moving objects. Subsequently, various methods using the TPR-tree have been intensively studied. However, although the TPR-tree is one of the most popular access methods in spatio-temporal databases, any cost model for the TPR-tree has not yet been proposed. Existing cost models for the spatial index such as the R-tree do not accurately estimate the number of disk accesses for spatio-temporal queries using the TPR-tree, because they do not consider the future locations of moving objects. In this paper, we propose a cost model of the TPR-tree for moving objects for the first time. Extensive experimental results show that our proposed method accurately estimates the number of disk accesses over various spatio-temporal queries.

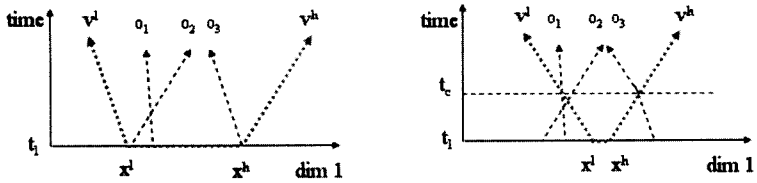
Key words : spatio-temporal databases, moving object, TPR-tree, cost model

1. 서 론

최근에 시공간 데이타베이스가 집중적으로 연구되었다. 대부분의 연구는 이동객체 모델링[1,2]과 색인[3-8] 분야에서 활발하게 진행되었다. 본 논문은 움직이는 객체의 미래 위치와 관련된다[2]. 차, 비행기들은 시간에 따라 움직이는 객체들로써 표현될 수 있다. 최근 갱신 정보를 가지고 객체의 미래 위치를 관리 할 수 있는 모델이 제안되었다[2]. 이러한 모델링은 움직이는 객체의 미래 위치를 시간 함수로 표현하며, 객체의 갱신 수를 감소시키는 효과가 있다. 최근 많은 연구들이 이 모델링을 바탕으로 연구되었다[3,4,6,7,9,10]. 본 논문도 이 모델링 기반의 시공간 색인과 관련된다.

움직이는 객체들의 미래 위치에 대한 시공간 질의를 지원하는 R-tree 기반의 TPR-tree가 제안되었다[7]. 공간 데이타베이스에서 R-tree[11]가 가장 널리 사용되는 색인처럼, 움직이는 객체들을 위한 시공간 데이타베이스에서 TPR-tree는 널리 사용되는 색인이다. 최근에 TPR-tree를 사용한 다양한 연구들[6,9,12]이 제안되었다. R-tree가 널리 사용되는 이유 때문에, R-tree에 대한 비용 모델의 연구가 많이 이루어졌다[13-15]. 그러나, TPR-tree가 시공간 분야에서 널리 사용되고 있음에도 불구하고, 아직까지 TPR-tree를 위한 비용 모델은 제안되지 않았다. 그 결과, 최적화기(optimizer)는 시공간 질의를 위한 비효율적인 계획(plan)을 사용할지도 모른다. 존재하는 공간 비용 모델들은 현재 시간의 공간 위치만을 고려하기 때문에, 시공간 질의를 위한 디스크 액세스 수를 정확하게 예측하지 못한다. 따라서, 객체들의 미래 위치를 고려하는 TPR-tree를 위한 비용 모델이 필요하다.

· 본 논문은 정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신기초기술연구지원사업(과제번호 : 03-기초-0114)의 연구결과입니다.
† 비 회 원 : 한국과학기술원 박사후과정
omni@islab.kaist.ac.kr
†† 종신회원 : 한국과학기술원 전자전산학과 교수
chungcw@cs.kaist.ac.kr
논문접수 : 2003년 8월 12일
심사완료 : 2004년 2월 20일



(a) t_l (b) t_c

그림 1 TPR-tree의 시간 경계 간격

본 논문에서는, TPR-tree를 이용하여 시공간 질의에 대한 디스크 액세스 수를 정확하게 예측할 수 있는 비용 모델을 제안한다. TPR-tree의 비용 모델을 위해, TPR-tree를 묘사하는 시공간 통계치를 제안한다. 이러한 통계치는 히스토그램 기반이며, 히스토그램은 임의의 데이타 분포를 잘 표현할 수 있으며 작은 오차율을 위해서 널리 사용되었다[16]. 또한, 본 논문에서는 적은 비용으로 시공간 통계치를 유지하는 방법을 제시한다. 실험 결과, 제안된 비용 모델이 다양한 시공간 질의에 대해서 정확한 디스크 액세스 수를 예측하였다. 지금까지, 움직이는 객체들을 위한 실세계 데이타들이 아주 미비한 관계로, 다른 연구들[4,7]처럼 인위적으로 움직이는 객체를 생성하여 실험하였다. 좀 더 현실적인 실험 환경을 위해서, 공간 데이타베이스에서 널리 사용된 Tiger/lines 데이타[17]와 Sequoia 데이타[18]를 이용하였다. 제안된 방법이 움직이는 객체를 위한 TPR-tree의 비용 모델에 관한 첫 번째 연구이다. 그래서, 제안된 시공간 비용 모델의 효과를 평가하기 위해서, 기존의 공간 비용 모델과 비교를 하였다. TPR-tree의 성능을 예측하기 위해서 기존의 공간 비용 모델을 사용하였을 때, 움직이는 객체의 미래 위치를 고려하지 않는 공간 비용 모델은 52%~93%의 평균 오차율을 보였다. 반면에, 우리의 방식은 11%~32%의 비교적 좋은 평균 오차율을 보였다.

본 논문의 구성은 다음과 같다. 2장 관련 연구에서는 TPR-tree를 간략하게 설명한다. 3장에서는 TPR-tree를 잘 묘사하는 시공간 통계치를 설명한다. 4장에서는 TPR-tree를 이용하여 시공간 질의에 대한 액세스 수를 예측하는 방법을 설명한다. 그리고, 5장에서는 실험 결과를 보이고 6장에서는 결론을 내린다.

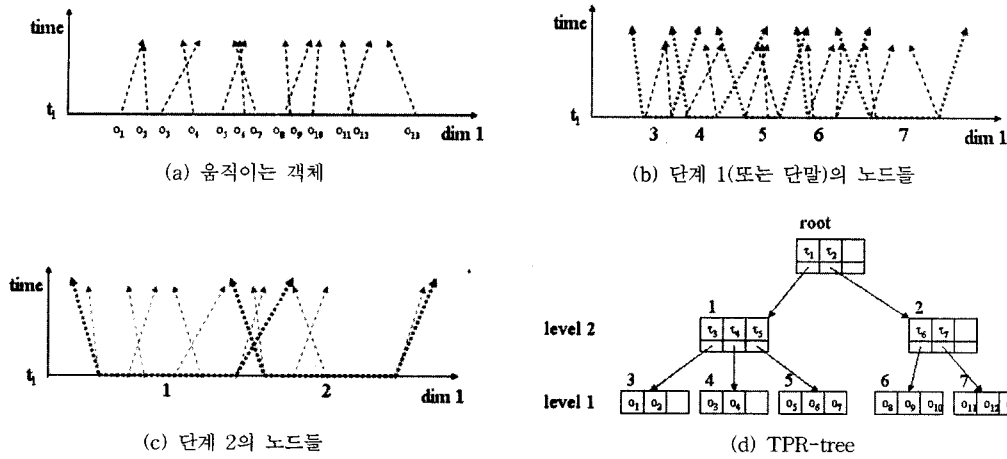
2. 관련 연구

시공간 데이타베이스에서는 시간의 흐름에 따라 움직이게 되는 객체들을 다룬다. 움직이는 객체를 위한 응용들은 이동객체 모델링[1,2]과 움직이는 객체들에 대한 질의 처리[5,7,8]를 지원하는 시공간 데이타베이스 관리 시스템을 요구한다. 이동객체 모델링에서는 시간의 흐름에 따라 위치를 연속적으로 변경하는 동적 객체를 소개한다[2]. 시공간 질의 처리를 위한 많은 연구들이 이 모델링을 기반으로 한다.

본 논문이 TPR-tree[7]를 위한 비용모델을 제시한 것이기 때문에, 우리는 R-tree 기반의 TPR-tree를 자세히 소개한다. 그림 1은 공간 색인 구조의 최소 경계 사각형(MBR)에 해당되는 TPR-tree의 시간 경계 간격(time-parameterized bounding interval)과 그 시간 경계 간격에 의해 감싸진 3개의 1차원 객체들(o_1~o_3)을 보인다. 시간 경계 간격 τ는 TPR-tree의 가장 중요한 개념이며, 그림 1(a)에서와 같이 하나의 공간 간격 $[x^l,x^h]$과 하나의 속도 간격 $[v^l,v^h]$으로 구성되어 있다. 간격(interval)의 최소값과 최대값을 나타내기 위해서 l과 h를 사용한다. 시간 경계 간격의 공간 간격은 색인 생성 시간 t_l에서의 값으로 표현된다. 화살표의 기울기는 속도를 나타낸다. 그림 1(a)에서와 같이, 객체들은 굵은 점선의 직선 안에서 시간의 흐름에 따라 움직인다. 또한, 객체가 갱신될 때, 시간 경계 간격을 최소화하는 방법이 제안되었다[7]. 그림 1(b)와 같이, 현재 시간 t_c에 o_3의 갱신된 속도로 인해서, 속도 간격의 v^l은 이전 상태의 o_3의 속도를 대신하여 새롭게 갱신된 o_3의 속도로 변경된다.

그림 2는 TPR-tree를 생성하기 위해서 구성된 시간 경계 간격들을 설명한다. 그림 2(a), 2(b), 그리고 2(c)는 각각 13개의 움직이는 객체, 그 객체들을 감싸는 단계(level) 1에서의 5개 노드, 그리고 단계 1의 노드들을 감싸는 단계 2의 2개 노드를 보인다. 그리고, 그림 2(d)는 그 대응되는 TPR-tree를 나타낸다.

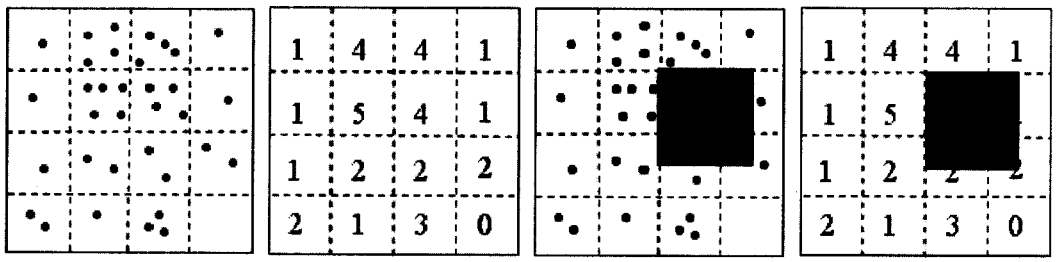
본 논문에서는 TPR-tree의 비용 모델을 제시하였다. 그러나, TPR-tree가 R-tree 기반이기 때문에, R-tree의 기존 비용 모델이 기본적으로 이용된다. Kamel과 Faloutsos는 R-tree를 사용한 영역 질의에 대한 평균 디스크 액세스 수를 예측하는 분석적인 방법을 제안하였다[13]. 그 방법은 질의가 공간 상에서 균등하게 분포한다고 가정하여, R-tree 성능 예측을 위한 질의와 노드 크기 사이의 효과를 제시하였다. Theodoridis 등은



(a) 움직이는 객체 (b) 단계 1(또는 단말)의 노드들

(c) 단계 2의 노드들 (d) TPR-tree

그림 2 TPR-tree를 구성하는 시간 경계 간격



(a) (b) (c) (d)

그림 3 공간 히스토그램

R-tree를 사용하여 영역 질의에 대한 평균 디스크 액세스 수를 정확하게 예측하기 위한 더욱 정교한 분석을 제안하였다[15]. Theodoridis 등은 편중된 공간 데이타의 분포를 다루기 위해서 공간 히스토그램을 이용하였다. 사용된 공간 히스토그램은 공간 데이타의 분포를 잘 묘사하는 단순한 격자(grid) 구조이다. 공간 히스토그램을 이용하여 공간 질의에 대한 데이타의 수를 예측하는 방법을 알아보자. 그림 3(a)는 34개의 공간 객체를 나타내며, 그림 3(b)는 대응되는 4×4 격자의 공간 히스토그램을 나타낸다. 그림 3(c)에서 알 수 있듯이, 공간 영역의 질의 Q안에 존재하는 객체 수는 6개이다. 공간 히스토그램을 이용하여 동일한 질의 영역 Q안에 속하는 객체 수를 구하는 방식은 질의 Q와 공간 히스토그램 셀들의 겹치는 비율을 계산하는 아주 단순한 방식이다. 그림 3(d)와 같이, 1개의 셀은 질의에 완전히 포함되며, 3개의 셀들은 일부분만 겹친다. 따라서, 4×1 + 1×0.5 + 2×0.5 + 2×0.25 = 6개의 객체가 질의 Q안에 있다고 예측한다.

3. TPR-tree를 위한 통계치

본 장에서는 TPR-tree를 위한 통계치와 그 통계치의 유지를 위한 간단한 전략을 설명한다. 표 1은 본 논문에서 전반적으로 사용된 기호들을 나타낸다. 2차원 공간상에서 움직이는 객체들에 대한 비용 모델은 1차원 공간상에서 움직이는 객체들에 대한 비용 모델의 단순한 확장 형태이기 때문에, 본 논문에서는 1차원 공간 상에서 움직이는 객체들에 대한 비용 모델 위주로 설명한다. 기호 사용의 편의를 위해서, x_1, v_1, a_1, s_1 대신 각각 x, v, a, s를 사용한다.

표 1 기호 설명

기 호	설 명
$I[I^l,I^h]$	간격: $I^l \le I^h$; I^l, I의 낮은 값; I^h, I의 높은 값
$r\,(x_1,x_2,v_1,v_2)$	2차원 이동객체를 위한 TPR-tree의 시간 경계 사각형: x_1,x_2, 공간 간격들; v_1,v_2, 속도 간격들
t_c	현재 시간
t_l	TPR-tree 생성 시간
$Q(a_1,a_2,t)$	시공간 질의: a_1,a_2, 공간 간격들; t, 시간 간격
N_l	TPR-tree의 단계 l의 노드들의 수
$S_l(s_1,s_2)$	시간 t_c에서 TPR-tree의 단계 l의 노드들로부터 공간 간격들의 평균 범위(extent)
$Vl(v_1,v_2)$	TPR-tree의 단계 l의 노드들로부터 평균 속도 간격

그림 4는 시간 경계 간격 τ와 시공간 질의 Q 사이의 관계를 나타낸다. Q는 하나의 공간 간격 $[a^l,a^h]$과 하나의 시간 간격 $[t^l,t^h]$로 구성된다. Q의 시간 간격의 최소값 t^l은 현재 시간 t_c보다 크거나 같아야 한다. 본 논문의 목적은 그림 4에서와 같이 τ와 Q 사이의 교차(intersect)하는 경우의 수를 예측하는 것이다. 그러나, 기존의 공간 비용 모델들은 움직이는 객체들의 미래 위치를 고려하지 않는다. 그래서, 그림 4과 같은 τ와 Q 사이의 교차하는 경우의 수를 효과적으로 예측할 수 없다.

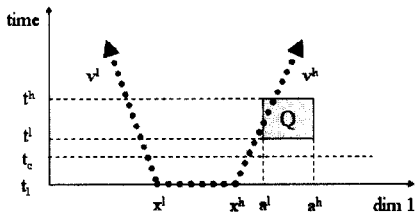


그림 4 시간 경계 간격과 시공간 질의

본 논문에서 제안하는 TPR-tree를 위한 통계치는 각 단계 l에 따른 N_l, V_l, S_l로 구성된다. N_l과 V_l은 각각 TPR-tree의 단계 l 노드 수와 노드들의 평균 속도 간격을 의미한다. 그리고, S_l은 t_c에서 단계 l 노드들의 평균 공간 간격 길이들을 의미한다. TPR-tree를 위한 통계치 S_l은 공간 데이타베이스에서 널리 알려진 R-tree의 단계 l의 노드들에 해당하는 MBR들의 각 차원의 평균 길이들에 해당된다. $SL(t_c,\tau)$은 $(\tau.x^h+\tau.v^h(t_c-t_l))-(\tau.x^l+\tau.v^l(t_c-t_l))$이며, S_l의 하나의 범위(extent) s는 $\frac{\sum_{i=1}^{N_l} SL(t_c,\tau)}{N_l}$이다. 그림 5는 현재 시간 t_c에서의 시간 경계 간격을 위한 SL을 나타낸다. 본 논문에서는 적은 비용으로 TPR-tree를 묘사하는 통계치를 유지할 수 있는 간단하고 실용적인 방법을 제안한다. 기본적인 전략은 노드들이 액세스되었을 때, 동적으로 N_l, V_l, S_l을 갱신하는 것이다. 이러한 방법은 액세스된 노드로부터 N_l과 V_l을 정확하게 유지한다. 그러나, S_l은 정확하게 유지되기 어렵다. 그 이유는 S_l은 현재 시간으로 정의되기 때문에, 현재 시간에 액세스되지 않는 노드들의

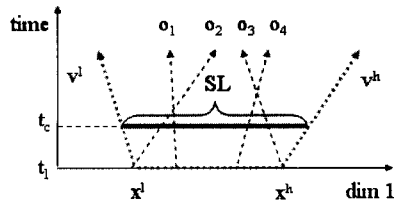


그림 5 t_c에서의 τ에 대한 SL(t_c, τ)

SL을 알기 어렵다. 따라서, S_l 대신 대략적인 평균 공간 간격 S_l^a을 유지하는 방법을 제안한다.

예를 들어, 객체의 갱신으로 인한 색인의 노드 액세스를 고려해 보자. 단말 노드의 단계는 1이고, 루트 노드의 단계는 h이다. 단계 l에 액세스된 노드로부터 단계 l-1의 시간 경계 간격들의 정보를 얻을 수 있다. 즉, 상위 노드가 하위 노드들에 대한 엔트리 정보를 가지고 있기 때문에, 하위 노드들의 시간 경계 간격들의 정보를 이용할 수 있다. 이것은 액세스 되지 않은 많은 노드들의 SL이 이용될 수 있음을 의미한다.

단말 노드를 제외한 단계 l에서 하나의 노드가 액세스 되었을 때, 그 노드를 위한 4가지 가능한 액세스들이 존재한다. 엔트리의 수정 없는 단순한 노드 액세스, 단계 l-1의 수정된 노드에 의한 수정된 엔트리를 갖는 노드 액세스, 단계 l-1의 노드 분할로 인한 삽입된 엔트리를 갖는 노드 액세스, 단계 l-1의 노드 병합으로 인한 삭제된 엔트리를 갖는 노드 액세스이다. 다음 식들은 단말 노드를 제외한 단계 l의 노드가 t_c에 액세스 되었을 때, S_{l-1}^a을 유지하는 방법을 설명한다. 노드 분할과 노드 합병에 의한 유도식도 비슷한 형태를 따르며, V_{l-1}과 N_{l-1}의 값은 쉽게 유지될 수 있다. $V_{l-1}\cdot v^h$는 $V_{l-1}\cdot v^l$과 유사하기 유지된다.

■ 수정 없는 노드 액세스

$$\cdot\ S_{l-1}^a.s \leftarrow \frac{(N_{l-1}-|\aleph|)S_{l-1}^a.s+\sum_{i=1}^{|\aleph|}SL(t_c,\aleph.\tau_i)}{N_{l-1}}$$

$|\aleph|$는 액세스된 노드 $\aleph(\tau_1,\ldots,\tau_n)$의 엔트리 수

■ 하나의 수정된 엔트리를 갖는 노드 액세스

$$\cdot\ V_{l-1}.v^l \leftarrow \frac{N_{l-1}V_{l-1}.v^l-\alpha.v^l+\beta.v^l}{N_{l-1}}$$

$$\cdot\ S_{l-1}^a.s \leftarrow \frac{(N_{l-1}-1)S_{l-1}^a.s+SL(t_c,\beta)}{N_{l-1}}$$

α는 수정 전의 엔트리, β는 수정 후의 엔트리

■ 하나의 삭제된 엔트리를 갖는 노드 액세스

$$\cdot\ V_{l-1}.v^l \leftarrow \frac{N_{l-1}V_{l-1}.v^l-\alpha.v^l}{N_{l-1}-1}$$

$$\cdot\ N_{l-1} \leftarrow N_{l-1}-1$$

α는 삭제 전의 엔트리

■ 하나의 삽입된 엔트리를 갖는 노드 액세스

$$\cdot\ V_{l-1}.v^l \leftarrow \frac{N_{l-1}V_{l-1}.v^l-\alpha.v^l+\beta.v^l+\gamma.v^l}{N_{l-1}+1}$$

$$\cdot\ S_{l-1}^a.s \leftarrow \frac{(N_{l-1}-1)S_{l-1}^a.s+SL(t_c,\beta)+SL(t_c,\gamma)}{N_{l-1}+1}$$

• $N_{l-1} \leftarrow N_{l-1} + 1$

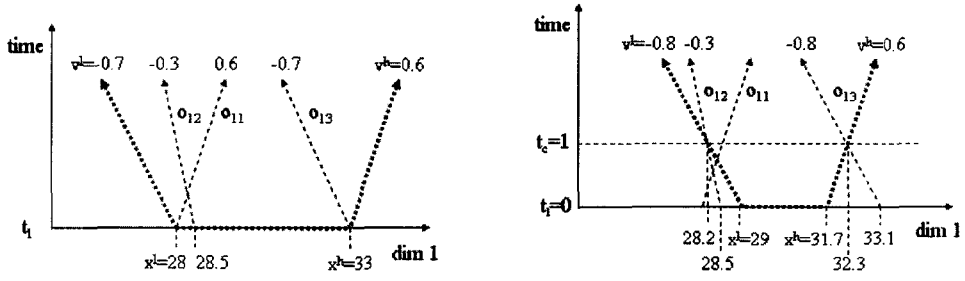
α는 overflow 전의 엔트리, α는 overflow후의 수정된 엔트리, γ는 overflow후의 새롭게 삽입된 엔트리

시공간 통계치의 갱신을 위한 간단한 예를 고려하자. 표 2는 그림 2(b)의 5개 단말 노드에 대한 정보를 나타낸다고 가정하자(단, t_l=0). 객체 o_{13}의 공간 위치는 33이고 그 객체의 속도는 -0.7이다. 현재 시간 t_c=0 때의 노드 7의 SL은 $(\tau_7.x^h + \tau_7.v^h(t_c - t_l)) - (\tau_7.x^l + \tau_7.v^l(t_c - t_l)) = 33 - 28 = 5$이다. 그리고, 그림 6과 같이, 시간 t_c=1에 속도 -0.8로 갱신된 객체 o_{13}으로 의해서 노드 7이 변경된다고 가정하자. τ_7의 v^l은 -0.8이 된다. 시간 t_c=0에 객체 o_{12}로부터 공간 위치 28.2(=28.5-0.3*(1-0))는 쉽게 계산되며, τ_7의 x^l은 28.2+0.8*(1-0)=29가 된다. 유사하게, x^h도 쉽게 계산된다.

이제, 단계 1에서의 통계치들이 유지되는 과정을 알아보자. 이전에 설명하였듯이, 단계 1에서의 통계치의 갱신들은 단계 2의 노드가 방문될 때에 이루어진다. 객체 o_{13}의 갱신을 처리하기 위해서, o_{13}과 관련된 단계 2의 노드 2는 두 번 방문된다(그림 2(d) 참조). 첫 번째 방문은 갱신 전 상태의 o_{13}의 찾기 위한 방문이고, 두 번째 방문은 갱신된 o_{13}으로 인해 변경된 τ_7을 수정하기 위한 방문이다. 표 3은 t_c=1에 단계 2의 노드 2에 대한 두 번의 액세스에 의한 N_l, $S_l^a \cdot s$, $V_l \cdot v^l$, $V_l \cdot v^h$의 변화를 나타낸다. 시간 t_l=0일 때의 $S_l^a \cdot s$는 $\frac{2+3+4+3+5}{5} = 3.4$이다. 시간 t_c=1에서 첫 번째의 단순한 노드 액세스(또는, 수정 없는 노드 액세스)를 위해서, $SL(t_c, \tau_6) = (\tau_6.x^h + \tau_6.v^h(t_c - t_l)) - (\tau_6.x^l + \tau_6.v^l(t_c - t_l)) = (25+0.7*(1-0)) - (22-0.5*(1-0)) = 4.2$이고

표 2 단계 1의 노드들 (또는 단말 노드들)

τ_3		τ_4		τ_5			τ_6			τ_7		
x^l:5		x^l:9		x^l:15			x^l:22			x^l:28		
x^h:7		x^h:12		x^h:19			x^h:25			x^h:33		
v^l:-0.3		v^l:-0.5		v^l:-0.6			v^l:-0.5			v^l:-0.7		
v^h:0.5		v^h:0.9		v^h:0.6			v^h:0.7			v^h:0.6		
SL:2		SL:3		SL:4			SL:3			SL:5		
o_1	o_2	o_3	o_4	o_5	o_6	o_7	o_8	o_9	o_{10}	o_{11}	o_{12}	o_{13}
5	7	9	12	15	17.8	19	22	23.2	25	28	28.5	33
0.5	-0.3	0.9	-0.5	0.6	-0.2	-0.6	0.7	-0.5	0.2	0.6	-0.3	-0.7



(a) 변경 전 (b) 변경 후

그림 6 갱신된 객체 o_{13}으로 의한 노드 7의 변경

표 3 N_l, $S_l^a \cdot s$, $V_l \cdot v^l$, $V_l \cdot v^h$의 변화

t_l=0	t_c=0
$N_1 = 5$	$N_1 = 5$
$S_1^a.s = \frac{2+3+4+3+5}{5} = 3.4$	$S_1^a.s = \frac{(5-1)*4.14+4.1}{5} = 4.132$
$V_1.v^l = \frac{-0.2-0.5-0.6-0.5-0.7}{5} = -0.5$	$V_1.v^l = \frac{5*(-0.5)-(-0.7)+(-0.8)}{5} = -0.52$
$V_1.v^h = \frac{0.5+0.9+0.6+0.7+0.6}{5} = 0.66$	$V_1.v^h = \frac{5*(0.66)-(0.6)+(0.6)}{5} = 0.66$

SL (t_c, τ_7) = $(\tau_7.x^h + \tau_7.v^h(t_c-t_l)) - (\tau_7.x^l + \tau_7.v^l(t_c-t_l))$ = (33+0.6*(1-0)) - (28-0.7*(1-0)) = 6.3이기 때문에, $S_l^a \cdot s$는 $\frac{(5-2)*3.4+4.2+6.3}{5}$ =4.14이다. 노드 2가 다시 액세스될 때, 즉 하나의 수정된 엔트리를 갖는 노드 액세스로 인해서 노드 2의 엔트리 τ_7은 변경된다. 변경된 τ_7을 위한 SL(t_c, τ_7) = (31.7+0.6*(1-0)) - (29-0.8*(1-0)) = 4.1이다. 최종적으로, $S_l^a \cdot s$는 $\frac{(5-1)*4.14+4.1}{5} = 4.132$이 된다.

4. TPR-tree 비용 모델

TPR-tree의 노드 수를 예측하는 방법의 이해를 돕기 위하여, 우선 R-tree의 노드 수를 예측하기 위한 중요한 개념을 설명한다. 그림 7은 공간 질의 Q와 R-tree의 어떤 단계의 노드들의 평균 MBR을 보인다. 이 MBR은 각 차원의 s_1과 s_2의 평균 길이로 이루어져 있다. 그림 7과 같이, 공간 질의 Q와 평균 MBR 사이의 교차되는 확률을 구하기 위해서, Q로부터 각 차원 별로 $\frac{s_1}{2}$와 $\frac{s_2}{2}$만큼 확장된 점선으로 이루어진 사각형이 이용된다 [14]. TPR-tree의 어떤 단계에서의 평균 시간 경계 사각형과 시공간 질의 사이의 교차되는 노드 수를 예측하는 방식은 유사하며 추가적으로 속도 간격을 고려하면 된다.

이제, 시공간 통계치를 이용하여 시공간 질의를 대한 TPR-tree의 노드 수를 예측하는 방법을 설명한다. 시공간 통계치로부터 단계 l의 S_l^a, V_l로 한정된 평균 시간 경계 간격을 고려해보자. 그림 8은 평균 시간 경계 간격과 시공간 질의와의 겹침을 설명한다. 시공간 질의와 겹치는 노드의 수를 예측하기 위하여 CQ와 ECQ를 이용한다. 노드 수는 시공간 질의를 지나는 객체와 지나지 않지만 질의 주변에 위치한 객체들 수에 의해서 결정된

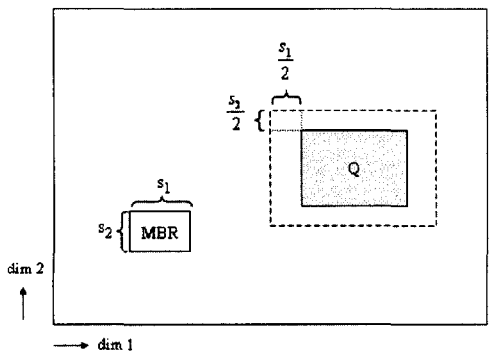


그림 7 공간 질의와 평균 MBR

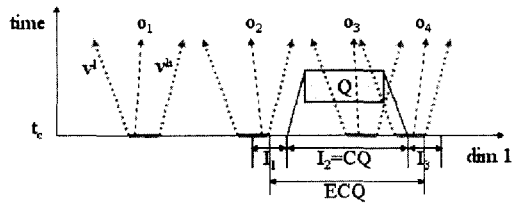


그림 8 CQ와 ECQ

다. 그림 8에서와 같이, CQ는 평균 시간 경계 간격과 질의가 만나기 시작하는 지점으로부터 쉽게 유도된다. CQ는 다음과 같이 정의된다.

정의 1. t_c를 현재 시간, v를 속도 간격, Q를 시공간 질의라고 하자. v에 의해 한정된 시간 경계 간격을 위해, CQ는 다음과 같은 조건을 만족하는 시간 t_c에서의 최대 공간 간격 I이다. 시간 t_c에서의 I의 부분 간격(sub-interval)을 포함하는 공간 간격을 가진 모든 가능한 시간 경계 간격은 Q와 겹쳐야 한다.

예를 들어, 하나의 질의와 속도 간격 $v[v^l, v^h]$에 의해 한정된 시간 경계 간격을 고려해보자(단 $v^l<0$, $v^h>0$이다). 그림 8과 같이, 시간 t_c에서의 I_2가 CQ를 나타낸다. 정의 1을 위한 CQ$(t_c, v, Q(a,t))$는 다음과 같은 하나의 간격 I를 생성한다.

- $I^l = \begin{cases} a^l - (t^l - t_c)v^h & \text{if } v^h < 0 \\ a^l - (t^h - t_c)v^h & \text{otherwise} \end{cases}$
- $I^h = \begin{cases} a^h - (t^l - t_c)v^l & \text{if } 0 < v^l \\ a^h - (t^h - t_c)v^l & \text{otherwise} \end{cases}$

시간 t_c에 CQ를 지나는 객체를 감싸는 평균 시간 경계 간격들은 항상 질의와 겹친다. 그러나, I_1 또는 I_3를 지나는 객체를 감싸는 평균 시간 경계 간격들은 질의와 겹칠 수도 있고 겹치지 않을 수도 있다. I_1과 I_3의 길이는 평균 시간 경계 간격의 길이이다. 그래서, CQ로부터 확장된 ECQ를 지나는 객체를 감싸는 시간 경계 간격들이 Q와 겹친다고 간주한다. 결국, ECQ를 지나는 객체의 수로부터 노드의 수를 예측하게 된다. ECQ는 CQ로부터 $\frac{S_l^a}{2}$만큼 양쪽으로 확장된 간격을 나타낸다. 이와 같이 TPR-tree를 묘사하는 시공간 통계치는 Theodoridis 등[15]이 제시하였던 방식에 쉽게 적용된다.

그림 9는 2차원의 움직이는 객체들을 위한 TPR-tree의 비용 모델을 위한 알고리즘을 나타낸다. 1차원 움직이는 객체를 위한 TPR-tree의 시간 경계 간격은 2차원 객체를 위한 TPR-tree의 시간 경계 사각형(time-parameterized bounding rectangle: TPBR)에 해당된다. 1차원에서의 시간 경계 간격과 시공간 질의 사이의 CQ를 구하는 방법은 2차원에서의 시간 경계 사각형과

시공간 질의 사이의 2차원 CQ를 구하는 방법으로 자연스럽게 확장된다. 그림 10은 2차원으로 확장된 형태를 보여준다. 1차원 시간 경계 간격을 위한 CQ와 ECQ를 대신하여, CQ_2와 ECQ_2는 2차원 시간 경계 사각형을 위한 것이다. DiskAccess 알고리즘은 TPR-tree를 위한 시공간 통계치와 Theodoridis 등[15]에서 사용한 편중된 공간 분포를 위한 공간 히스토그램을 이용하여 디스크 액세스 수 DA를 예측한다. Theodoridis 등[15]에서 제시한 방법과 같이, 우리는 단순한 격자(grid) 형태의 공간 히스토그램을 움직이는 객체들의 현재 위치로부터 생성한다. 그림 9에서, 1줄은 TPR-tree의 루트 노드의 액세스를 의미한다. 2줄에서는 객체 수 N과 TPR-tree의 평균 fanout f를 이용한 TPR-tree의 높이(height) h를 구한다. 3-8줄의 반복문에서, 단계 1부터 단계 h-1까지의 디스크 액세스 수를 합한다. 4-5줄은 2차원 객체들을 위한 시간 t_c에서의 공간 영역 R을 생성한다. 그리고, 7줄에서는 TPR-tree의 단계 l에서 질의를 만족하는 노드 수를 예측한다.

Algorithm DiskAccess

1. DA ← 1 /* access of root node */
2. $h \leftarrow 1 + \lceil \log_f \frac{N}{f} \rceil$ /* height h of the TPR-tree */
3. for $l \leftarrow 1$ to h-1 do {/* access for each level */
4. $R \leftarrow CQ_2(t_c, V_l, Q)$
5. $R \leftarrow ECQ_2(S_l^a, R)$
6. let n be the estimated number of objects in R using the spatial histogram
7. $DA \leftarrow DA + \frac{n}{f^l}$ /* the estimated number of nodes */
8. }
9. return DA /* return the total number of disk accesses */

그림 9 DiskAccess 알고리즘

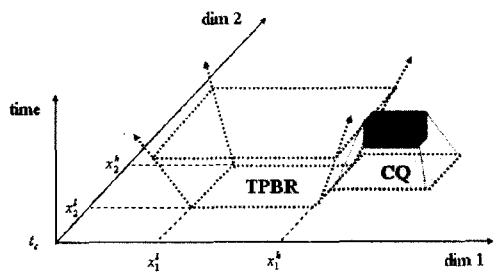


그림 10 2차원으로 확장

5. 실 험

5.1 실험 환경

본 논문의 실험 환경은 기존의 방식들[4,7]과 유사하다. 공간 데이타베이스 분야에서 널리 사용하고 있는 실세계 데이타인 Tiger/lines 데이타[17]의 캘리포니아 지역의 도로 데이타를 이용하여 움직이는 객체들의 초기 공간 위치를 표현하였다. 그림 11과 같이, 1000(1000의 2차원 공간상에의 편중된 공간 분포를 나타내는 객체 수는 약 37만개이다.

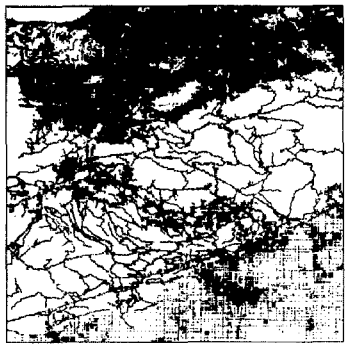


그림 11 초기 공간 위치

이러한 객체들은 1000(1000의 2차원 공간상에서만 움직이며, 객체들의 최대 속도는 단위 시간당 3.0으로 객체들의 움직이는 방향은 무작위로 선택되게 설정하였다. 움직이는 객체들은 시간의 흐름에 따라 매우 빈번하게 갱신되는 특징이 있기 때문에, 주기적으로 현재 상태의 위치들을 파악하기 위해서 샘플링을 기반의 공간 히스토그램이 필요하다. 히스토그램을 전체 데이타로부터 생성하지 않고, 전체 데이타의 샘플 데이타로부터 생성하는 방식이 제안되었다[19]. 편중된(skewed) 공간 분포를 위한 공간 히스토그램을 위해서 40×40 셀의 사용하였다[15].

다양한 시공간 질의를 위해서, 공간 영역(QS)는 전체 데이타 공간의 0.25%, 0.5%, 1%, 2%, 4%가 고려되었으며, 공간 영역의 위치는 데이타 공간에서 임으로 선택되었다. 질의의 시간 간격의 길이는 0, 10, 20, 30, 40이 고려되었다.

5.2 실험 결과

그림 12는 약 10,000개의 질의로부터 평가된 실험 결과를 나타낸다. 질의의 시간 간격과 공간 영역에 관한 상대적인 평균 오차를 나타낸 것이다. 그림 12는 Tiger/lines 데이타를 사용한 움직이는 객체의 실험 결과를 보인다. 상대적인 평균 오차는 14%에서 32%사이이다. 일반적인 실험 결과[16]와 마찬가지로, 질의의 공간 영역

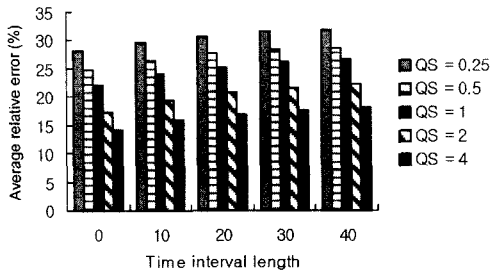


그림 12 질의의 시간 간격과 공간 영역에 따른 상대적 평균 오차 (Tiger/Lines 데이타)

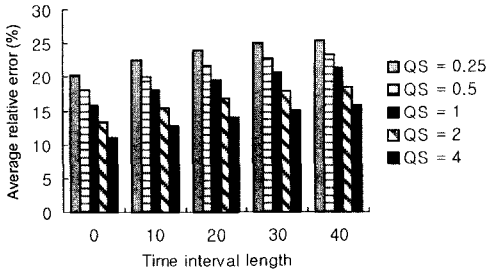
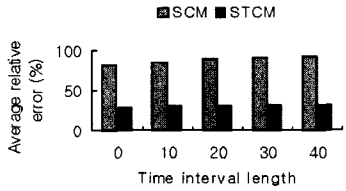
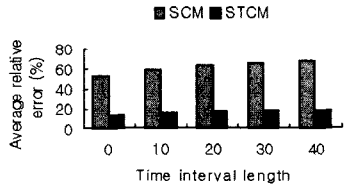


그림 14 질의의 시간 간격과 공간 영역에 따른 상대적 평균 오차 (Sequoia 데이타)

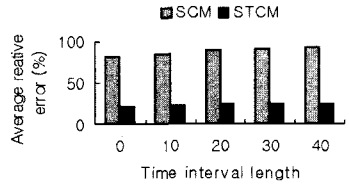


(a) QS 크기 : 0.25%

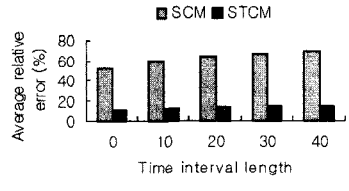


(b) QS 크기 : 4%

그림 13 질의의 시간 간격에 따른 SCM과 STCM의 상대적 평균 오차 (Tiger/Lines 데이타)



(a) QS 크기 : 0.25%



(b) QS 크기 : 4%

그림 15 질의의 시간 간격에 따른 SCM과 STCM의 상대적 평균 오차 (Sequoia 데이타)

크기가 증가할수록 상대적인 평균 오차는 감소한다.

그림 13은 시간 간격에 관한 상대적인 평균 오차를 나타낸 것이다. TPR-tree를 사용한 시공간 질의에 대한 비용 모델 연구가 없기 때문에, 기존의 공간 비용 모델(SCM)과 제안된 시공간 비용 모델(STCM)을 비교하였다. 우리는 기존 공간 비용 모델(SCM)을 위해서 Theodoridis 등[15]이 제안하였던 비용 모델을 사용하였다. 기존의 공간 비용 모델을 TPR-tree에 적용시킬 때, 시공간 질의의 공간 영역만 고려하고 시간 간격은 현재 시간으로만 고려한다. 그림 13(a)는 QS 0.25%가 적용된 질의들의 시간 간격별로 SCM과 STCM의 실험 결과를 나타낸다. 예상했던 것과 같이, 움직이는 객체의 미래 위치를 고려하지 않는 SCM은 높은 오차율을 보였다. 그림 13(b)는 QS 4%의 실험 결과를 나타낸다. 그림 13에서 알 수 있듯이, QS가 0.25%일 때는 QS가 4%일 때와 비교하여 더욱 큰 오차율을 보인다. QS가 작을수록 더욱 큰 오차율을 보이는 일반적인 실험 결과를 따른다[16].

같은 방식으로, 우리는 움직이는 객체들을 생성하기 위해서 Sequoia 데이타[18]를 사용한 현실적인 실험 환경을 만들었다. 그림 14는 Sequoia 데이타를 사용한 움직이는 객체들에 대한 유사한 실험 결과를 보인다. 다른 시간 간격의 길이를 위한 공간 영역의 크기에 관한 상대적인 평균 오차를 보인다.

추가적으로, 두 개의 QS를 위한 시간 간격의 길이에 대해서 SCM과 STCM 사이의 상대적인 평균 오차를 평가하였다. 그림 15(a)를 위해서, QS는 0.25%로 설정되었다. 기대되었듯이, STCM은 SCM에 비해 정확한 예측 결과를 보인다. 그림 15(b)는 QS 4%가 적용된 시간 간격 길이에 대한 실험 결과를 보인다.

6. 결 론

TPR-tree는 움직이는 객체들의 미래 위치를 빠르게 검색하기 위해서 가장 널리 사용되는 색인이다. 본 논문에서는 TPR-tree를 사용한 시공간 질의를 위한 디스크 액세스 수를 예측하는 비용 모델을 제안하였다. 객체의

움직임을 정확하게 예측할 수 있는 분석적인 방법이 사용되었다. 현실적인 실험 환경을 위해서, 적절히 편중된 분포를 갖는 실세계 공간 데이타를 사용하여 움직이는 객체를 생성하였다. 이러한 움직이는 객체에 대한 실험에서, 제안된 방법은 다양한 질의에 대해서 정확한 예측 결과를 보였다. 제안된 방법은 TPR-tree의 비용 모델에 관한 첫 번째 연구이다.

참 고 문 헌

[1] L. Forlizzi, R.H. Guting, E. Nardelli, and M. Schneider, *"A Data Model and Data Structures for Moving Objects Databases,"* ACM SIGMOD, 2000.

[2] A.P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao, *"Modeling and Querying Moving Objects,"* ICDE, 1997.

[3] P.K. Agarwal, L. Arge, and J. Erickson, *"Indexing Moving Points,"* PODS, 2000.

[4] G. Kollios, D. Gunopulos, and V.J. Tsotras, *"On Indexing Mobile Objects,"* PODS, 1999.

[5] D. Pfoser, C.S. Jensen, and Y. Theodoridis, *"Novel Approaches in Query Processing for Moving Object Trajectories,"* VLDB, 2000.

[6] S. Saltenis and C.S. Jensen, *"Indexing of Moving Objects for Location-Based Services,"* ICDE, 2002.

[7] S. Saltenis, C.S. Jensen, S.T. Leutenegger, and M.A. Lopez, *"Indexing the Positions of Continuously Moving Objects,"* ACM SIGMOD, 2000.

[8] Y. Tao and D. Papadias, *"MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries,"* VLDB, 2001.

[9] Y. Tao and D. Papadias, *"Time-Parameterized Queries in Spatio-Temporal Databases,"* ACM SIGMOD, 2002.

[10] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez, *"Cost and Imprecision in Modeling the Position of Moving Objects,"* ICDE, 1998.

[11] N. Beckmann, H.P. Kriegel, R. Schneider, and B. Seeger, *"The R*-Tree: An Efficient and Robust Access Method for Points and Rectangle,"* ACM SIGMOD, 1990.

[12] K. Porkaew, I. Lazaridis, and S. Mehrotra, *"Querying Mobile Objects in Spatio-Temporal Databases,"* SSTD, 2001.

[13] I. Kamel and C. Faloutsos, *"On Packing R-trees,"* CIKM, 1993.

[14] S.T. Leutenegger and M.A. Lopez, *"The Effect of Buffering on the Performance of R-Trees,"* TKDE, 2000.

[15] Y. Theodoridis, E. Stefanakis, and T.K. Sellis, *"Efficient Cost Models for Spatial Queries Using R-Trees,"* TKDE, Vol.12, No.1, pp. 19-32, 2000.

[16] S. Acharya, V. Poosala, and S. Ramaswamy, *"Selectivity Estimation in Spatial Databases,"* ACM SIGMOD, 1999.

[17] U. S. Bereau of Census, *"Tiger/lines precensus files: 1994 technical documentation,"* Technical report, 1994.

[18] M. Stonebraker, J. Frew, K. Gardels, and J. Meredith, *"The SEQUOIA 2000 Storage Benchmark,"* ACM SIGMOD, 1993.

[19] P.B. Gibbons, Y. Matias, and V. Poosala, *"Fast Incremental Maintenance of Approximate Histograms,"* VLDB, 1997.

최 용 진
정보과학회논문지 : 데이타베이스
제 31 권 제 1 호 참조

정 진 완
정보과학회논문지 : 데이타베이스
제 31 권 제 1 호 참조