한국지리정보학회지 6권 4호 : 59~70(2003)

밀도 기반의 k-최근접 질의 처리

장인성[1※] · 한은영[1] · 조대수[1]

A Density-based k-Nearest Neighbors Query Method

In-Sung JANG[1※] · Eun-Young HAN[1] · Dae-Soo CHO[1]

요 약

공간 데이터베이스 관리시스템에서 제공하는 공간 질의는 많은 디스크 참조와 CPU 처리시간을 필요로 한다. 이 중에서 k-최근접 질의는 많은 디스크 참조를 요구하는 질의로써 지금까지 많은 연구가 이루어져 왔다. 트리 구조의 색인을 사용하는 k-최근접 질의 처리방법은 조건을 만족하지 않는 노드를 가지치기 기법을 사용하여 노드 방문횟수를 줄인다. 그러나 이 방법은 가지치기 과정에서 불필요한 디스크 참고가 발생하여 성능을 저하시키는 단점을 가지고 있다. 본 논문에서는 가지치기 기법 대신 주어진 k개의 최근접 객체가 존재할 영역을 미리 예측함으로써 디스크 참조횟수를 줄이는 방법을 제시한다. 이 영역을 예측하기 위해서 본 연구에서는 데이터 분포에 대한 밀도를 이용하였다. 실험에 의하면 이러한 방법은 기존의 가지치기 기법을 이용한 방법에 비해서 최고 22%, 평균 7% 정도의 디스크 참조횟수의 감소 효과가 있음을 알 수 있다.

주요어: 공간 데이터베이스, 지리정보시스템, 최근접

ABSTRACT

Spatial database system provides many query types and most of them are required frequent disk I/O and much CPU time. k-NN search is to find k-th closest object from the query point and up to now, several k-NN search methods have been proposed. Among these, MINMAX distance method has an aim not to access unnecessary node by adapting pruning technique. But this method accesses more disks than necessary while pruning unnecessary nodes. In this paper, we propose new k-NN search algorithm based on density of object. With this method, we predict the radius to be expected to contain k-NN objects using density of data set and search those objects within this radius and then adjust radius if failed. Experimental results show that this method outperforms the previous MINMAX distance method. This algorithm visit less disks than MINMAX method by the factor of maximum 22% and average 7%.

2003년 10월 25일 접수 Received on October 25, 2003 / 2003년 12월 12일 심사완료 Accepted on December 12, 2003

1 한국전자통신연구원 공간정보기술센터

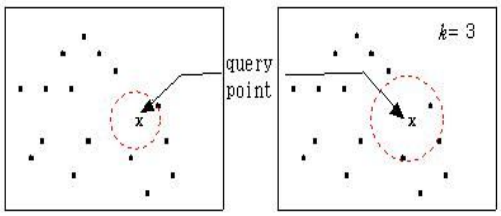
Spatial Information Technology Center, Electronics and Telecommunications Research Institute

※ 연락저자 E-mail: e4dol2@etri.re.kr

KEYWORDS: Spatial Database, GIS, Nearest

서 론

공간 데이터베이스 관리시스템에서 다루는 공간 데이터는 그 양이 방대할 뿐만 아니라 형태가 복잡하여 질의 처리시 빈번한 디스크 입출력을 필요로 한다. 공간 데이터베이스 관리시스템에서 제공하는 질의는 점질의, 영역질의, 조인질의, 최근접 질의 등 매우 다양하다. 이 중에서 최근접 질의는 질의 점에서 가장 가까운 공간 객체를 찾는 질의(그림 1a)로써 이것을 일반화시킨 것이 k-최근접 질의이다. 이는 주어진 질의 점에서 가장 가까운 k개의 객체를 찾는 질의(그림 1b)로써 지금까지 이에 대한 연구는 많이 행해졌다(Beyer 등, 1999; Hjaltsson과 Samet, 1999; Hinneburg 등, 2000; Berchtold 등, 2001).



(a) 최근접 질의 (b) 3-최근접 질의

FIGURE 1. The sample of k-nearest query

최근접 탐색을 수행하기 위해서는 모든 공간 객체와 거리를 계산하여야 하는데, 방대한 공간 데이터를 다루는 공간 데이터베이스에서는 이는 바람직하지 않다. 이를 고려하여, 지금까지 다양한 공간 색인 방법들과 공간 색인을 이용하여 불필요한 공간 객체는 비교하지 않는 여러 질의 처리 기법들이 제안되었다. 기존에 많이 사용하여 온 최근접 탐색 알고리듬은 Rousopoulos 등(1995)이 제안한 분기 및 한정 방법(이하 가지치기 방법, pruning method)이다. 그러나 이 방법은 MINMAX 거리를 이용하므로 필요한 노드보다 더 많은 노드를 방문하기 때문에 최적의 방법이라고는 할 수 없다. 이 문제를 피하기 위해 본 논문에서는, 트리-기반 공간 색인 기법을 이용하면서 객체의 분포 상태를 고려하여 k개의 최근접 객체가 존재할 영역을 미리 예측하고, 이를 질의 영역으로 이용하는 새로운 최근접 탐색 방법을 제안한다.

가지치기(Pruning) 방법을 이용한 k - 최근접 탐색

여기서는 k-최근접 탐색을 위해 제안된 방법으로써, 다차원 색인을 이용하는 방법 중 대표적인 것인 R*-tree를 이용한 가지치기 방법(MINMAX pruning method)(Rousopoulos 등 1995; Cheung 등 1998)과 이 방법의 문제점을 살펴본다.

Rousopoulos 등(1995)과 Cheung 등(1998)에서는 다차원 색인 기법의 일종인 R*-tree에 대하여 가지치기를 적용한 탐색 알고리듬이 제안되었다. 이 알고리듬은 R*-tree에서 질의 점이 주어졌을 때 질의 점과 최소 경계 사각형의 거리를 이용하여 최근접 질의를 처리하는 방법을 이용한다. 즉 주어진 질의 점과 하나의 최소 경계 사각형 사이의 최소 거리를 MIN DIST라고 정의하고, 노드를 방문할 때 MIN DIST가 가장 작은 노드를 방문한다. 그러나 MINDIST는 그림 2와 같이, 질의 점에서 노드의 최소 경계 사각형과 가장 가까이 있어도 노드 안에 실제 객체와의 최소 근접성은 보장하지 못한다.

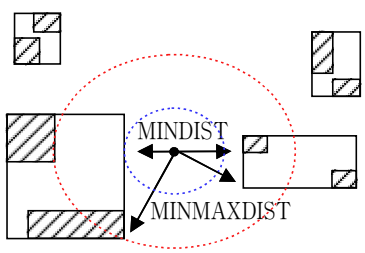


FIGURE 2. Pruning method

따라서 실제 객체와의 최소 근접성을 보장하기 위하여 최소 거리를 주어진 질의 점에서 최소 거리에 있는 최소 경계 사각형의 두 번째 가까운 꼭지점까지의 거리를 MINMAXDIST라고 정의한 다음, 후보 노드를 방문할 때 MINDIST와 MINMAXDIST의 성질을 이용하여 방문할 후보 노드들을 그림 2와 같이 가지치기하여 나아감으로써 최근접 질의를 처리한다.

이 가지치기 방법은 다음과 같은 두 가지 중요한 문제를 가지고 있다. 첫번째 문제는 탐색 영역을 최소화하지 못한다는 것이다. 가지치기 방법에서 성능을 높이는 가장 중요한 방법은 탐색되는 영역을 결정하는 반지름인 MINMAXDIST를 가능한 적게 만드는 것이다. 즉, 가능한 탐색되는 질의 영역을 최소로 하여 노드의 방문을 줄이는 것이 매우 중요하다. 그러나 가지치기 방법에 의하면, 그림 3과 같이 노드의 최소 경계 사각형의 모양이나 위치에 따라 최적의 반지름을 얻지 못하여, 불필요한 노드를 참조하고 결국 성능이 저하될 수 있다.

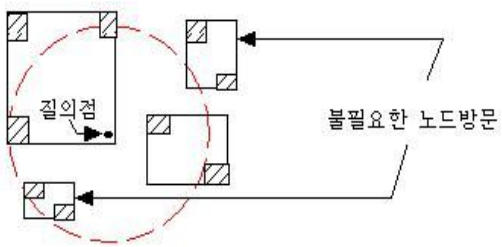


FIGURE 3. Unnecessary access

예를 들어 그림 4와 같이 질의 점 q 에 대해 최근접 객체 탐색 시, 기존의 가지치기 방법을 이용하면 트리의 방문 중 M1, M11, M12, M2, M21 순서로 5번의 디스크 참조를 해야 한다. 뿐만 아니라 참조되는 모든 노드와 실제 거리를 계산해야 한다. 이는 불필요한 노드 참조와 CPU 계산을 포함하여 성능을 저하시킨다는 것을 의미한다.

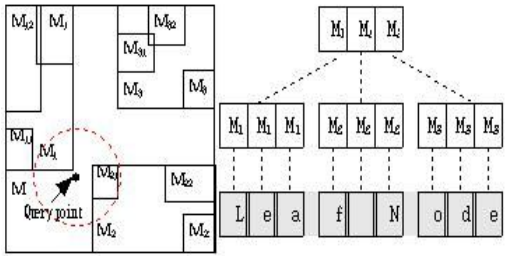


FIGURE 4. The sample of accessing unnecessary node

두번째 문제점은 탐색되는 영역이 가지치기 과정 중에 동적으로 결정된다는 단점을 가지고 있다. 즉 공간 데이터베이스 시스템을 병렬화할 때, 동적으로 결정되는 탐색영역은 부하의 분산을 결정하는 작업을 어렵게 한다.

밀도를 이용한 k-최근접 탐색 알고리듬

1. 밀도와 최근접 탐색 공간

가지치기 방법에서 이용하는 탐색공간은 객체의 분포와 많은 관계가 있다. 객체가 조밀하게 분포된 그림 5의 질의 점(a) 지역에는 탐색공간은 좁아지게 되며, 객체가 별로 없는 그림 5의 질의 점(b) 지역은 탐색공간이 넓어지게 된다. 본 논문에서는 이와 같은 특성을 고려하여 탐색지역을 결정하는데 이용되는 반지름을 질의지역의 밀도를 이용하여 추정하는 방법을 제안한다. 그림 5는 7-최근접 질의시 질의지역의 밀도와 탐색공간과의 관계를 보여준다.

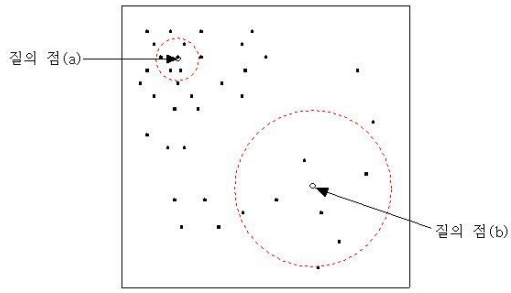


FIGURE 5. Relation density with area

밀도를 이용한 질의 처리 방법에 사용되는 기본 개념과 관련하여 우선 사용되어지는 기호와 함수에 대하여 정리를 하면 다음과 같다.

TABLE 1. Symbol and function definition

기 호	정 의
d	차원
r	반지름
q	d 차원의 점, $(q_1, q_2, \ldots q_d)$
R_d	d 차원에서 다차원 사각형과 내접하는 다차원 구의 부피비
$A(q)$	초기 선택율 추정을 위해 사용된 정다면체의 영역
$A_R(q, r)$	i-번째 차원이 $[q_i - r, q_i + r)$ 길이를 가지는 다차원 사각형의 영역
$A_S(q, r)$	중심점이 q이며 반지름이 r인 다차원 구의 영역
$\hat{N}(q)$	초기 선택율 추정에 의해 계산된 공간 객체 수
$\hat{N}(A)$	A 영역 내에 추정되어지는 객체의 수(객체 빈도)
$N(A)$	A 영역 내에 실제 객체의 수
$V(A)$	A 영역의 부피
$S_d(r)$	반지름이 r인 d-차원 구의 부피
$P_{den}(q)$	q 지역에 대한 밀도
$R_{den}(q, r)$	$A_R(q, r)$의 밀도
$S_{den}(q, r)$	$A_S(q, r)$의 밀도

가. 다차원 구의 부피

본 논문에서 제안하는 방법을 이해하기 위해서는 다차원 상의 구의 부피의 개념이 중요하다. 반지름이 r인 d-차원 구의 부피, $S_d(r)$은 다음과 같이 구할 수 있다(Faloutos과 Kamel, 1994).

$$S_d(r) = \int_{-r}^{r} S_{d-1}(f(x))dx, \quad d > 2$$

여기서 $S_1(r) = 2r$, $f(x) = \sqrt{r^2 - x^2}$ 이다.

이것을 일반화하면 다음과 같다.

$$S_d(r) = \frac{\sqrt{\pi^d}}{\Gamma(\frac{d}{2} + 1)} \cdot r^d$$

여기서

$\Gamma(x+1) = x \cdot \Gamma(x)$, $\Gamma(\frac{1}{2}) = \sqrt{\pi}$, $\Gamma(1) = 1$ 이다. 따라서,

$S_2(r) = \pi r^2$, $S_3(r) = \frac{4}{3}\pi r^3$,

$S_4(r) = \frac{\pi^2}{2} r^4$, $S_5(r) = \frac{8}{15}\pi^2 r^5, \cdots$이다.

그리고, 감마 함수($\Gamma(x)$)는 다음 식으로 단순화 할 수 있다(Berchtold 등, 1997).

$$\Gamma(x) \approx (x/e)^x \cdot \sqrt{2 \cdot \pi \cdot x}$$

나. 차원 구와 외접하는 다차원 사각형의 부피비

반지름이 r인 다차원 구에 외접하는 다차원 사각형의 한 변의 길이는 $2r$이다. 그러므로 이 다차원 사각형의 부피는 $(2r)^d$ 이고, 다차원 구의 부피는 $S_d(r)$이다. 따라서 다차원 구와 외접하는 다차원 사각형의 부피비(R_d)는 다음과 같다.

$$R_d = \frac{V(A_S(q, r))}{V(A_R(q, r))} = \frac{\sqrt{\pi^d}}{\Gamma(\frac{d}{2} + 1) \cdot 2^d} \quad (1)$$

2. 밀도 추정 방법

밀도를 효과적으로 구하기 위해 다음과 같은 방법을 이용하였다. 먼저 데이터 분포의 통계적인 정보를 이용해서 밀도 분산 히스토그램(density histogram)을 만든다. 즉 d차원의 작업 공간이 있을 때, 각 차원을 동일한 간격으로 나누어 히스토그램을 만든다. 그 다음, 해당 격자에 속하는 객체의 수를 저장하고 이것을 밀도 분산 히스토그램이라고 정한다.

밀도는 점에 대한 밀도와 영역에 대한 밀도로 나눌 수 있다. 먼저, 점에 대한 밀도는 질의 점을 포함하는 단위격자 내에 포함된 공간 객체의 수를 의미한다. 즉, d차원의 점 q에 대한 밀도는 다음과 같다.

점 q에 대한 밀도,

$$P_{den}(q)= \frac{N(q)}{V(A(q))}$$

여기서 $V(A(q))$는 q를 포함하는 단위 공간의 부피이고, $N(q)$는 단위 공간 안에 있는 객체의 수이다. $V(A(q))$는 밀도 테이블을 생성할 때 나눈 격자의 부피이므로 값을 알 수 있다. $N(q)$는 q를 포함하는 격자 내의 객체의 수이므로 밀도 테이블을 이용해서 구할 수 있다. 따라서 점에 대한 밀도를 쉽게 구할 수 있다.

다음으로, 영역에 대한 밀도는 다차원 사각형 영역에 대한 밀도와 다차원 구 영역에 대한 밀도로 구분 할 수 있다. 먼저 다차원 사각형 영역에 대한 밀도는 다음과 같다. 여기서 다차원 사각형 영역은 다차원 점(q)과 거리(r)이 주어졌을 때, i-번째 차원에 대해 $[q_i - r,\ q_i + r)$의 간격을 가지는 영역의 $A_R(q,r)$ 부피는 $V(A_R(q,r))$이고, $\widehat{N}(A_R(q,r))$는 $A_R(q,r)$내에 있다고 추정되어지는 객체의 수이다.

따라서 다차원 사각형 영역에 대한 밀도,

$$R_{den}(q,r)= \frac{\widehat{N}(A_R(q,r))}{V(A_R(q,r))}$$ 이다.

$\widehat{N}(A_R(q,r))$
여기서 는 다음의 식으로 구할 수 있다.

$\widehat{N}(A_R(q,r))=$

$$\sum\left(\frac{\text{격자와 겹치는 질의영역의부피}}{\text{격자의 부피}} \times \text{격자안의 객체수}\right)$$

또한 다차원의 구 영역에 대한 밀도는

$$S_{den}(q,r)= \frac{\widehat{N}(A_s(q,r))}{S_d(r)} = \frac{\widehat{N}(A_s(q,r))}{V(A_s(q,r))}$$

더욱 정확한 밀도를 구하기 위해서는 다차원 구 영역에 대한 밀도를 구하여야 하지만, 다차원 구 영역에 대한 밀도를 구하는 것은 계산 과정, 특히 $\widehat{N}(A_s(q,r))$구하는 것이 복잡하고 최근접 질의 처리시 많은 CPU 처리 시간을 요구하게 된다. 따라서 본 논문에서는 여러 개의 격자로 나누었을 때 하나의 격자 내의 분포는 균등 분포를 따른다고 가정한다. 따라서 다차원 사각형 영역에 대한 밀도를 구한 다음, 그 밀도 값을 다차원 구 영역에 대한 밀도 값으로 간주한다.

그러므로 $S_{den}(q,r) \approx R_{den}(q,r)$이다.

그러면 q를 중심으로 반지름이 r인 구의 내부에 있다고 추정, 객체의 수를 다음과 같이 유도할 수 있다.

$$S_{den}(q,r)= \frac{\widehat{N}(A_S(q,r))}{V(A_S(q,r))} \approx$$

$$R_{den}(q,r)= \frac{\widehat{N}(A_R(q,r))}{V(A_R(q,r))}$$ 이므로

$$\widehat{N}(A_S(q,r)) \approx$$

$$\widehat{N}(A_R(q,r)) \times \frac{V(A_S(q,r))}{V(A_R(q,r))}$$ 이다.

여기서 식 (1)에 의하면,

$$\frac{V(A_S(q,r))}{V(A_R(q,r))} = R_d$$ 이므로

$$\widehat{N}(A_S(q,r)) \approx \widehat{N}(A_R(q,r)) \times R_d \quad (2)$$

단, $R_d = \dfrac{\sqrt{\pi^d}}{\Gamma(\frac{d}{2}+1)\cdot 2^d}$

$\widehat{N}(A_R(q,r)) =$

$\sum($ 격자와 겹치는 질의영역의 부피 / 격자의 부피 $\times$ 격자안의 객체수) 로 계산된다.

3. 탐색 알고리듬

식 (2)를 이용하면 $\widehat{N}(A_S(q,r)) = k$ 일 때의 반지름 r을 추정할 수 있다. 추정된 반지름 r을 이용하여 k-최근접 질의 방법을 설명하고자 한다.

본 논문에서 제안하는 최근접 객체의 질의 처리 방법은 크게 다음과 같은 네 단계의 작업으로 구성된다.

단계 1. 질의 지역에 대한 밀도의 추정
단계 2. 밀도를 이용한 탐색 영역 반지름 추정
단계 3. 단계 2에서 구한 탐색영역으로 k-최근접 질의 처리
단계 4. 단계 3에서 k개의 객체가 찾아지지 않았을 경우, 탐색영역의 확대

[단계 1] 질의 점 주위의 밀도 추정

위의 방법을 적용하기 위해서는 우선 주어진 질의 점의 밀도를 추정하여야 한다. 이 문제는 선택율 추정결과를 이용하여 계산될 수 있다. 본 논문에서는 선택율을 추정방법 중 가장 단순한 밀도 분산 히스토그램을 이용하는 방식을 이용한다. 질의 영역 주위의 밀도는 다음과 같이 계산된다.

$$P_{den}(q) = \frac{N(q)}{A(q)}$$

$A(q)$는 선택율 추정을 위해서 사용된 영역, 즉 q를 포함하는 단위 격자의 부피이고, $N(q)$는 히스토그램의 격자 안에 객체의 수이다.

[단계 2] 탐색 반지름 추정

단계 1에서 구한 점에 대한 밀도를 이용해 초기 반경(r_0)을 추정한다.

질의 점 q를 중심으로 주위(거리 r 이내)의 분포가 균등하다고 가정하면,

$P_{den}(q) = S_{den}(q,r)$ 이다.

따라서 $P_{den}(q) = \frac{\widehat{N}(A_S(q,r))}{S_d(r)}$

이 때 $\widehat{N}(A_S(q,r)) = k$ 인 반지름 r을 구하므로,

$$P_{den}(q) = \frac{k}{S_d(r)} \text{ 이다.} \quad (3)$$

따라서, 초기 반지름 r_0은 다음과 같이 추정되어진다.

- $P_{den}(q) \neq 0$ 일 때, 식 (3)을 이용하면,

  $r_0 = S_d^{-1}(\frac{k}{p_{den}(q)})$ 이다.

- $P_{den}(q) = 0$ 이면

초기 선택율 추정을 위해 사용된 정다면체의 영역인 $A(q)$가 히스토그램의 격자영역과 동일하다. $\widehat{N}(A_S(q,r)) = 0$인 경우에는 초기 반경을 격자크기의 반으로 정하고자 한다.

따라서 격자의 부피,

$V(A(q)) = (2r_0)^d$ 이므로,

$$r_0 = \frac{\sqrt[d]{V(A(q))}}{2}$$

그런데 여기서 구한 r_0은 처음에 질의 점 (q)을 중심으로 균등 분포를 따른다고 가정한 것이므로, q를 중심으로 균일 분포가 아니면 r_0 값은 보정되어야 한다. 이 과정을 정리하면 다음과 같다.

단계 2-1. $P_{den}(q)$을 이용하여 r_0을 추정

$r_0 = S_d^{-1}(\frac{k}{p_{den}(q)})$ 또는 $\frac{\sqrt[d]{V(A(q))}}{2}$

단계 2-2. 밀도 분산 히스토그램을 이용하여 $A_s(q, r_0)$의 선택율을 추정, 즉 $\widehat{N}(A_s(q, r_0))$를 추정한다.

단계 2-3. 만일 $|\widehat{N}(A_s(q, r_0)) - k| > \varepsilon$ 이면,

$\alpha = \sqrt[d]{\frac{k}{\widehat{N}(A_S(q, r_0))}}$, $r_0' \leftarrow r_0 \times \alpha$

로 하여 단계 2-2를 반복

(단, $\alpha > \lambda$ 또는 $\widehat{N}(A_S(q, r_0)) = 0$ 이면 $\alpha = \lambda$)

[단계 3] k-최근접 질의 수행

색인을 이용하여 위에서 구한 r_0를 반지름으로 하는 d-차원의 구와 겹치는 영역에 대해 최근접 질의를 수행한다.

[단계 4] 탐색영역 확대

만일 단계 3의 탐색 결과로 k개 이상의 객체가 찾아지면 이중에서 가까운 k개를 선택하여 최근접 탐색을 완료한다. 그러나, 만일 탐색 결과로 찾아진 객체의 수가 k개보다 적으면 단계 4의 과정을 거쳐야 한다. 그런데 단계 4를 반복하게 되면 질의 처리 성능이 매우 심각하게 저하된다. 따라서 단계 4의 과정이 필요하게 되는 확률을 가장 최소로 하면서 동시에 반지름을 가능한 작게 만들기 위하여 δ 만큼 r을 증가시킨다.

$r' = r + \delta$,

$\delta = 2r(\sqrt[d]{\frac{k}{N(A_S(q, r))}} - 1)$

(단, $\delta > Threshold$이면 $\delta = Threshold$)

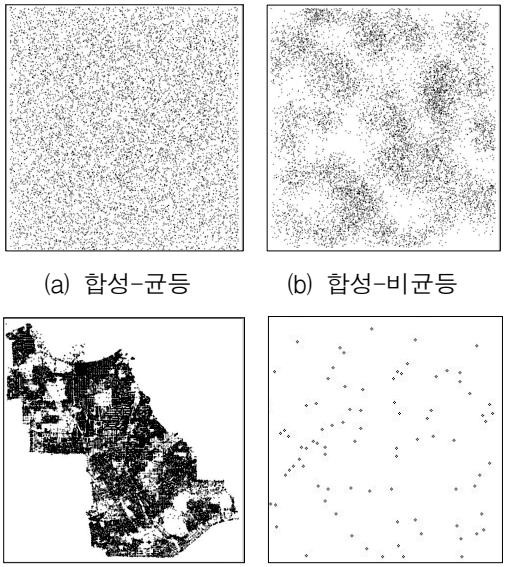
이상과 같이 밀도를 이용하여 k-최근접 질의를 수행하는 새로운 기법을 소개하였다. 밀도 분산 히스토그램이라는 데이터 분포에 대한 통계적 정보를 이용하므로 가능한 최적의 탐색영역을 결정한다. 뿐만 아니라 질의영역이 미리 결정되기 때문에 병렬화 등에 용이하게 사용되어질 수 있고, 기존의 많은 방법들이 k-최근접 객체를 모두 찾은 다음에 결과를 돌려주는 데 비해 제안된 방법은 질의 수행 중에 결과 값의 일부를 제공할 수 있다는 장점이 있다. 또한 제안된 방법은 영역 질의와 최소 비용거리를 지원하는 모든 색인구조에 적용될 수 있다는 것이 특징이다.

실험 및 성능 평가

이 실험에서는 지금까지 트리구조 공간 색인 방법 중 성능이 우수한 것 중의 하나로 알려진 R*-tree(Beckmann 등, 1990)를 사용하였다. 또한 선택율 추정을 위하여 다차원 히스토그램을 이용하였으며, 가지치기 방법은 Cheung과 Fu(1998)에서 제안된 방법을 구현하였다.

1. 2차원 데이터에 대한 실험

실험에 사용된 데이터는 2차원의 점 객체만을 대상으로 하였다. 10,000개의 점으로 이루어진 균등한 합성 데이터(그림 6a)와 비균등한 합성 데이터(그림 6b), 그리고 공간 데이터베이스 관리시스템의 벤치 마크를 위한 데이터로 자주 사용되는 36,548개의 점으로 이루어진 롱비치 지역의 실제 데이터(그림 6c)를 대상으로 실험하였다. k-최근접 질의에서 k값은 1, 20, 40, 60, 80과 100으로 하였으며, 질의는 100개의 임의로 생성된 2차원 점 데이터(그림 6d)로 최근접 질의를 수행하였다. 실험에 사용된 데이터 분포는 그림 6과 같다. 성능 평가는 디스크 참조횟수를 기준으로 하였다.



(a) 합성-균등 (b) 합성-비균등

(c) 실제-롱비치지역 (d) 질의 데이터

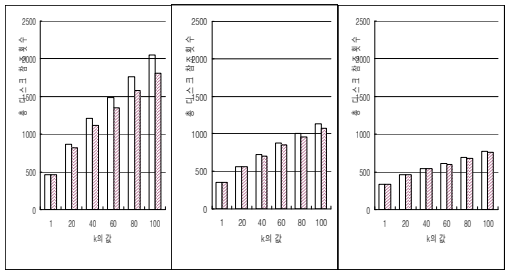
FIGURE 6. 2D dataset and query dataset

R*-tree 구현시 페이지 크기는 1K, 2K, 4K 바이트로 하였고, 각 색인의 정보를 간략히 정리하면 표 2와 같다.

TABLE 2. 2-dimension index information

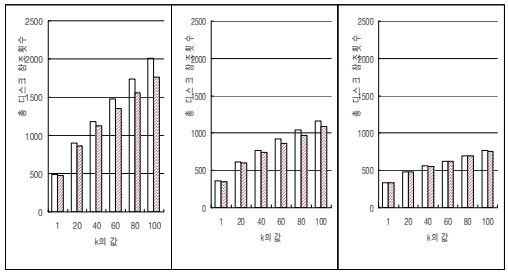
		노드 크기	최대 분기율	높이	비단말 노드수	단말 노드수	최상위노드 저장이용률 (%)	비단말노드 저장이용률 (%)
합성데이터	균등 (10,000)	1K	28	4	41	733	7.8	68.5
		2K	56	3	10	357	16.2	70.1
		4K	112	3	3	181	1.9	79.7
	비균등 (10,000)	1K	28	4	37	712	7.8	73.6
		2K	56	3	10	355	16.2	69.7
		4K	112	3	3	183	1.9	80.6
실제 데이터 (36548)		1K	28	4	181	2,951	28.9	61.7
		2K	56	3	38	1,440	65.4	68.8
		4K	7,112	3	11	730	8.9	64.3

그림 7, 8 및 그림 9는 위의 세 가지 데이터에 대한 실험 결과를 나타낸다. 이 그림에서 세로축은 질의 100개에 대한 총 디스크 참조 횟수를, 가로축은 k(1, 20, 40, 60, 80, 100)의 값을 의미한다. 이 그림의 (a), (b)와 (c)는 각 1K, 2K, 4K 바이트의 페이지 크기를 말한다. 또한 그래프의 막대에서 왼쪽은 가지치기 방법이고, 오른쪽은 본 논문에서 제안한 밀도를 이용한 방법이다.



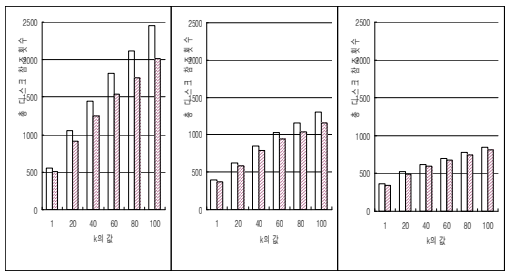
(a) Pagesize-1K (b) Pagesize-2K (c) Pagesize-4K

FIGURE 7. Synthetic uniform data



(a) Pagesize-1K (b) Pagesize-2K (c) Pagesize-4K

FIGURE 8. Synthetic skew data



(a) Pagesize-1K (b) Pagesize-2K (c) Pagesize-4K

FIGURE 9. Long Beach real data

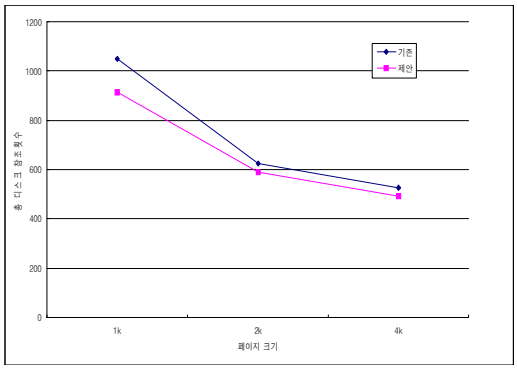


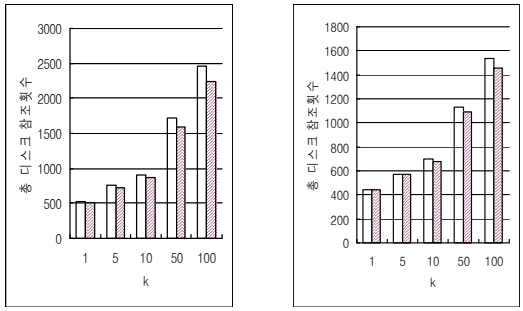
FIGURE 10. Performance by page size

결과는 연구에서 제안한 방법이 기존의 가지치기 방법에 비하여 평균 7%, 최고 22%의 성능향상을 보여주고 있으며, 밀도를 고려함으로써 데이터의 분포에 상관없이 성능이 향상된 것을 알 수 있다. 페이지의 크기가 4K바이트일 때는, 최대 분기율이 112이고, 평균 분기율이 72~89 정도인데, k가 작을수록 질의 점에서 가까운 k개의 객체가 하나의 노드 안에 포함될 확률이 높아진다. 따라서 대략 R*-tree의 높이만큼 디스크 참조를 하기 때문에 성능 향상을 기대할 수는 없었다. 그림 10에서와 같이 세가지 데이터 모두 페이지 크기가 작은 경우, 즉 분기율이 작은 경우 노드의 수가 많고, R*-tree의 높이가 높아 성능 향상이 더욱 크다는 것을 알 수 있었다.

2. 다차원 데이터에 대한 실험

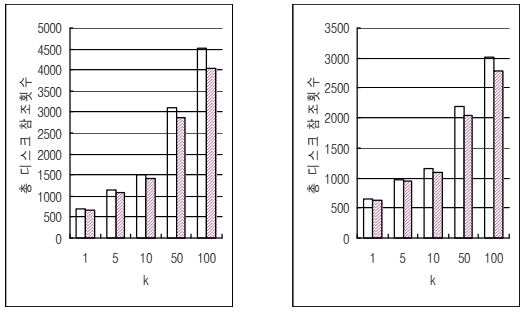
실험에 사용된 데이터는 3, 4, 6, 8, 10차원의 각각 10,000개의 임의로 생성한 점 객체를 대상으로 성능 평가를 하였고, R*-tree 구현 시 페이지 크기는 2K, 4K 바이트로 하였다.

k는 1, 5, 10, 50과 100의 값을 사용하였고, 질의는 100개의 임의로 생성된 해당 차원의 점으로 수행하였다. 또한 그래프의 막대에서 왼쪽은 가지치기 방법이고, 오른쪽은 본 논문에서 제안한 밀도를 이용한 방법이다(그림11~그림15). 각 차원에 대해 생성된 R*-tree의 정보는 표 3과 같다.



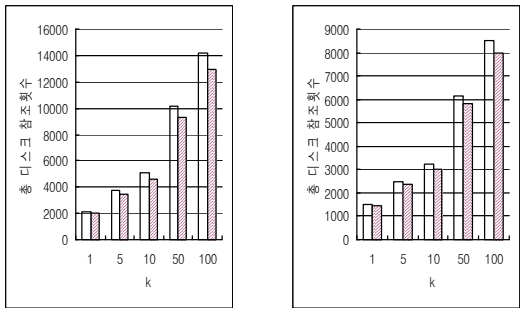
(a) Pagesize- 2K (b) Pagesize- 4K

FIGURE 11. 3-Dimension



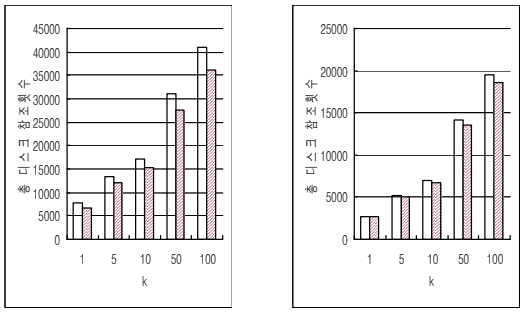
(a) Pagesize- 2K (b) Pagesize- 4K

FIGURE 12. 4-Dimension



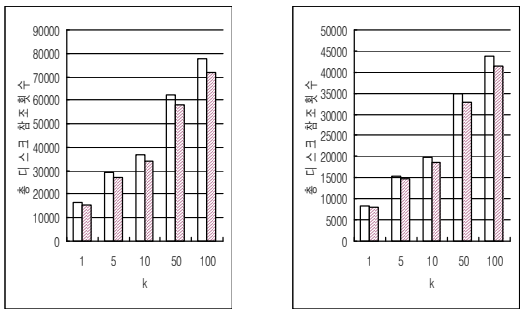
(a) Pagesize- 2K (b) Pagesize- 4K

FIGURE 13. 6-Dimension



(a) Pagesize- 2K (b) Pagesize- 4K

FIGURE 14, 8-Dimension



(a) Pagesize- 2K (b) Pagesize- 4K

FIGURE 15. 10-Dimension

TABLE 3. Multi-dimension index information

차원	노드 크기	최대 분기율	높이	비단말 노드수	단말 노드수	최상위노드 저장이용률 (%)	비단말노드 저장이용률 (%)
3	2K	39	3	18	462	43.5%	69.3%
	4K	78	3	5	231	5.2%	73.5%
4	2K	30	3	29	578	99.3%	68.9%
	4K	60	3	9	290	13.4%	60.3%
6	2K	20	4	61	801	19.9%	70.1%
	4K	40	3	15	400	34.3%	69.9%
8	2K	15	4	105	1027	58.3%	69.9%
	4K	31	3	24	518	74.3%	72.7%
10	2K	12	5	162	1213	16.4%	68.6%
	4K	24	4	42	634	8.2%	65.9%

다차원 데이터에 대한 성능 평가에서도 기존의 가지치기 방법에 비해 제안된 방법이 평균 7% 정도의 성능 향상을 보였고, 2차원 데이터의 평가결과와 같이 페이지의 크기가 작을 때 더 좋은 성능을 보이는 것을 알 수 있다.

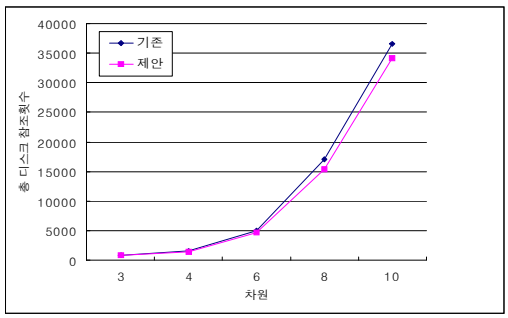
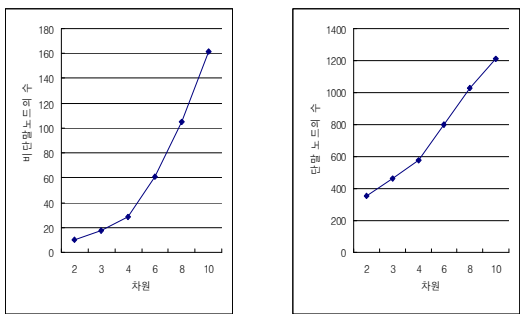


FIGURE 16. 10-NN disk acesses number (Page size: 2K)

또한 페이지의 크기가 2K바이트일 때, 10-최근접 질의에 대한 차원별 성능 평가를 실시하였는데, 그림 16과 같이 디스크 참조 횟수가 급격히 증가함을 확인할 수 있었고, 저차원보다는 고차원에서 상대적으로 좋은 성능을 보였다. 이것은 표 3과 그림 17에서처럼 차원이 증가할수록 분기율이 낮아지므로 비단말 노드의 수가 급격히 증가하고, 이들 노드간의 중복도 많이 생기므로 기존 방법의 문제점이였던 불필요한 노드 방문을 많이 하기 때문이라 할 수 있다.



(a) 비단말노드의 수 (b) 단말 노드의 수

FIGURE 17. Number of node by dimension (page size: 2K)

3. 성능 평가

2차원 및 다차원 데이터를 가지고 논문에서 제안한 알고리듬과 기존의 가지치기 방법

의 성능을 비교한 결과, 제안된 방법이 밀도 분산 히스토그램이라는 데이터 분포에 대한 통계적 정보를 이용하여 가능한 최적의 탐색 영역을 결정하기 때문에 데이터 분포에 상관없이 기존의 방법보다는 평균 7% 정도의 성능 향상을 보였다. 특히 페이지 크기가 작을수록, 또 차원이 높을수록, 즉 분기율이 낮을수록 상대적으로 더 좋은 성능을 보였다.

결론 및 향후 과제

k-최근접 질의는 주어진 질의 점에서 가장 가까운 k개의 객체를 찾는 질의로서, 공간 데이터베이스 관리시스템 뿐만 아니라 다양한 데이터베이스 관리시스템의 응용 프로그램에서 빈번히 요구되어지는 질의이다. 본 논문에서는 많은 디스크 참조를 요구하는 k-최근접 탐색을 위한 새로운 기법을 제안하였다. 제안된 방법은 객체분포에 대한 밀도를 이용하여 질의 조건을 만족하는 영역을 미리 예측함으로써 불필요한 노드 방문을 제거하는 것이다.

실험 결과, 제안된 방법이 데이터의 분포와 상관없이 최고 22%, 평균 7%의 디스크 참조 횟수 감소의 성능 향상을 보였다. 그리고 기존의 많은 방법들이 k-최근접 객체를 모두 찾은 다음에 결과를 돌려주지만, 제안된 방법은 질의 수행 중에 결과 값의 일부를 제공할 수 있다는 것과, 질의 처리 이전에 미리 탐색영역을 알 수 있어 병렬화에 효과적으로 활용될 수 있다는 것이 장점이다.

향후 과제로는 현재는 디스크 참조횟수만을 성능 평가의 기준으로 삼았으나 질의 수행 시 CPU 처리시간도 비교 분석해야 할 것이다. 그리고 제안된 방법에서는 다차원 히스토그램을 이용하여 구현하였으나 보다 정확한 선택율 추정방법을 적용하면 성능이 더욱 향상되리라 예상된다. 또한 기존에 k-최근접을 위해 제안된 색인 기법인 X-tree, SR-tree, A-tree, SCM 등에 제안된 방법을 이용한 성능 평가도 필요하다. KAGIS

참고문헌

Beckmann, N., H.P. Kriegel, R. Schneider and B. Seeger. 1990. The R*-tree, an efficient and robust access method for points and rectangles. Proceedings of the 1990 ACM SIGMOD international conference on management of data. Atlantic City, NJ, May 23-25, 1990. SIGMOD Record Volume 19(2), pp. 322-331.

Berchtold, S., C. Böphm, D. Keim and H.P. Kriegel. 1997. A cost model for nearest neighbor search in high-dimensional data space. Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. Tucson, Arizona, May. 12-14, 1997. ACM Press 1997. pp. 78-86.

Berchtold, S., C. Bohm, D. Keim, F. Krebs and H.-P. Kriegel. 2001. On optimizing nearest neighbor queries in high-dimensional data spaces. Proceeding of ICDT the 8th International Conference. London, UK, January. 4-6, 2001. pp. 435-449.

Beyer, K.S., J. Goldstein, R. Ramakrishnan, and U. Shaft. 1999. When is ´Nearest Neighbor´ meaningful?. Proceeding of ICDT the 7th International Conference. Jerusalem, Israel, January. 10-12, 1999. pp. 217-235.

Cheung, K. and A. Fu. 1998. Enhanced nearest neighbour search on the R*-tree. Proceedings of ACM SIGMOD. Record Volume 27(3), pp. 16-21.

Faloutsos, C. and I. Kamel. 1994. Beyond uniformity and independence, analysis of R*-trees using the concept of fractal dimension. Proceedings of the Thirteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. Minneapolis, Minnesota, May 24-26, pp. 4-13.

Hinneburg, A., C. Aggarwal and D. Keim. 2000. What is the nearest neighbor in high dimensional spaces?. Proceedings of the 26th International Conference on Very Large Data Bases. Cairo, Egypt, September. 10-14, 2000. pp123-129.

Hjaltason, G. R. and H. Samet. 1999. Distance browing in spatial database. ACM Transactions on Database System 24(2) : 265-318.

Roussopoulos, N., S. Kelley and F. Vincent. 1995. Nearest neighbor queries. Proceedings of the ACM SIGMOD International Conference. San Jose, California, May 22-25, 1995. ACM press 1995. pp. 71-79.