공간 데이터베이스의 효율적인 검색을 위한 X-트리와 kd-트리의 병합 알고리즘

유 장 우† · 신 영 진† · 정 순 기††

요 약

공간적인 자료구조를 기반으로 하는 공간 데이터베이스에서는 일차원 색인구조와는 달리 공간객체들의 다차원적인 특성에 부합되는 새로운 색인구조가 요구되고 있다. 본 논문에서는 이러한 요구사항을 충족시키기 위하여 기존 다차원 색인구조들의 특징 분석을 통하여 공간 데이터베이스의 효율적인 검색을 위한 새로운 색인구조를 제안하였다. 기존 X-트리에서 슈퍼노드의 순차적인 검색방법의 개선과 방대한 슈퍼노드가 생성되는 경우에도 검색시간의 단축이 가능하도록 하기 위하여, 포인트 색인구조를 갖는 kd-트리를 X-트리에 병합시킨 색인구조를 제안하였다. 제안된 색인구조를 실제로 구현하여 실험 데이터의 차원과 분포에 따라 검색시간을 분석하였다.

An Integration Algorithm of X-tree and kd-tree for Efficient Retrieval of Spatial Database

Jang-Woo Yoo† · Young-Jin Shin† · Soon-Key Jung††

ABSTRACT

In spatial database based on spatial data structures, instead of one-dimensional indexing structure, new indexing structure which corresponds to multi-dimensional features of spatial objects is required. In order to meet those requirements, in this paper we proposed new indexing structure for efficient retrieval of spatial database by carrying through the feature analysis of conventional multi-dimensional indexing structures. To improve the sequential search method of supernodes in the conventional X-tree and to reduce the retrieval time in case of generating the huge supernode, we proposed a indexing structure integrating the kd-tree based on point index structure into the X-tree. We implemented the proposed indexing structure and analyzed its retrieval time according to the dimension and distribution of experimental data.

1. 서 론

최근 지리정보 시스템, 이미지 데이터베이스 시스템 등과 같은 다양한 분야에서 공간 데이터들을 다루고 있다[1]. 공간 데이터는 다중 속성을 갖는 레코드로 모형화되어야 하며, 방대한 데이터 양과 데이터 객체간의 복잡한 관계성 때문에 다차원 특징벡터에 기반한 새로운 색인구조를 필요로 한다[2]. 다차원 데이터의 색인구조 설계시 고려해야 할 중요사항은 다음과 같다[3, 4]. 1) 검색시 다중경로 설정을 최소화하기 위해 객체간에 겹침영역을 최소화 해야한다. 2) 다양한 종류의 질의처리를 수용할 수 있어야 한다. 3) 데이터 특성에

* 본 연구는 1998년도 정보통신부의 정보통신 우수 시범학교 지원 사업에 의해 수행되었음.
† 준 회 원 : 충북대학교 대학원 컴퓨터공학과
†† 정 회 원 : 충북대학교 컴퓨터공학과 교수
논문접수 : 1999년 1월 26일, 심사완료 : 1999년 11월 12일

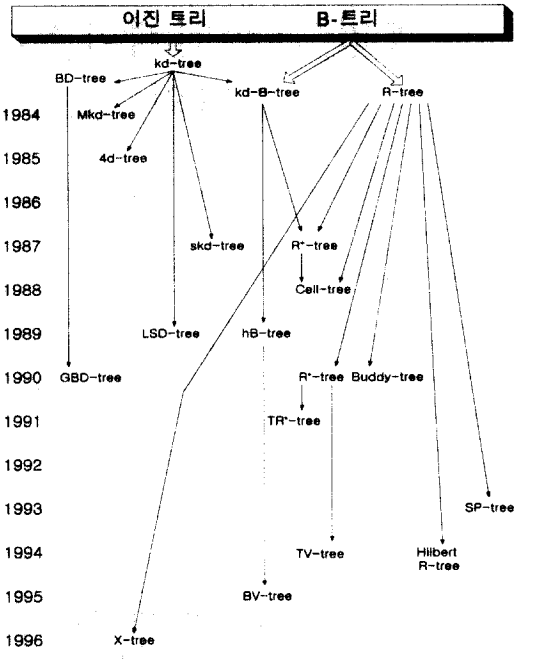
관계없이 저장공간을 효율적으로 활용할 수 있어야 한다. 4) 데이터의 분할, 병합을 효율적으로 처리할 수 있는 동적인 알고리즘이 제공되어야 한다.

본 논문에서는 공간 데이터의 색인구조로 사용되고 있는 X-트리에서 겹침영역 증가로 인한 검색성능 저하 문제를 개선하기 위해 X-트리와 kd-트리를 병합시킨 X-kd-트리를 제안하였다. 제안한 트리구조를 구현, 다른 트리구조들과의 성능을 비교, 분석하여 그 우수성을 보여 주었다. X-트리에서 최소 분할함수의 반환값이 실패하게되면 슈퍼노드가 생성되어 이로 인해 검색성능이 떨어지게 된다. 겹침영역의 증가로 인한 방대한 슈퍼노드가 생성될시 슈퍼노드의 검색성능을 제고시키기 위하여 포인트 색인구조를 갖는 kd-트리를 슈퍼노드에 첨가시키고, 슈퍼노드의 순차적인 검색 대신에 kd-트리에 의한 포인트검색을 수행함으로서 검색성능을 보다 효율적으로 개선할 수 있다.

본 논문의 2장에서는 공간 데이터의 색인구조로 사용되는 트리들의 특성을 분석하였고, 3장에서는 X-트리에서 슈퍼노드의 순차적인 검색방법을 개선할 수 있는 X-kd-트리 구조를 제안하였으며, 4장에서는 제안한 트리의 삽입 알고리즘 구현과 그 검색성능을 분석하였다. 5장에서는 결론 및 향후 연구과제에 대해 논하였다.

2. 공간 색인기법

공간 데이터베이스의 검색효율을 제고하기 위하여 다차원 공간 데이터들의 색인방법에 대한 많은 연구가 진행되어 왔으나[3] 트리 구조를 기반으로 하는 공간 색인기법은 (그림 1)과 같이 크게 두 가지로 분류할 수 있다. 이들 중 kd-트리, kd-B-트리, skd-트리[5]등은 포인트 색인구조를 갖는데, 이러한 색인기법들은 데이터 객체들이 다차원 공간상에서 포인트로 나타나게 된다. 따라서 R-트리[6]에서는 최소 경계 사각형들의 겹침 현상 등을 고려할 필요가 없으나, 모든 객체들이 다차원 공간상에서 포인트로만 표시되기 때문에 R-트리는 객체들의 교차관계나 포함관계와 같은 공간 연산에는 적합치 못하다. 또한 R-트리나 skd-트리와 같은 색인구조는 동일한 데이터 집합이더라도 데이터의 삽입 순서에 따라 상이한 트리를 형성하므로 동적인 삽입 알고리즘을 사용하여 트리의 성능이 데이터의 삽입 순서에 의존되지 않도록 해야한다.



(그림 1) 공간 색인기법의 발전과정

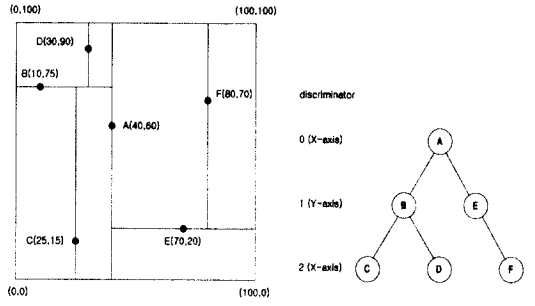
이진 트리와 B-트리를 기반으로 하는 고차원 데이터의 색인구조들로는 kd-트리[7], kd-B-트리[8], 쿼드트리[9], skd-트리[5], R-트리[6], TV-트리[10], BV-트리[11] 및 X-트리[12]등이 있으며, 이들 중에서 대표적인 트리들의 특성을 분석하면 다음과 같다.

2.1 kd-트리

k차원 이진검색 트리인 kd-트리[7]의 구조는 (그림 2)와 같다. 트리의 노드는 실제 데이터를 나타냄과 동시에 데이터의 검색 방향을 결정한다. 노드는 검색 방향을 결정하기 위하여 0부터 k-1까지의 값을 갖는 키를 사용한다. 따라서 2d-트리에서는 각 차원에 대한 키를 트리의 두 레벨마다 번갈아 가며 비교하게 되고, 3d-트리에서는 세 레벨마다 비교하게 된다[13].

노드의 삭제연산시 복잡도가 높게된다. 내부노드의 삭제시 노드들의 자리바꿈이 발생되며, 한 노드의 자리바꿈은 연속적인 자리바꿈을 일으킬 수가 있다. 또한 동일한 데이터 집합이더라도 입력순서에 따라서 상이한 트리구조를 형성하게 되므로 검색성능이 저하된다. 이러한 단점들을 극복하기 위하여 분할축의 결정과 트리의 중앙위치에 해당하는 근사 키값을 이용하는

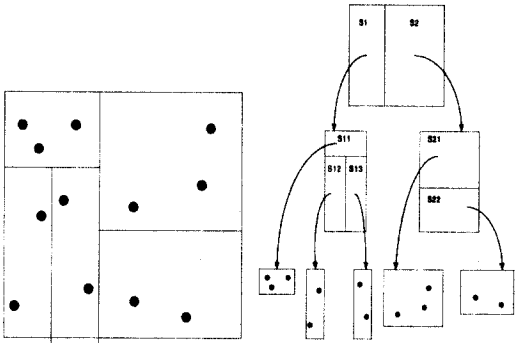
VAM kd-트리[14] 등이 제안되었다.



(그림 2) kd-트리의 구조

2.2 kd- B-트리

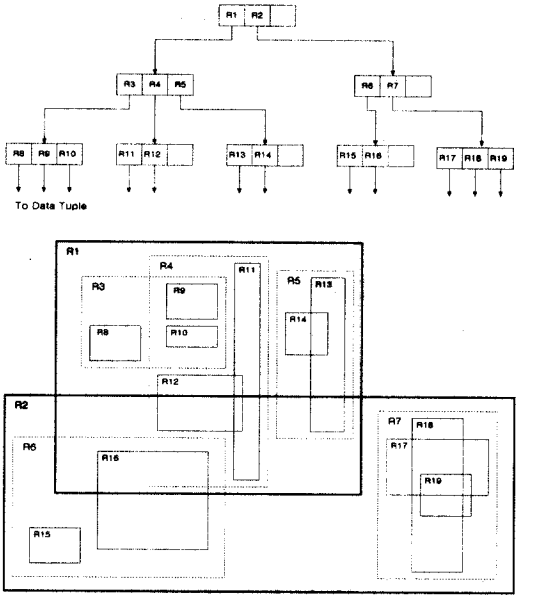
kd-트리와 B-트리의 조합인 kd-B-트리[8]는 (그림 3)과 같은 구조를 가지며, 객체 식별자를 포함하는 포인트 페이지와 하위 페이지에 대한 포인터를 저장하는 영역 페이지로 구성된다. 포화 상태인 포인트 페이지에 새로운 객체를 삽입시킬 때 페이지 분할이 일어나며, 이러한 경우 트리의 높이는 항상 균형을 유지토록 한다. 영역 페이지가 분할될 때는 두 페이지가 거의 동일한 수의 엔트리를 갖도록 한다. kd-B-트리의 노드 분할이 하향으로 파급시 언더플로우가 발생될 수 있으므로 저장공간의 이용 효율성이 저하된다는 단점이 있다.



(그림 3) kd-B-트리의 구조

2.3 R-트리

R-트리[6]의 구조는 (그림 4)와 같으며, k차원의 데이터 처리를 위한 B-트리의 확장으로서 공간 데이터 검색에 주로 사용된다. R-트리의 단말노드에는 데이터 객체에 대한 엔트리가 저장되고, 중간노드에는 하위노드의 엔트리를 포함하는 사각형 엔트리들로 구성된다. R-트리는 최소 경계 사각형들간의 포함과 겹침 관계가 최소일 때 가장 효율적이며, 이를 위하여 경계 사각형들의 겹침 관계를 개선한 R^+-트리[15]와 R^*-트리[16] 등이 제안되었다.



(그림 4) R-트리의 구조

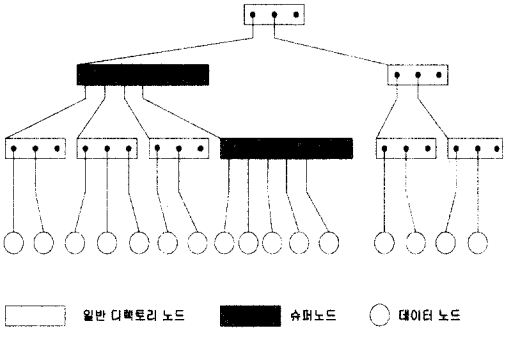
R^+-트리는 R-트리에서의 겹침 관계를 제거한 색인 구조로, 겹침 현상이 생기는 최소 경계 사각형들을 여러 개로 분할시킴으로서 겹침을 없애는 대신 트리의 높이를 증가시키게 된다. R^*-트리는 검색 경로를 최적화하기 위해서 최소 경계 사각형들의 면적뿐만 아니라 둘레까지도 색인시 이용한다.

R-트리 기반의 색인구조는 공간 데이터를 저장하기 위해서 데이터를 고차원으로 변환시킬 필요가 없기 때문에 양호한 공간 클러스터를 제공하나, 데이터의 차원이 5차원을 넘게 되면 경계 사각형들의 겹침 현상이 약 90% 정도로 급격히 증가하여 검색성능을 저하시키는 결과를 초래한다.

2.4 X-트리

X-트리[12]는 R-트리 기반의 색인구조로서 (그림 5)와 같은 구조를 갖는다. X-트리는 R^*-트리에서 공간

데이터들이 5차원을 넘게 되면 경계 사각형의 겹침 현상이 급격히 증가되어 트리의 성능이 떨어지는 단점을 보완하기 위해 제안되었다. X-트리는 공간 객체를 가리키는 데이터 노드와 고정 크기의 디렉토리 노드, 그리고 디렉토리 노드의 과중한 겹침현상을 방지하는 가변 크기의 슈퍼노드로 구성된다. 슈퍼노드를이용해 겹침 현상이 증가될 때 디렉토리내의 여러 경로를 검색해야 하는 부하를 줄이고, 슈퍼노드내에서 순차적인 검색을 수행한다. 겹침현상이 적은 경우에는 디렉토리 노드가 분할되어 트리의 높이가 높아지지만, 겹침현상이 많은 곳에서는 노드의 분할 대신에 슈퍼노드가 생성된다.



(그림 5) X-트리의 구조

이러한 슈퍼노드의 동적인 생성은 검색성능에 영향을 미친다. 슈퍼노드가 방대하게 생성되어 X-트리 전체가 슈퍼노드들 만으로 구성될 경우에는 슈퍼노드의 순차적인 검색으로 인해 검색성능이 급격히 저하된다.

3. X-트리와 kd-트리의 병합

X-트리에서 슈퍼노드의 동적인 생성으로 인한 데이터 검색성능의 저하를 방지하기 위해 kd-트리를 이용할 수 있다. 다음의 삽입 알고리즘에서와 같이 X-트리에서 최소 분할 함수의 반환 값이 실패했을 경우에는 슈퍼노드가 동적으로 생성된다. 겹침현상이 빈번히 발생할 경우에는 X-트리 전체가 슈퍼노드들 만으로 구성되거나, 또는 방대한 크기의 슈퍼노드와 디렉토리노드로 구성될 수 있다. 이러한 경우에 X-트리의 검색성능은 트리의 높이와 슈퍼노드의 순차적인 검색 시간에 종속된다. 따라서 슈퍼노드가 많이 생성되어 검색성능이 저하되는 것을 방지하기 위하여 슈퍼노드가 특정한 크기를 벗어나게 되면 슈퍼노드에 부가적인 색인구조로서 kd-트리를 첨가한다. kd-트리를 이용한 슈퍼노드의 포인트 검색을 통해 검색성능을 제고시킬 수 있다.

[삽입 알고리즘]

```
객체가 삽입될 자식 노드의 검색;
객체 삽입;
객체 삽입 후 노드의 최소 경계 사각형 계산;
 IF (삽입함수의 반환값==SPLIT) {
  IF (노드의 최소 경계 사각형 개수가 초과) {
        최소 분할 함수 호출;
        IF (최소 분할 함수의 반환값==TRUE) {
           새로운 디렉토리 노드 생성;
           노드의 분할;
          return SPLIT;
          }
       ELSE {
             새로운 슈퍼노드 생성;
             노드의 분할;
             return SUPERNODE;
             }
        }
  }
  ELSE {
        IF (삽입함수의 반환값==SUPERNODE)
            현재 노드를 슈퍼노드로 확장;
        }
  IF ( 슈퍼노드>MAX )
     슈퍼노드에서의 kd-트리 생성;
```

위의 알고리즘은 X-트리에서 슈퍼노드가 특정 크기 MAX를 넘게되면 해당 슈퍼노드에 kd-트리형의 부가적인 색인구조를 삽입하는 알고리즘이다. 최소 경계 사각형을 구성할 필요가 없이 kd-트리는 데이터 객체를 고차원 데이터 공간으로 매핑하여 색인을 구성하기 때문에 슈퍼노드내에서 최소 경계 사각형의 겹침현상을 고려할 필요가 없다.

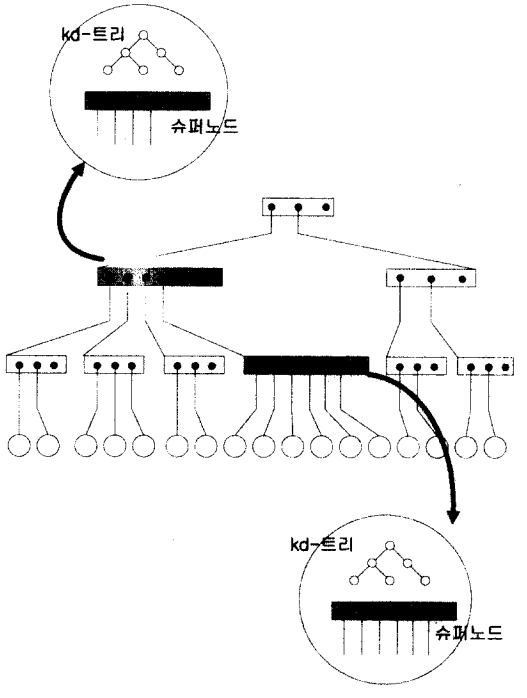
(그림 6)은 앞에서 제시한 알고리즘에 의해 X-트리의 슈퍼노드에 색인구조로서 kd-트리를 첨가한 것을 나타내고 있다. X-트리에 데이터를 삽입할 경우에 데이터들이 고차원으로 구성되거나, 데이터 분포가 밀집될 경우에 데이터들 사이에 겹침 현상이 급격히 증가하게 된다. 그러므로 디렉토리노드의 분할에 따른 트리의 성능저하를 방지하기 위해 X-트리에 슈퍼노드를 생성하여 불합리한 노드 분할을 방지한다. 그러나 이러한 슈퍼노드가 빈번하게 생성되고, 생성된 슈퍼노드가 점점 확대될 경우 슈퍼노드의 순차검색에 따른 검색성능 저하를 초래한다. 이를 방지하기 위하여 (그림 6)과 같이 슈퍼노드의 크기가 최대값 이상일 때 슈퍼노드

에 부가적인 색인구조로 kd-트리를 생성시킨다. 슈퍼노드를 검색해야 할 경우에 순차검색 대신에 kd-트리에 의한 포인트 검색을 수행하여 검색성능의 저하를 방지할 수 있다.

X-kd-트리는 연결리스트를 이용하여 구현할 수 있으며, 각 노드의 크기는 검색성능에 영향을 미치게 되므로 kd-트리의 효율적인 생성과 검색 시간을 분석하여 슈퍼노드의 최대 허용크기를 확정해야 한다.

[삭제 알고리즘]

```
삭제할 데이터 노드를 루트로부터 검색
데이터 노드의 삭제;
Do While (노드 != 루트노드)
      {
      IF (디렉토리 노드 == Null)
         디렉토리 노드 삭제;
      IF (디렉토리 노드 == 슈퍼노드)
          {
            kd-트리에서 해당 노드 삭제;
            kd-트리 재구성;
            슈퍼노드에서 해당 최소 경계
            사각형 삭제;
          }
      IF (슈퍼노드 < MAX)
         kd-트리 삭제;
    }
```



(그림 6) X-트리와 kd-트리가 병합된 구조

본 논문에서는 주로 X-kd-트리의 검색성능 분석에 필요한 알고리즘 만을 구현, 실험에 이용하였다. 향후 연구과제로 삭제용 알고리즘을 기술하면 위와 같다.

X-kd-트리에서 데이터 노드의 검색방법은 X-트리의 검색방법과 유사하다. 관련 데이터를 포함하고 있는 최소 경계 사각형을 검색하기 위해서 루트 노드로부터 하위 디렉토리 노드로의 검색이 진행된다. 검색코자하는 데이터가 여러 최소 경계 사각형에 중복 저장되었거나, 겹쳐있을 수 있기 때문에 디렉토리 노드로부터 하위 노드까지의 다중 경로를 통하여 검색하게 된다. 그러나 하위 디렉토리 노드를 검색하는 도중에 슈퍼노드를 만나게 되면 최소 경계 사각형들을 순차적으로 검색하지 않고, 미리 생성된 kd-트리를 이용해 최소 경계 사각형을 검색한다.

[검색 알고리즘]

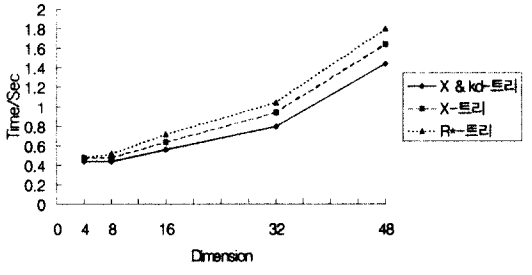
```
검색할 데이터 객체를 포함하고 있는 사각형 S의
최소 경계 사각형 계산;
//서브트리의 검색
Do While (루트노드 != 터미날노드)
      {
      IF (서브트리 노드 == 디렉토리 노드)
         최소 경계 사각형 개수에 따라
         디렉토리 노드의 계층적인 검색;
      IF (서브트리 노드 == 슈퍼노드) AND
         (슈퍼노드 < MAX)
         최소 경계 사각형 크기에 맞는 엔트리를
         슈퍼노드에서 순차적으로 검색;
      ELSE
          kd-트리에 의한 수퍼노드 검색;
      }
//터미날 노드에서의 객체검색
      IF (터미날노드 == 디렉토리 노드)
         S와 겹치는 최종 엔트리 검색;
      IF (터미날노드 == 슈퍼노드) AND
         (슈퍼노드 < MAX)
         슈퍼노드에서 최종 엔트리 검색;
      ELSE
          {
          kd-트리에 의해 슈퍼노드에서
          최종 엔트리 검색;
          }
      최종 엔트리의 포인터에 의해 데이터
      객체의 검색;
```

4. 성능분석

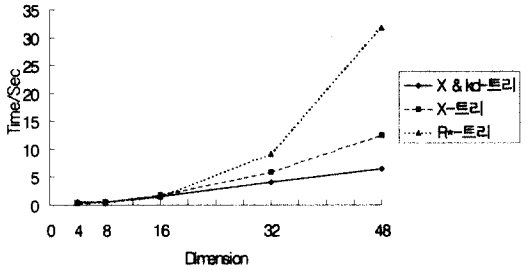
본 연구에서 제안한 X-kd-트리의 성능을 평가하기 위하여 4, 8, 16, 32, 48차원에 대한 균등분포, 지수분

포, 정규분포를 이루는 데이터를 각 각 15,000개씩 갖는 트리를 구성하였다. 각 데이터분포에 대한 포인트 질의와 최근접 검색질의의 결과는 (그림 7), (그림 8), (그림 9)와 같다. 실험에 사용한 X-트리 소스는X-트리 제안자인 S. Berchtold가 개발한 프로그램을 이용하였으며, R*-트리의 소스는 X-트리 소스에서 겹침과 관계된 부분만을 수정하여 사용하였다[17].

<표 1>, <표 2>, <표 3>에서와 같이 각 각의 데이터 분포에 따라 R*-트리, X-트리, X-kd-트리의 검색성능을 비교, 분석한 결과 X-kd-트리의 성능이 다른 트리들 보다 우수한 것으로 나타났다. R*-트리는 데이터가 고차원으로 구성될수록 최소 경계 사각형들 사이에 겹침현상이 급격히 증가되며, 이로인해 빈번한 노드분할이 발생된다. 결과적으로 트리의 높이가 증가되어 다른 트리들 보다 검색시간이 길어지게 된다. X-트리에서는 겹침현상이 자주 발생하는 노드에 슈퍼노드를 생성시킴으로서 R*-트리에 비해 불합리한 노드분할을 최소화할 수 있지만 고차원 데이터에 의해 슈퍼노드가 방대하게 생성될 수 있다. 데이터의 차원이 높아지거나, 데이터의 갯수가 증가할수록 방대한 크기의 슈퍼노드가 여러 노드에 생성될 수 있다. 결과적으로 슈퍼노드의 순차적인 검색으로 인해 검색효율이 저하되는 현상이 발생된다. X-kd-트리는 X-트리의 슈퍼노드에 kd-트리를 부가시킨 트리구조로서, 슈퍼노드의 검색을

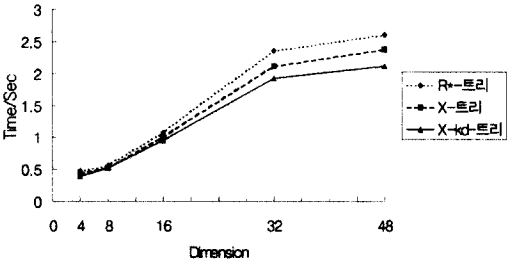


(a) 포인트질의

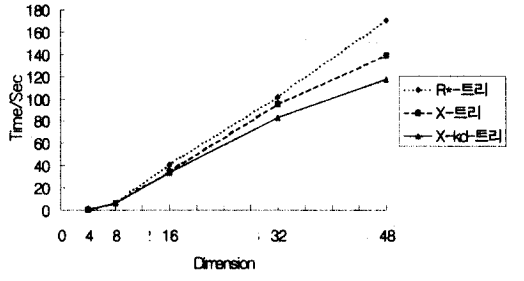


(b) 최근접 검색질의

(그림 7) 균등분포 데이터의 검색성능 측정



(a) 포인트질의



(b) 최근접 검색질의

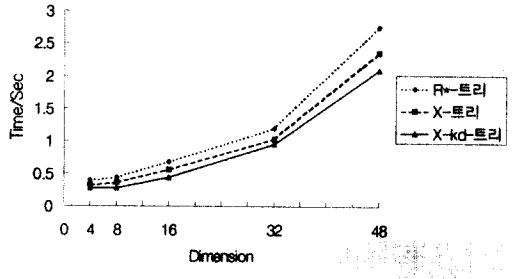
(그림 8) 지수분포 데이터의 검색성능 측정

〈표 1〉 균등분포 데이터의 검색시간 비교표

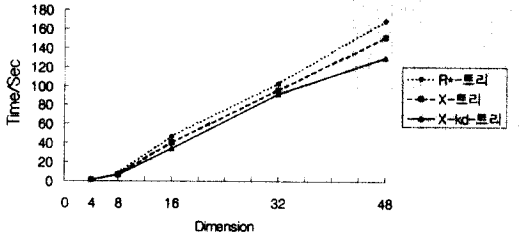
트리 \ 차원		4	8	16	32	48
R*-트리	포인터질의	0.48	0.52	0.72	1.04	1.80
	최근접검색질의	0.44	0.63	1.51	9.20	31.78
X-트리	포인터질의	0.44	0.48	0.64	0.94	1.64
	최근접검색질의	0.44	0.62	1.80	5.84	12.46
X-kd-트리	포인터질의	0.44	0.44	0.56	0.80	1.44
	최근접검색질의	0.52	0.54	1.61	4.06	6.51

〈표 2〉 지수분포 데이터의 검색시간 비교표

트리 \ 차원		4	8	16	32	48
R*-트리	포인터질의	0.48	0.56	1.08	2.36	2.60
	최근접검색질의	0.80	6.13	40.93	101.99	170.80
X-트리	포인터질의	0.44	0.52	1.00	2.12	2.38
	최근접검색질의	0.79	6.11	34.09	95.37	139.27
X-kd-트리	포인터질의	0.40	0.52	0.96	1.92	2.12
	최근접검색질의	0.78	6.10	33.77	83.01	117.67



(a) 포인트 질의



(b) 최근접 검색질의

(그림 9) 정규분포 데이터의 검색성능 측정

〈표 3〉 정규분포 데이터의 검색시간 비교표

트리 \ 차원		4	8	16	32	48
R*-트리	포인터질의	0.40	0.44	0.68	1.20	2.76
	최근접검색질의	1.17	7.78	47.53	103.31	168.90
X-트리	포인터질의	0.32	0.36	0.56	1.04	2.36
	최근접검색질의	1.13	6.89	40.56	96.07	152.06
X-kd-트리	포인터질의	0.28	0.28	0.44	0.96	2.10
	최근접검색질의	1.11	6.79	34.52	91.95	130.47

kd-트리에 의해 수행하므로 데이터가 고차원으로 구성되더라도 다른 트리들에 비해 양호한 검색성능을 나타낸다.

(그림 7), (그림 8), (그림 9)에서 확인할 수 있듯이 X-kd-트리는 데이터 차원이 증가해도 그래프상의 기울기가 완만하게 증가한다. 그러므로 고차원 데이터와 데이터 갯수의 증가로 인해 빈번한 겹침현상이 발생되는 경우에도 X-kd-트리는 다른 트리의 색인구조 보다 우수한 성능을 갖는 색인구조로 입증되었다.

5. 결 론

본 논문에서 제안한 X-kd-트리는 슈퍼노드에서 순차검색을 해야만 하는 X-트리의 검색방법을 보완하기 위하여 X-트리에 부가적인 색인구조로서 kd-트리를 접목시킨 구조이다. 고차원, 대용량 공간 데이터에 대한 검색효율을 높이기 위해 기존의 X-트리 알고리즘에 포인트 색인구조를 갖는 kd-트리 알고리즘을 병합하여 X-트리의 단점을 보완하였다. X-kd-트리에서는 데이터들이 고차원 벡터들로 구성되거나, 공간상의 어느 특정지역에 밀집, 분포되어 방대한 슈퍼노드가 생성되더라도 양호한 데이터 검색이 가능하다. 본 논문에서는 부가적인 색인 생성에 따른 오버헤드를 최소화하기 위하여 포인트 색인구조를 갖는 트리들 중 가장 간단한 구조인 kd-트리를 이용하였으나, kd-트리 이외에도 다양한 트리 구조들을 X-트리에 접목시켜 검색효율성을 상호 비교, 분석할 필요성이 있다. 또한 향후 연구과제로는 X-kd-트리에서 데이터의 삭제, 갱신연산의 구현과 함께 제안한 트리형태의 색인구조를 응용분야에 실제로 적용시켜 그 활용 가능성을 검증하는 문제 등을 들 수 있다.

참 고 문 헌

[1] B. Prabhakaran, "Multimedia Database Management System," Kluwer Academic Publishers, 1997.

[2] David Lomet, "A Review of Recent Work on Multi-attribute Access Methods," SIGMOD RECORD, Vol.21, No.3, September, 1992.

[3] Elisa Bertino, 외6인, "Indexing Techniques for Advanced Database Systems," Kluwer Academic Publishers, 1997.

[4] Thomas A. Mueck, Martin L. Polaschek, "Index Data Structures in Object-Oriented Databases," Kluwer Academic Publishers, 1997.

[5] Ooi B. C., McDonell K. J., Sacks-Davis R., "Spatial kd-tree : An Indexing Mechanism for Spatial Databases," In Proc. 11th International Conference on Computer Software and Applications, 1987.

[6] Guttman A., "R-tree : A Dynamic Index Structure for Spatial Searching," ACM SIGMOD, 1984.

[7] Bently, J. L., "Multidimensional Binary Search Trees Used for Associative Searching," Communications of the ACM, 18(9), 1975.

[8] J. T. Robinson, "The K-D-B-tree : A Search Structure for Large Multidimensional Dynamic Indexes,"

ACM SIGMOD, Apr., 1981.

[9] Finkel R. A., Bentley J. L., "Quad Trees : A Data Structure for Retrieval on composite keys," Acta Informatica, 1974.

[10] K. I. Lin, H. Jagadish, C. Faloutsos, "The TV-tree : An Index Structure for High Dimensional Data," VLDB Journal, Vol.3, pp.517-542, 1994.

[11] Freeston M., "A General Solution of the N-Dimensional B-tree Problem," In Proc. ACM SIGMOD International Conference on Management of Data, pp.80-91, 1995.

[12] S. Berchtold, D. A. Keim, H-P. Kriegel, "The X-tree : An Index Structure for High-Dimensional Data," Proc. of the 22nd VLDB Conference, 1996.

[13] 이석호, "화일처리론", 정익사, 1997.

[14] David A. White, Ramesh Jain, "Algorithms and Performance," In Proc. of the SPIE : Storage and Retrieval for Image and Video Databases IV, Vol.2670, pp.62-75, 1996.

[15] Sellis T., Roussopoulos N., Faloutsos C., "The R^+-tree : "A Dynamic Index for Multi-Dimensional Objects," In Proc. 13th International Conference on VLDB, 1987.

[16] Beckmann N., Kriegel H. P., Schneid er R., Seeger B., "The R^*-tree : An Efficient and Robust Access Methods for Points and Rectangles," ACM SIGMOD, 1990.

[17] http://www.research.att.com/~berchtol/

유 장 우

e-mail : juyu@selim.co.kr

1997년 충북대학교 컴퓨터공학과 졸업(공학사)

1997년~현재 충북대학교 대학원 컴퓨터공학과 석사과정

관심분야 : 공간 데이터베이스, 객체지향 데이터베이스

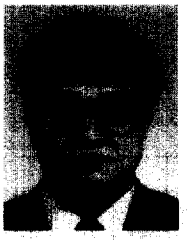
신 영 진

wonshin@enveng.chungbuk.ac.kr

1998년 충북대학교 환경공학과 졸업 (공학사)

1998년~현재 충북대학교 대학원 컴퓨터공학과 석사과정

관심분야 : 데이터 마이닝, 공간 데이터베이스



정 순 기

e-mail : soonkey@cbucc.chungbuk.ac.kr

1971년 고려대학교 독어독문과 졸업

1982년 서독 Dortmund 대학교 졸업 Informatik, Dipl.-Inform.

1994년 Groningen 대학교 졸업 Computing Science, Dr.

1985년~현재 충북대학교 컴퓨터공학과 교수

1993년 충북대학교 전자계산소장

1998년~현재 충북대학교 도서관장

관심분야 : 데이터베이스 시스템, 실시간시스템, 소프트웨어 공학