PdR-트리 : 고차원 데이터의 검색 성능 향상을 위한 효율적인 인덱스 기법

조 범 석† · 박 영 배††

요 약

피라미드 기법은 n-차원 공간 데이터를 1차원 데이터로 변환하여 B⁺-트리로 표현하고, n-차원 데이터 공간에서 하이퍼큐브 영역질의 처리로 발생하는 "차원의 저주현상"에 영향을 받지 않게 검색 시간 문제를 해결하고 있다. 구형 피라미드 기법은 피라미드 기법의 공간 분할 전략을 응용하여 유사도 검색에 적합하도록 구 영역질의 방법을 사용하고 검색 성능을 개선하고 있다. 그러나 두 방법은 데이터 크기와 차원 변화에 따른 검색 성능이 100만건 이상과 16차원 이상일 때 현저하게 저하하는 현상을 보이고 있다. 이 논문에서는 멀티미디어 데이터와 같은 고차원 데이터의 검색 성능을 향상시키기 위한 새로운 인덱스 구조로 PdR-트리를 제안한다. 모의 데이터와 실제 데이터를 이용하여 실험한 결과, PdR-트리가 피라미드 기법과 구형 피라미드 기법보다 검색 성능이 향상되었음을 보이고 있다.

PdR-Tree : An Efficient Indexing Technique for the improvement of search performance in High-Dimensional Data

Beom-Seok Joh† · Young-Bae Park††

ABSTRACT

The Pyramid-Technique is based on mapping n-dimensional space data into one-dimensional data and expressing it as B⁺-tree ; and by solving the problem of search time complexity the pyramid technique also prevents the effect of "phenomenon of dimensional curse" which is caused by treatment of hypercube range query in n-dimensional data space. The Spherical Pyramid-Technique applies the pyramid method's space division strategy, uses spherical range query and improves the search performance to make it suitable for similarity search. However, depending on the size of data and change in dimensions, the two above techniques demonstrate significantly inferior search performance for data sizes greater than one million and dimensions greater than sixteen. In this paper, we propose a new index-structured PdR-Tree to improve the search performance for high dimensional data such as multimedia data. Test results using simulation data as well as real data demonstrate that PdR-Tree surpasses both the Pyramid-Technique and Spherical Pyramid-Technique in terms of search performance.

Key word : PdR-Tree(PdR-트리), Pyramid-Technique(피라미드 기법), Spherical Pyramid-Technique(구형 피라미드 기법)

1. 서 론

최근 디지털기술과 초고속 통신망 그리고 멀티미디어 관련 기술의 급속한 발달에 따라 사용자가 원하는 형태의 문서(text), 이미지(image), 그래프(graph), 오디오(audio), 동영상(video), 지도(map)등의 통합적인 멀티미디어 객체를 효율적으로 저장, 관리해 주며 신속하고 정확한 검색결과를 제공할 수 있는 검색기술의 필요성이 절실하게 요구되고 있다.

내용기반 이미지검색(content-based image search)기법은 해당 이미지로부터 추출된 특징벡터(feature vectors)를 이용하여 실제 이미지와의 상호 유클리드 거리(euclidean distance)를 비교하여 그 거리가 가까울수록 검색 성공률이 높은 이미지가 된다. 즉 정확한 일치(exactly matching)보다는 유사도 검색(similarity search)에 비중을 두는 멀티미디어 이미지 검색 기법이다. 고차원의 멀티미디어 데이터를 이용한 내용기반 검색에서 가장 어려운 부분중의 하나가 데이터의 차원이 증가함에 따라 검색성능이 순차검색에 미치지 못하는 경우이다. 이러한 현상을 "차원의 저주(dimensional curse)"라고 한다.

기존의 대부분의 다차원 인덱스 기법[10-14]은 주로 특징 데이터의 환경이 저차원인 경우에는 좋은 검색성능을 보여주고 있다. 그러나 이미지데이터와 같은 고차원의

† 종신회원 : 명지대학교 컴퓨터공학과 데이터베이스 연구실
†† 정 회 원 : 명지대학교 컴퓨터공학과 교수
논문접수 : 2001년 3월 16일, 심사완료 : 2001년 4월 16일

멀티미디어 데이터를 사용할 경우 차원 수가 급격히 증가되므로 비효율적인 인덱스 구조가 된다. 또한 검색시 노드간의 겹침현상(overlap)을 초래하게 되므로 전체노드를 순회검색해야 하는 문제점이 일어난다. 이러한 인덱스 구조는 특징 데이터의 차원이 고차원으로 증가할수록 선형접근(linear scan)의 결과와 같아지거나 그 이하의 결과를 보여주는 검색시간문제(time complexity)가 발생된다.

결과적으로 내용 기반의 유사도 검색을 위한 효율적인 인덱스 구조는 데이터 고유의 고차원적인 특징을 저차원으로 변환한 후 인덱싱해야 할 필요가 있다. 따라서 이러한 문제점을 해결하고자 하는 여러 가지 방법 중에서 피라미드 기법[6]과 구형 피라미드 기법[19]은 데이터의 크기가 100만건과 16차원 이내에서는 좋은 검색성능을 보였으나 그 이상의 조건에서는 검색성능이 현저하게 저하하는 문제점이 있다.

이러한 문제점을 개선하기 위해, 이 논문에서는 dR-트리의 인덱스 생성기법과 구형 피라미드 기법의 공간분할 방식을 응용한 새로운 인덱스 기법으로 PdR-트리를 제안한다. PdR-트리의 검색성능이 기존의 피라미드 기법과 구형 피라미드 기법보다 향상되었음을 보이기 위해 모의 데이터와 실제 이미지 데이터를 이용하여 데이터 크기와 차원의 크기를 변화시키면서 실험하여 그 결과를 비교분석한다.

2. 관련연구

디스크 공간상의 고차원 데이터를 신속하게 처리하기 위한 디스크 기반 인덱스(disk-based index)기법에 대한 연구가 최근에 활발하게 진행되고 있다.

Hellerstein[1]은 인덱스 성능평가에 대해 저장장치 중복(storage redundancy)과 액세스 오버헤드(access overhead)라는 두 가지 기준을 설정하여 인덱스에 대한 효율성을 측정하였고 그 결과 데이터의 차원이 증가하면 선형검색과 동일한 검색시간 문제를 갖는다는 점을 이론적으로 증명하였다.

Weber[2]는 R^*-트리의 검색성능이 10차원부터 순차접근(sequential scan)의 결과와 같아지며 X-트리[9]는 22차원 이상부터 순차검색에 의한 검색결과와 동일하다는 점을 성능비교실험을 통해 증명하였다. 또한 차원의 저주현상을 해결하기 위해서 제안된 여러가지 인덱스 기법[7, 8, 16, 17]은 주로 벡터 공간과 매트릭스 공간을 이용하여 고차원 데이터에서 발생하는 급격한 차원증가에 대한 문제점을 해결하고자 하였다.

SS(Similarity Search Tree)-트리[13]는 특징벡터간의 유사도를 가중치 유클리드 거리 척도(Weighted euclidean distance metric)로 구분한다. 따라서 기존의 사각형 질의를 사용하지 않고 구 형태의 질의로 유사한 벡터를 검색할 수 있는 구조를 가지고 있다. 그러나 하위노드의 개수가 일정수준 이상 늘어나는 경우 부모노드를 비롯한 전체 트리의 볼륨이 커지게 되므로 과중한 검색시간이 소요되는 단점을 내포하고 있다.

dR-트리[15]는 임의의 포인트 위치를 중심으로 사각형을 결정하는 R-트리 구조를 사용하지 않고, 저장 영역을 차원의 반경(radius)을 중심으로 영역을 분할하는 방법을 사용하여 노드간의 영역겹침을 해결하고자 하였다. 그러나 고차원 데이터의 특징을 고려하지 않아 이미지데이터를 처리하기에는 미흡하다.

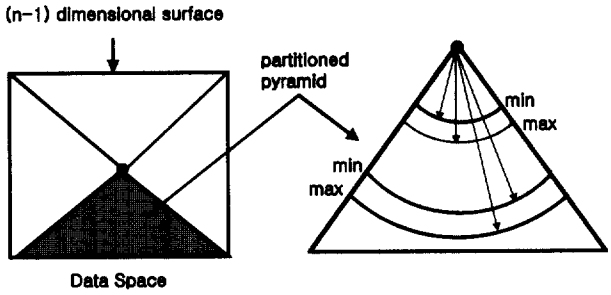
SR(Sphere Rectangle)-트리[10]는 SS-트리의 단점을 개선하기 위해 제안되었으며 R-트리의 사각형 질의방법과 SS-트리의 구 형태 질의를 결합한 구조를 가지고 있다. 그러나 SS-트리에 비해 최근접 질의(k-NN)에 대한 검색효율이 향상된 경우만을 기술하고 있고 영역질의에 대한 언급은 없다.

피라미드기법[6]은 전체 데이터공간을 2n개의 피라미드로 나눈 후 각 피라미드를 중심점에서 수평방향으로 분할하여 변환된 1차원의 값을 B^+-트리에 매핑 시키는 기법으로 고차원 데이터에서 발생하는 급격한 차원증가에 대한 문제를 극복해낸 진보적인 검색기법이다. 그러나 구 영역으로 질의할 경우 불필요한 데이터 페이지를 액세스하게되어 종합경과시간이 증가하는 문제점을 갖고 있다.

구형 피라미드기법[19]은 피라미드 기법을 이용하여 전체 데이터 공간을 2n개의 피라미드로 나눈 다음 데이터 공간의 중심점에서 구 형태로 분할하여 B^+-트리의 페이지에 대응시키는 기법으로 영역질의형태를 유사도 검색에 적합한 구 형태의 질의를 적용하여 전체적으로 피라미드 기법보다 검색성능을 향상시켰다. 그러나 피라미드 기법과 구형 피라미드 기법은데이터의 크기변화에 따른 성능 비교 실험에서 데이터가 100만건 이상일 경우에 종합경과시간이 급격하게 상승하는 단점을 보이고 있고, 차원의 변화에 따른 성능 비교 실험에서도 16차원부터 급격한 검색성능의 저하가 발생하는 단점을 보이고 있다.

3. PdR-트리

이 논문에서 제안한 PdR(Pyramid doughnut Range)-트리는 dR-트리 생성기법과 구형 피라미드 기법의 공간분할 방식을 응용한 새로운 고차원 인덱스 기법이다. 즉 데이터 공간을 피라미드 모양으로 나눈 후 각 조각(slice)에 새로운 데이터가 입력되면 최소값(Rmin)과 최대값(Rmax)이 결정됨에 따라 가변적인 구 영역공간으로 조각을 분할하여 새로운 인덱스인 PdR-트리에 적용하게 된다.



(그림 1) PdR-트리의 공간분할 전략

따라서 PdR-트리는 데이터의 삽입과 삭제가 자유로운 장점을 지닌 유연한 동적(dynamic) 인덱스 구조라 할 수 있으며 이미지의 유사도 검색에서는 사각형 질의보다 구 형태의 영역질의가 적합하다고 판단된다. PdR-트리는 영역 정보를 이용하여 전체적인 노드의 방문회수를 감소시키기 때문에 결과적으로 페이지 액세스 회수와 CPU-사용시간 그리고 종합경과시간(total elapsed time)의 단축으로 인한 검색성능의 향상을 기대할 수 있다는 기본적인 아이디어에 기초한다.

3.1 공간분할전략

피라미드 기법과 구형 피라미드 기법은 d-차원의 전체 데이터 공간에서 2d개의 꼭지점을 기준으로 대각선 방향으로 교차하여 연결한다. 그러면 분할된 2d개의 피라미드 형태의 데이터공간이 생성된다. 그 다음 전체 데이터를 Quick Sort 알고리즘을 적용하여 정렬한 후 해당 페이지 단위(4096Byte)로 분할하는 방법을 이용한다. 그러나 입력 시 한꺼번에 모든 데이터를 입력시키기 때문에 새로운 데이터가 삽입 또는 삭제되는 경우 고비용이 소요된다는 단점을 가지고 있다.

PdR-트리는 (그림 1)과 같이 데이터의 삽입과 삭제연산이 가능하도록 각각의 피라미드를 하나의 데이터 페이지로 보고 새로운 데이터가 입력될 때마다 해당 페이지 단위로 분할하여 인덱스를 구성하게 된다. 또한 피라미드기법은 중심점에서의 높이로 각각의 피라미드번호를 결정했지만 PdR-트리는 각 피라미드의 중심점에서 해당 포인트까지의 유클리드 거리 함수를 이용하여 피라미드 번호를 결정하게 된다. 2d개로 분할된 각 피라미드의 번호를 결정하는 방법은 [6]의 방법을 따르며 [정의 1]과 같이 나타낸다.

[정의 1] 임의의 점 v가 속한 피라미드 번호 결정

$$i = \begin{cases} j_{\max} & \text{if}(v_{j_{\max}} < 0.5) \\ (j_{\max} + d) & \text{if}(v_{j_{\max}} \geq 0.5) \end{cases}$$

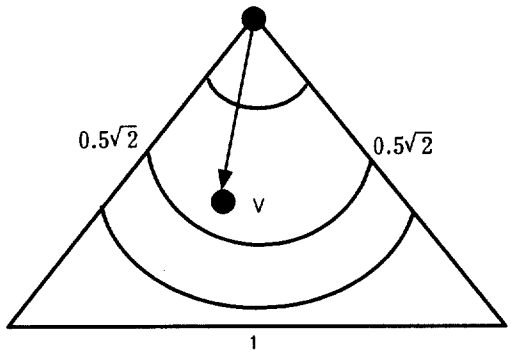
$$j_{\max} = (j \mid \forall k, 0 \leq (j,k) < d, j \neq k : |0.5 - v_j| \geq |0.5 - v_k|)$$

여기서 i는 임의의 점 v가 속한 피라미드 번호를 나타내며, j_{max}는 d-차원의 점 v의 각 차원중 중심점에서 가장 멀리 떨어진 차원을 나타낸다.

그러나 피라미드 기법에서는 j = k인 경우에 대한 언급이 없기 때문에 부가적으로 정의 1.1을 아래와 같이 적용한다.

[정의 1.1] j = k인 경우의 피라미드번호 결정

$$j_{\max} = (j \mid (\forall k,\ 0 \leq (j,k) < d,\ j = k : |0.5 - v_j| \leq |0.5 - v_k|))$$



(그림 2) 중심점에서 포인트 v까지의 거리

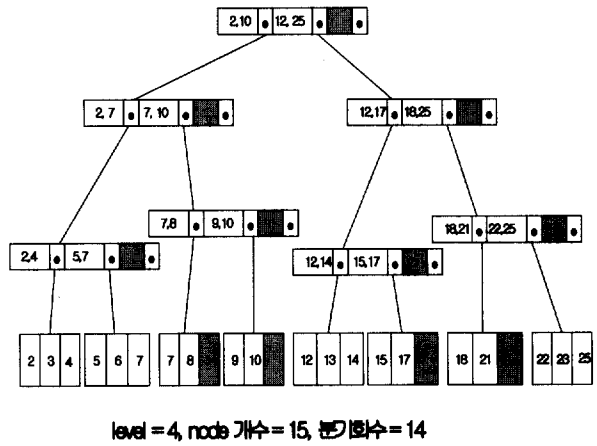
[정의 2] PdR-트리에서 포인트 v의 거리(d_v) 결정

$$d_v = \sqrt{\sum_{i=0}^{d-1} (0.5 - v_i)^2}$$

피라미드 기법의 높이에 의한 포인트 v의 거리를 결정하는 방법과는 다르게 PdR-트리는 (그림 2)처럼 중심점에서의 유클리드 거리(euclidean distance)를 사용[19]하여 결정한다.

따라서 2차원의 경우 최대거리구간은 중심점에서의 최대 반경과 같은 $[0, 0.5\sqrt{2}]$이다.

3.2 PdR-트리의 구조



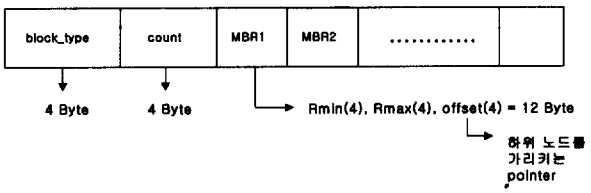
(그림 3) PdR-트리의 구조

하나의 노드에 삽입 가능한 최소 엔트리 개수가 3이라고 가정하고 20개의 입력 데이터를 이용하여 구성한 PdR-트

리의 구조는 (그림 3)과 같다. PdR-트리는 유사 이미지영역에 대한 정보를 보유하고 있기 때문에 노드의 방문회수가 감소된다. 따라서 전체페이지 접근회수가 감소되어 검색성능을 개선하게되는 장점을 가지고 있다. 피라미드 기법의 경우 불필요한 데이터페이지에 접근하여 검색 시 리프 노드까지 순회한 후 유사도에 대한 판별이 이루어지게 된다. 이 점은 종합경과시간에 영향을 주어 검색성능이 저하되는 원인이 된다.

3.2.1 루트 및 중간노드

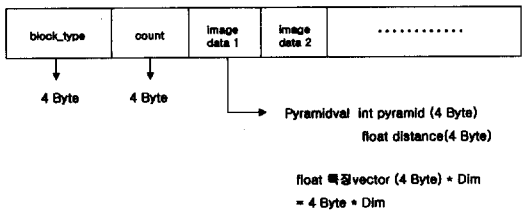
PdR-트리의 루트노드와 중간노드는 하위노드가 포함할 수 있는 반지름의 영역인 MBR(Minimum Bounding Region)과 하위노드와 연결된 주소를 나타내는 자식 포인터(pointer)로 구성되어 있다. 따라서 (그림 4)와 같이 MBR은 페이지단위의 최소경계영역으로 반지름의 최소값(R_{min})과 최대값(R_{max})을 포함하고 있다.



(그림 4) 루트 및 중간노드의 구조

3.2.2 리프노드

PdR-트리의 리프 노드는 동일한 레벨에 있으며 최소 m개에서 최대 M개의 페이지를 포함하며 그림 5와 같이 해당 이미지를 지정하는 이미지 번호(image_no), 그리고 해당 이미지에 대한 특징벡터 값으로 구성되어 있다.



(그림 5) 리프노드의 구조

3.3 PdR-트리의 알고리즘

3.3.1 삽입

먼저 삽입할 리프노드를 선택한 후, 그 리프노드에 데이터를 삽입하고 루트에서 삽입되는 노드까지의 모든 노드들의 MBR 내용은 데이터 삽입과 동시에 갱신된다.

```
Insert Data (Data *dp) {
    Choose Leaf {
        트리를 검색하여 리프노드 선택 ;
        선택 경로의 노드들을 스택에 저장 ;
    }
    if (leaf_is_full) {
        leaf_split {
            리프노드 분할 ;
            스택에 저장된 노드들에 MBR 추가 및 갱신 ;
            필요한 경우, MBR도 연쇄적으로 분할 ;
        }
    }
    else {
        leaf_append {
            리프노드에 데이터 삽입 ;
            오름차순으로 리프노드의 데이터 정렬 ;
        }
        Update Bounding Region {
        스택에 저장된 노드들의 MBR을 갱신 ;
        }
    }
}
```

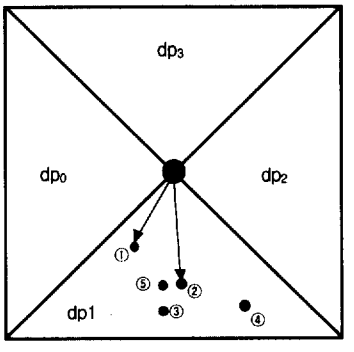
(삽입 알고리즘)

삽입 시 노드에 오버플로우가 발생하는 경우, 최소 엔트리 개수 m개의 반경으로 이루어진 해당영역내의 점들에 대해 중심점에서의 거리를 구해 Min과 Max를 선택하고 나머지 엔트리는 거리 상 가까운 쪽으로 배당한다. 최대 엔트리 개수를 초과하여 새로운 엔트리가 삽입되는 경우, 분할이 발생한다. 이 경우 부모노드는 자식노드의 Min값과 Max값을 저장하게 된다. 따라서 검색을 할 경우 구간의 정보를 가지고 있기 때문에 보다 빠른 검색성능을 보장하게 된다.

예를 들어 (그림 6)에서 2차원 평면상 1번 피라미드(dP_1)에서 4개의 평면의 길이가 1이고 빗면의 길이가 $0.5\sqrt{2}$라고 할 때, <표 1>과 같이 주어진 2차원 좌표 값에 대한 중심점과의 거리는 [정의 2]에 따라 구할 수 있고, 거리순위는 가장 먼 거리를 우선으로 순위가 정해진다. 또한 한 페이지에 삽입 가능한 최대 엔트리 개수를 3이라 가정하자.

<표 1> 입력 데이터

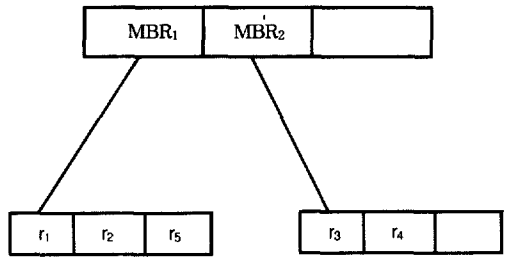
번 호	좌 표 값	중심점에서 거리	거리순위	삽입순위
①	(0.4, 0.3)	r_1 = 0.05	5	1
②	(0.6, 0.2)	r_2 = 0.1	3	2
③	(0.5, 0.1)	r_3 = 0.16	2	3
④	(0.8, 0.1)	r_4 = 0.25	1	4
⑤	(0.5, 0.2)	r_5 = 0.09	4	5



(그림 6) dP_1에서의 삽입

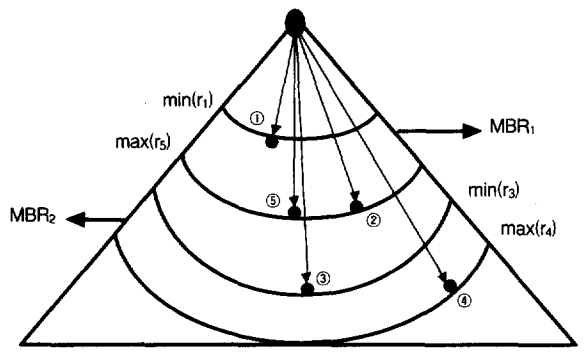
먼저 r_1, r_2, r_3는 같은 리프노드에 노드 분할 없이 삽입

된다. r_4 삽입 시, 리프노드에 오버플로우가 발생하여 분할이 일어난다.



(그림 7) 리프 노드의 분할

분할은 (그림 7)과 같이 MBR_1의 최소값은 r_1, 최대값은 r_2가 되고, MBR_2의 최소값은 r_3, 최대값은 r_4가 된다. 마지막으로 r_5가 삽입되는 경우 (그림 8)과 같이 이미 존재한 r_2보다 먼 거리에 있는 r_5가 MBR_1의 최대값이 된다.



(그림 8) 공간분할의 예

3.3.2 삭제

R* 트리의 삭제 알고리즘과 같이 삭제할 해당 엔트리를 포함하고있는 노드에서 엔트리를 삭제하고, 해당 노드의 엔트리 개수가 최소 엔트리 개수 미만인 경우는 해당 노드를 삭제하고, 해당 노드의 엔트리들을 재 삽입한다.

3.3.3 영역 질의

영역 질의는 <중심 q, 반지름 ε>의 구 형태로 주어진다. PdR 트리에서는 영역 질의 처리는 먼저 겹치는 피라미드를 결정하고, 결정된 피라미드에서 질의가 포함하는 영역의 Min, Max에 겹치는 페이지를 디스크로부터 읽어들인다.

읽어들인 데이터 모두에 대해 질의 점과의 거리를 비교하기 전에 중심과 반지름으로 이루어진 하이퍼큐브로 읽어들인 데이터를 1차 필터링한 후, 필터링되어진 데이터들과의 거리를 비교하여 중심 q에서 거리가 ε이하 떨어진 데이터를 결과로 출력한다. 2차원의 예를 들면, 하이퍼큐브는 $(q.x-\varepsilon, q.y-\varepsilon, q.x+\varepsilon, q.y+\varepsilon)$의 사각형이다.

```
Point_Set PdR_Tree : : hypersphere_query(point q, range ε){
  for (i=0 ; i<2d : i++) {
    if (피라미드가 질의영역에 겹치는가?) {
      range ← 질의 영역에 겹치는 피라미드 영역 결정 ;
      cs ← PdR 영역질의 ;
      for (c=cs.first ; cs.end : cs.next) {
        if (inside_Hypercupe(c, q, ε)) {
          if (point_in_circle(c, q, ε)) res.add(c)
        }
      }
    }
  }
  return res ;
}
```

PdR-트리의 영역질의 처리

PdR-트리는 영역정보를 저장하고있기 때문에 피라미드 기법과 구형 피라미드 기법에서 사용한 B^+-트리에 비해 노드방문회수가 감소된다.

4. 실험 및 분석

이 논문에서 제안한 PdR-트리가 영역 질의처리에 효율적임을 나타내기 위해 기존의 피라미드 기법과 동일한 실험조건과 환경에서 비교하여 실험하였다. 즉 비교실험에 대한 공정성을 부과하기 위해서 피라미드 기법과 구형 피라미드 기법에서 사용한 실험방식과 동일하게 실제 데이터(real data)를 이용한 실험과 모의 데이터(simulated data)를 이용한 실험을 하였고 세부적으로 데이터 크기변화에 따른 실험과 데이터 차원변화에 따른 페이지 액세스 회수실험과 CPU-사용시간을 비롯한 종합경과시간을 비교하여 실험하였다. 또한 구형 피라미드기법과의 성능비교는 상호 실험환경이 다르기 때문에 직접적인 비교실험은 할 수가 없었다. 다만 간접비교의 수단으로 종합경과시간을 비교해보면 검색성능에 대해 PdR-트리가 우수함을 알 수 있었다. 사용언어는 C-언어를 이용하여 구현했으며 gcc로 컴파일 하였다. 실험환경은 SUN-Ultra II 워크스테이션으로 메인 메모리 256MB이며 12GB의 기억용량을 가진 HDD를 사용하였다. 실험차원은 최소 8차원에서 최대 24차원이며 할당된 페이지 크기는 4KByte로 설정하여 실험하였다.

4.1 모의 데이터를 이용한 경우

실험결과, PdR 트리는 피라미드 기법보다 페이지 접근회수가 현저하게 감소한다는 것을 알 수 있었고, 차원변화 시에는 피라미드 기법보다 CPU 시간을 많이 소비한다는 사실을 알 수 있었다. 이러한 결과는 PdR 트리는 상위 노드가 하위 노드들에 대한 영역정보를 가지고 있으므로 불필요한 페이지 접근 횟수를 최소화하였고, 질의 시 피라미드와는 달리 수학적인 연산을 많이 행하므로 CPU 시간은 피라미드보다 많이 허비하기 때문이다. 그러나, CPU 시간은 디스크 읽기에 걸리는 시간에 비해 매우 적은 시간이므로 전체적인 시간 측면에서는 피라미드 기법보다 PdR-트리의 검색성능이 향상되었음을 알 수 있었다. 영역질의를 수행하기 위하여 균

등분포 되어있는 2,000,000개의 인위적으로 생성한 모의 데이터를 사용하였으며 0.001%의 선택도(selectivity)를 만족하도록 설정하여 실험하였다. 여기서 말하는 선택도는 입력된 총 객체의 개수 중에서 질의조건에 만족하여 선택된 객체의 개수를 말한다. 일반적으로 유사도 검색에서 리콜되는 데이터량이 일정한 조건 이상이 되면 정확도가 현저하게 감소하여 순차검색방법보다 성능이 저하되는 현상이 나타나게 된다. 따라서 선택도는 특정 도메인에서 적합한 영역질의를 하기 위한 척도가 된다. 모의 데이터를 이용한 실험은 데이터 크기와 차원변화에 따른 성능을 비교하여 10회 반복하여 실험하였다.

4.1.1 데이터 크기변화에 따른 성능 비교

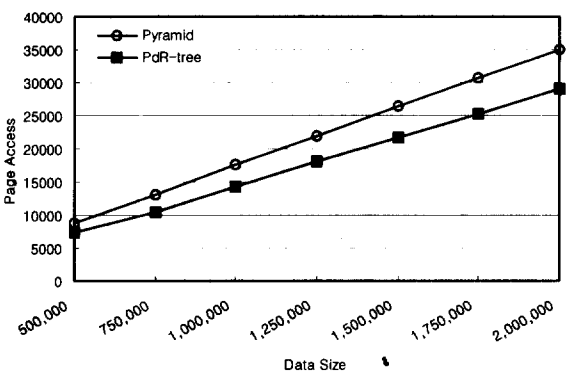
데이터의 크기를 500,000개부터 250,000개씩 증가시켜 2,000,000개까지 변화시키면서 성능을 비교하였다. <표 2>는 데이터 크기변화에 따른 검색성능을 수치로 산정하여 비교하였음을 보여주고 있고 (그림 9)는 성능비교에 대한 실험결과를 그래프로 표현하였다.

실험결과 피라미드 기법에 비해 (그림 9)의 (a)에서 보는 바와 같이 전체 페이지 액세스회수가 최소 17%에서 최대 21% 감소하였고, (그림 9)의 (c)에서 보는 바와 같이 종합경과시간의 경우에도 최소 57%에서 최대 60%검색속도가 향상되었다. 또한 구형 피라미드 기법의 실험결과[19]에 비해 최소 20%~최대 32%정도 종합경과시간이 감소하였음을 알 수 있었다.

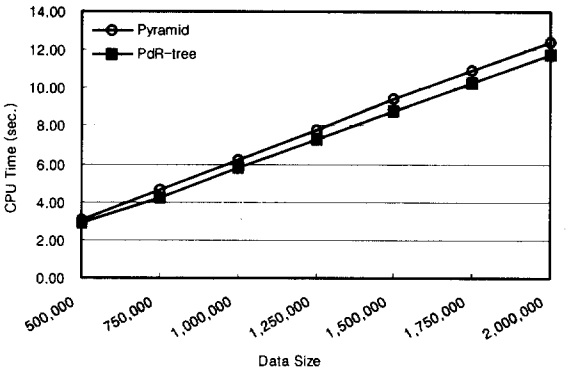
〈표 2〉 성능에 대한 수치비교

(Dimension = 16, Selectivity = 0.001%)

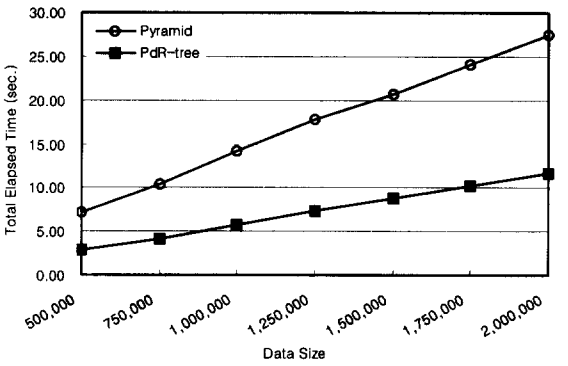
Data Size	Page Access			CPU Time(sec.)			Total Time (sec.)		
	Pyramid	PdR-tree	drop	Pyramid	PdR-tree	drop	Pyramid	PdR-tree	drop
500,000	8787	7291	17%	3.10	2.91	6%	7.15	2.91	59%
750,000	13175	10358	21%	4.62	4.19	9%	10.36	4.19	60%
1,000,000	17561	14304	19%	6.19	5.76	7%	14.14	5.77	59%
1,250,000	21944	18050	18%	7.76	7.28	6%	17.72	7.28	59%
1,500,000	26328	21715	18%	9.45	8.77	7%	20.61	8.76	57%
1,750,000	30717	25354	17%	10.92	10.23	6%	24.05	10.26	57%
2,000,000	35103	29020	17%	12.47	11.74	6%	27.45	11.75	57%



(a) 페이지 액세스회수 비교



(b) CPU-사용시간 비교



(c) 종합경과시간 비교

(그림 9) 데이터 크기변화에 따른 성능 비교

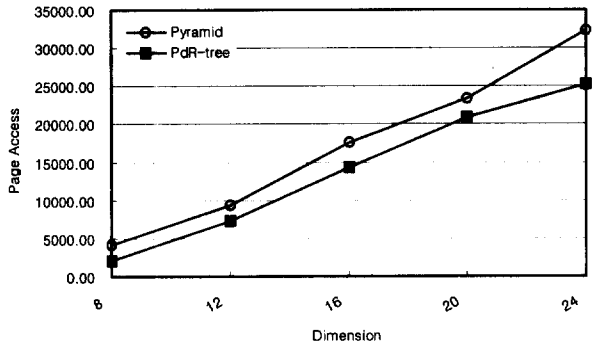
4.1.2 차원의 변화에 따른 성능 비교

차원의 변화에 따른 성능 비교실험은 차원의 크기를 8차원부터 24차원까지 변화시키면서 실험하였다. <표 3>에서 차원에 변화를 주어 실험한 수치를 비교할 수 있고, (그림 10)에서 살펴보면 입력데이터의 차원이 8차원일 때 가장 큰 속도의 향상을 보여주었다. 페이지 액세스회수의 경우 (그림 10)의 (a)에서 보는 바와 같이 피라미드 기법에 비해 최소 25%에서 최대 65%의 감소효과를 볼 수 있었으며, 종합경과시간의 경우 (그림 10)의 (c)에서 보는 바와 같이최소 34%에서 최대 74%의 검색속도 향상이 있었다. 또한 구형 피라미드 기법[19]의 실험결과와 비교해보면 8%~31%의 종합경과시간에 대한 성능향상이 있었음을 알 수 있었다.

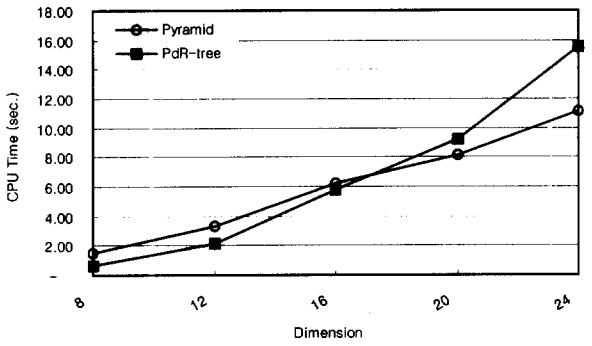
〈표 3〉 성능에 대한 수치비교

(Data size = 1,000,000 Selectivity = 0.001%)

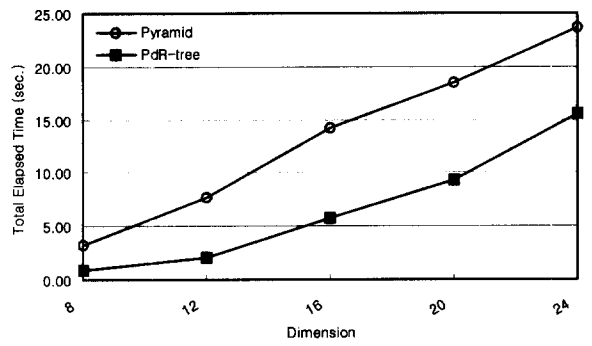
Dim.	Page Access			CPU Time(sec.)			Total Time (sec.)		
	Pyramid	PdR-tree	drop	Pyramid	PdR-tree	drop	Pyramid	PdR-tree	drop
8	4091	2110	48%	1.48	0.61	59%	3.30	0.85	74%
12	9269	7351	21%	3.30	2.11	36%	7.66	2.12	72%
16	17561	14304	19%	6.19	5.76	7%	14.14	5.77	59%
20	23332	20894	10%	8.09	9.27	-15%	18.45	9.28	50%
24	32402	25190	22%	11.10	15.49	-40%	23.65	15.49	34%



(a) 페이지 액세스회수 비교



(b) CPU-사용시간 비교



(c) 종합경과시간 비교

(그림 10) 차원변화에 따른 성능비교

4.2 실제 데이터를 이용한 경우

실험에 사용된 실제 데이터는 내용기반 이미지 검색 시스템[18]에서 사용하였던 16차원으로 구성된 56,230개의 이미지 특징벡터를 이용하였다. 실제실험 데이터를 사용하여 질의범위를 0.1에서 0.9까지 변화시키면서 실험한 결과를 <표 4>에서 비교해보면 피라미드 기법에 비해 페이지 액세스회수가 최대 89%, 종합경과시간이 최대 94% 감소하였다.

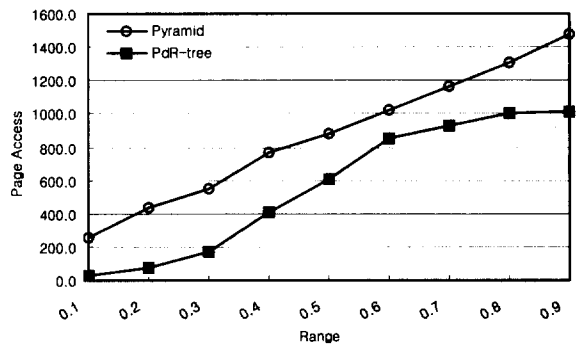
PdR-트리는 영역정보를 이용하여 전체적인 노드의 방문회수를 감소시키기 때문에 페이지 액세스 회수와 CPU-사용시간, 결과적으로 종합경과시간이 단축되어 검색성능의 향상을 이룰 수 있었다.

전체적인 성능비교실험에 대한 결과를 보면 PdR-트리가 피라미드 기법과 구형 피라미드 기법 보다 향상된 검색성능을 보여주고 있다. 다만 CPU-사용시간 비교의 경우, (그림 10)의 (b)에서 18차원 이상에서 CPU-사용시간이 증가하였고, (그림 11)의 (b)에서 질의범위 0.6~0.9에서 상대적으로 CPU-사용시간이 다소 소모되는 점을 발견할 수 있었다.

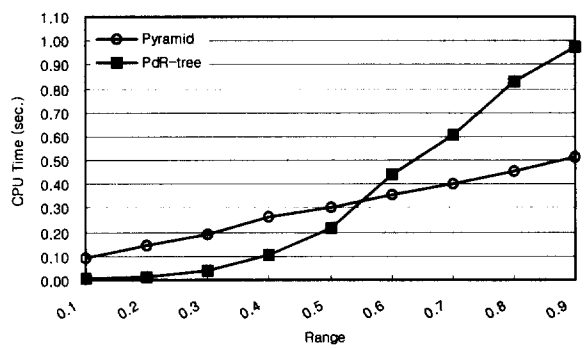
〈표 4〉 성능에 대한 수치비교

(Data size = 56230)

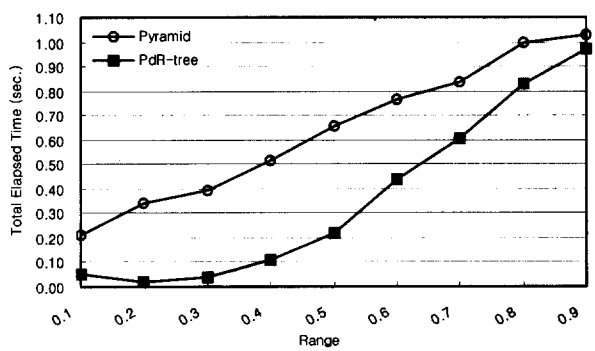
Range	Page Access			CPU Time (sec.)			Total Time (sec.)		
	Pyramid	PdR-tree	drop	Pyramid	PdR-tree	drop	Pyramid	PdR-tree	drop
0.1	259.2	28.7	89%	0.09	0.01	92%	0.22	0.05	76%
0.2	432.6	76.0	82%	0.15	0.01	92%	0.34	0.02	94%
0.3	550.8	172.7	69%	0.19	0.04	80%	0.39	0.04	90%
0.4	764.4	402.8	47%	0.26	0.11	60%	0.52	0.11	79%
0.5	884.4	601.7	32%	0.31	0.22	29%	0.65	0.22	67%
0.6	1021.4	850.0	17%	0.35	0.44	-24%	0.77	0.44	43%
0.7	1165.2	923.2	21%	0.40	0.60	-49%	0.84	0.60	28%
0.8	1311.2	999.3	24%	0.46	0.83	-83%	0.99	0.83	16%
0.9	1477.7	1016.0	31%	0.51	0.97	-90%	1.03	0.97	6%



(a) 페이지 액세스회수 비교



(b) CPU-사용시간 비교



(c) 종합경과시간 비교

(그림 11) 질의범위변화에 따른 성능 비교

5. 결 론

이 논문에서는 고차원 데이터의 검색성능 향상을 위한 새로운 인덱스 구조인 PdR-트리를 제안하였다. PdR-트리는 dR-트리의 인덱스 생성기법과 구형 피라미드 기법의 공간분할 방식을 응용한 새로운 고차원 인덱스 기법이다.

피라미드 기법과 구형 피라미드 기법은 모든 입력 데이터를 한꺼번에 입력시키기 때문에 새로운 데이터에 대한 삽입연산과 삭제연산의 경우에 고비용이 소요된다는 단점과 데이터 크기를 변화시키면서 실험한 성능비교에서 데이터 크기가 100만건 이상일 때 종합경과시간이 급격하게 증가하며, 데이터 차원이 16차원 이상일 때부터 검색성능이 급격하게 저하되는 문제점이 있다.

PdR-트리는 피라미드 기법과 구형 피라미드 기법에서 나타난 16차원 이상의 고차원과 100만건 이상 대량데이터에서 발생하는 문제점을 개선하여 검색성능을 향상하였다. PdR-트리의 검색성능이 향상되었음을 보이기 위해 피라미드 기법과 동일한 실험조건을 부여하여 실험하였다. 또한 종합경과시간을 간접비교 실험하여 구형 피라미드 기법 보다 PdR-트리의 검색성능이 향상되었음을 알 수 있었다.

검색 성능 향상에 대한 이유는 PdR-트리가 영역정보를 저장하고 있기 때문에 전체 노드의 방문회수를 기존의 방법보다 감소시켜 전체적인 검색성능이 향상되었다고 분석된다. 종합경과시간의 경우 데이터 크기변화에 따른 실험에서 피라미드 기법에 비해 최대 60% 향상하였고, 구형 피라미드 기법에 비해 최대 32%의 향상을 이룰 수 있었다. 또한 차원변화에 따른 실험에서 피라미드 기법에 비해 최대 74%의 향상을 구형 피라미드 기법에 비해 최대 31% 검색속도가 향상되었다. 실제 데이터를 이용하여 질의범위 변화에 따른 실험에서도 페이지 액세스회수가 최대 89% 그리고 질의범위가 0.2일 때 종합경과시간이 최대로 감소하였음을 알 수 있었다.

참 고 문 헌

[1] J. M. Hellerstein, E. Koutsoupias, C. H. Papadimitriou, "On the Analysis of Schemes," ACM PODS, pp.249-256, 1997.

[2] R. Weber, H. J. Schek, S. Blott, "A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. 24th VLDB Conference," NY, USA, 1998.

[3] C. C. Chang, S. Y. Lee, "Retrieval of Similar Pictures on Pictorial Databases," Pattern Recognition 24, pp.675-680, 1991.

[4] K. Aizawa, H. Harashima, "Model based image coding," SPIE/IS, Electronic Imaging, Vol.4-1, pp.1-2, 1994.

[5] A. D. Bimbo, P. Pala, S. Santini, "Image Retrieval by Elastic Matching of Shapes and Image Patterns," Proceeding of the itn'l conf. on multimedia computing and systems, Japan, 1996.

[6] S. Berchtold, C. Bohm, H-P. Kriegel. "The Pyramid-Technique : Towards Breaking the Curse of Dimensionality," Proc. ACM SIGMOD Int. Conf. on Management of Data, 1998.

[7] R. Weber, S. Blott, "An Approximation-Based Data Structure for Similarity Search," Esprit Project Hermes, technical report Oct, 1997.

[8] A. Hinneburg, D. A. Keim. "Optimal Grid_Clustering : Towards Breaking the Curse of Dimensionality in High-Dimensional Clustering," proceedings of the 25th VLDB Conference, Edinburgh, Scotland, 1999.

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

[10] N. Katayama, S. Satoh, "The SR-tree : An Index Structure for High-Dimensional Nearest Neighbor Queries," ACM SIGMOD, 1997.

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

[12] 조범석, 박영배, "색상과 모양특징을 이용한 효율적인 이미지 검색기법", 제27회 한국정 보과학회 춘계학술대회발표논문집, 2000.

[13] D. A. White, R. Jain, "Similarity Indexing with the SS-tree," IEEE, 1995.

[14] L. I. Lin, H. V. Jagadish, C. Faloutsos, "The TV-tree-an index structure for high-dimensional data," VLDB, 1995.

[15] 이혜명, 임채명, 박영배, "시계열 패턴을 위한 dR-트리", 한국정보과학회, 가을학술발표논문집(A), 1996.

[16] P. Ciaccia, M. Patella, P. Zezula, "M-tree : An Efficient Access Method for Similarity Search in Metric Spaces," 23rd VLDB Conference, 1997.

[17] A. Henrich, "The LSDh-tree : An Access Structure for Feature Vectors," ICDE, 1998.

[18] 이동호, 송용준, 김형주. "SCARLET : 웨이브릿 변환을 이용한 내용기반 이미지검색시 스템의 설계 및 구현", 한국정보과학회 논문지(C), 3(4), 1997.

[19] 이동호, 정진완, 김형주. "고차원 데이터의 유사성 검색을 위한 효율적인 색인기법", 한 국정보과학회 논문지(B), 제26권 제11호, 1999.

조 범 석

e-mail : jbseok@hanmail.net
1986년 명지대학교 전자계산학과 (공학사)
1988년 명지대학교 대학원 전자계산학과 (공학석사)
1995년 명지대학교 대학원 컴퓨터공학과 (박사수료)
1996년~1999년 태성대학 컴퓨터정보과 교수
1999년~현재 명지대학교 컴퓨터공학과 데이터베이스 연구실
관심분야 : 멀티미디어 검색기법, 데이터 베이스, 데이터 마이닝

박 영 배

e-mail : parkyb@mju.ac.kr
1974년 동아대학교 공학사
1980년 연세대학교 공학석사
1993년 서울대학교(컴퓨터공학)공학박사
1974년~1981년 한국전력공사 (현)정보처리처과장
1990년~1992년 명지대학교 전자계산소 소장
1997년~현재 산업대학원 원장 재직
1981년~현재 명지대학교 컴퓨터공학과 교수 재직중이며, 데이터베이스, 자료구조, 화일처리 등을 강의
관심분야 : 인터넷 DB, 멀티미디어 DB, 시스템통합(SI) 등