GB-색인 : 고차원 데이타의 복합 유사 질의 및 적합성 피드백을 위한 색인 기법

(GB-Index: An Indexing Method for High Dimensional Complex Similarity Queries with Relevance Feedback)

차 광 호 †

(Guang-Ho Cha)

요 약 멀티미디어 데이타베이스와 같은 고차원 응용에서 유사 색인과 검색은 어려운 문제이며, 특히, 다수의 특성을 함께 색인하는 경우에는 더욱 어렵다. 본 논문에서는 고차원 이미지 데이타베이스에서 복합 유사 질의 및 적합성 피드백을 효율적으로 처리하기 위한 새로운 색인 기법인 GB-색인을 제시한다. GB-색인은 각 특성 차원을 독립적으로 처리함으로써 다수의 특성과 다수의 질의 객체를 유연하게 제어한다. 아울러, 비트맵 색인을 통해 데이타베이스에 있는 모든 객체를 비트맵의 집합으로 표현하여 질의를 효율적으로 처리한다. GB-색인의 기술적인 주된 공헌은 다음과 같다: (1) 고차원 데이타를 위한 효율적인 색인, (2) 효율적인 복합 유사 질의 처리, (3) 적합성 피드백을 위한 분리형 질의의 효과적 처리. 실험 결과에 따르면 GB-색인은 순차 탐색 및 VA-파일에 비해 큰 성능 향상을 보였다.

키워드 : GB-색인, 고차원 색인, 비트맵 색인, 유사 질의, 적합성 피드백

Abstract Similarity indexing and searching are well known to be difficult in high-dimensional applications such as multimedia databases. Especially, they become more difficult when multiple features have to be indexed together. In this paper, we propose a novel indexing method called the GB-index that is designed to efficiently handle complex similarity queries as well as relevance feedback in high-dimensional image databases. In order to provide the *flexibility* in controlling multiple features and query objects, the GB-index treats each dimension independently. The *efficiency* of the GB-index is realized by specialized bitmap indexing that represents all objects in a database as a set of bitmaps. Main contributions of the GB-index are three-fold: (1) It provides a novel way to index high-dimensional data; (2) It efficiently handles complex similarity queries; and (3) Disjunctive queries driven by relevance feedback are efficiently treated. Empirical results demonstrate that the GB-index achieves great speedups over the sequential scan and the VA-file.

Key words : GB-index, high-dimensional index, bitmap index, similarity query, relevance feedback

1. 서 론

인터넷을 통해 시각적 데이타가 증가하고, 그러한 시각적 데이타에 대한 색인과 검색의 어려움으로 인해 내용 기반 이미지 검색(content-based image retrieval: CBIR)에 대한 활발한 연구가 진행되고 있다. CBIR은 이미지에서 추출한 색상, 질감, 모양과 같은 특성에 기반하여 검색을 지원한다. 사용자가 이러한 특성을 통해 표현된 질의를 하면 CBIR 시스템은 질의 객체에 대한 유사성(similarity)에 따른 순서대로 정렬된 관련 객체의 리스트를 반환한다.

CBIR에 관한 연구가 10년 이상 활발히 진행되어 왔지만, 특성 벡터의 고차원성과 다수의 객체와 다수의 특성을 갖는 다중-객체 다중-특성 유사 질의(multi-object multi-feature similarity query)의 복잡성이라는 근본적인 어려움 때문에 멀티미디어 데이타의 색인과 검색 성능은 아직도 만족스럽지 못하다. 또한 검색 효과를 높이기 위한 의미적 검색의 일환으로 적합성 피드백(relevance feedback)이 요구되고 있다. 적합성 피드백은 사용자의 피드백에 기초하여 각 단계에서 검색 결과를 갱신하는 반복적인 과정이다[1]. 적합성 피드백에서

· 본 연구는 정보통신부의 정보통신기초기술 연구지원사업(과제번호: 04-기초-052)으로 수행한 연구 결과입니다.

† 종신회원 : 서울산업대학교 컴퓨터공학과 교수
ghcha@snut.ac.kr

논문접수 : 2004년 10월 5일
심사완료 : 2005년 5월 3일

는 보통 복합 유사 질의를 사용하게 되어 분리형(disjunctive) 질의를 처리해야 한다.

이 논문은 고차원 유사 색인과 적합성 피드백을 갖는 복합 유사 질의 문제라는 CBIR의 두 가지 근본 문제의 해결을 위한 새로운 색인 기법인 GB-색인(Grid Bitmap Index: GB-index)을 제시한다. GB-색인은 복합 유사 질의 처리에 특히 효과적이며 그 구조와 알고리즘이 매우 간단하고 효율적이다.

1.1 고차원 복합 유사 질의 문제

1.1.1 고차원 색인 문제

고차원 색인 문제는 차원이 높아질수록 검색 성능이 급격히 하락하는 차원의 저주(dimensionality curse) 문제로 잘 알려져 있다. 유사 질의 또는 최근접 질의 문제는 효율적인 색인 기법이 개발되어 있는 저차원의 경우에는 많은 해결책이 있지만 차원이 대략 10에 가까워질 때 대부분의 색인 기법들은 범위 질의나 유사 질의의 수행을 위해 거의 모든 데이타를 접근해야 하는 걸로 알려져 있다. [2]는 고차원에서는 L_p-norm같이 널리 사용되는 유사 함수에서 주어진 질의 점에서 가장 가까운 이웃과 가장 멀리 있는 이웃간의 상대적인 거리가 별 차이가 없어서 인접(또는 유사)하다는 개념 자체가 의미가 없다고 기술하고 있다. 실제로 유사성의 관점에서 이러한 상대적인 차이의 결여는 바람직하지 않다.

Hinneburg 등은 [3]에서 최근접 검색의 품질에 대해 언급하고, 주어진 질의에 대해 관련된 특성(또는 차원)을 규정하고 이들 특성을 이용하여 가장 유사한 객체들을 결정하였다. 그러나 실제로 주어진 질의 점의 이웃에서 최선의 차원들의 결합을 찾기가 매우 어려워서 이 접근법은 순차 탐색보다 속도가 느리다.

Aggarwal과 Yu는 [4]에서 가장 가까운 이웃과 가장 멀리 있는 이웃간의 상대적인 차이가 없는 이유로써 차원이 높아질수록 데이타 공간의 희박성때문에 가장 가까운 객체 간에도 모든 차원에 대해 가까울 수는 없고, 특정 차원에 대해서는 멀리 떨어져있게 되어 결국 L_p-norm같이 모든 차원에 대해 거리의 합을 구하는 거리 함수로는 가까운 객체 간에도 거리가 멀어서 상대적인 거리의 차를 나타내기 어렵다고 하였다.

차원이 높아질수록 어떤 객체가 모든 차원에서 주어진 질의 점에 가까워질 확률은 줄어든다. 즉, 가장 가까운 객체라 하더라도 모든 차원에서의 거리가 균등 분포를 할 확률은 매우 작다. 따라서 L_p-norm같은 거리 함수의 경우에 거리를 결정하는 지배적인 요소는 가장 먼 거리를 결정하는 차원이 된다.

예를 들어, 그림 1에서 질의 점은 +로 표시하고, 세 개의 최근접 이웃을 찾는다고 하자. 최소한의 검색 공간은 최근접 이웃인 c, d, h를 포함하는 '검색 공간'으로 표시한 원이 될 것이다. 이 검색 공간은 세 개의 최근접 이웃 중에서 질의 점과 객체를 가장 멀리 떨어지게 하는 차원에 의해 결정되고, 보통 그 거리는 고차원에서는 전체 데이타 공간을 점유할 만큼 매우 크다. 따라서 이러한 환경에서 검색 공간의 일부를 잘라내기(pruning)란 매우 어렵다. 따라서, 관련된 차원, 즉, 질의 점과 데이타 점이 충분히 가까운 차원만 고려하여 유사도를 측정할 수 있는 차원을 선택할 필요가 있다. 이러한 개념을 색인에 적용하기 위해 관련된 차원에서만 색인과 검색을 수행할 수 있도록 각 차원을 독립적으로 다룰 수 있어야 한다.

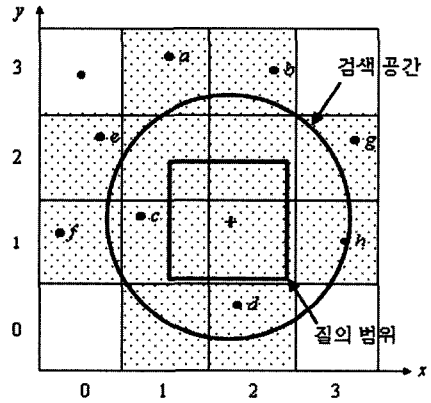

그림 1 최근접 질의에서 변환된 범위 질의

수행 성능의 관점에서, 검색 공간을 어느 정도까지 제한하기 위해 k-최근접(k-nearest neighbor: k-NN) 질의를 범위(range) 질의로 변환하는 것을 살펴보자. 그림 2는 이러한 변환이 의미가 있음을 보여준다. 점 d와 e는 질의 점 +의 네번째와 다섯번째로 가까운 이웃이다. 순위에 기반한 순서의 관점에서 d와 e는 아주 가까운 이웃이지만 실제로 그들은 그렇게 관심있는 이웃이라고 할 수는 없다. k-최근접 질의에서 범위 질의로의 변환은 검색 공간을 어느 정도 제한하는 효과를 가져온다. 만약 질의 범위를 적절하게 결정할 수만 있다면 검색 시간을 크게 줄일 수 있을 것이다.

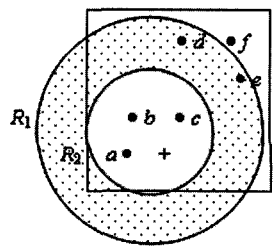

그림 2 순위에 기반한 거리 측정의 문제점

그림 1에서 '질의 범위'로 표현된 사각형이 적절히 결정된 질의 범위라고 가정하자. 각 차원에서의 근접 임계치(proximity threshold)를 미리 결정하고 각 차원을 몇 개의 영역으로 (그림 1에서는 각각 4 영역으로) 미리 나누었다. 만약 두 객체가 차원 i에서 같은 영역을 공유하고 있으면 해당 객체는 차원 i에서 유사하다고 가정한다. 각 차원에서 질의 범위와 겹치는 범위에 있는 객체들은 질의 객체와 유사한 객체라고 간주한다. 그러나, 고차원에서는 모든 차원에서 질의 범위와 겹치는 객체는 거의 존재하지 않을 것이다. 그림 1에서, 차원 x에서 '질의 범위'와 겹치는 범위에 있는 객체는 a, b, c, d이고, 차원 y에서는 c, e, f, g, h이다. c만이 양 쪽 차원에서 다 '질의 범위'와 겹치는 범위에 놓이는 객체이다. 이러한 상황에서, 만약 k 개의 최근접 이웃을 다 찾으려면 고차원에서는 질의 구(검색 공간)의 크기가 충분히 커야 한다. 따라서, 모든 차원을 한꺼번에 다루는 전통적인 색인 기법이 검색 공간을 잘라낼 확률은 극히 희박하다. 따라서 관련된 차원만 고려할 수 있도록 각 차원을 독립적으로 다루는 것이 필요하다. 이러한 아이디어를 가지고, 본 논문에서는 각 차원을 독립적으로 다루는 색인 구조와 그 구조 위에서의 유사성 함수를 개발한다. 이러한 색인 구조는 필요한 차원만 선택적으로 사용하는 융통성을 갖는다.

1.1.2 복합 유사 질의 문제

복합 유사 질의(complex similarity query, 또는, 단순히 복합 질의)는 다중-객체 다중-특성 (multi-object multi-feature) 질의이다. 즉, 다수의 질의 객체와 다수의 특성 벡터를 사용하는 유사 질의이다[5]. 복합 질의 Q는 n 개의 참조 객체 $\{Q_1, Q_2, ..., Q_n\}$로 구성되고, 각 참조 객체 Q_i에 대해 임의의 개수의 특성 벡터 $(F_{i,1}, ..., F_{i,m[i]})$가 존재한다. 특성 벡터 $F_{i,j}$는 해당 특성 공간에서의 점 $(v_{i,j,1}, ..., v_{i,j,f[i,j]})$으로 표현된다(그림 3 [5]). 다음은 단일-객체 다중-특성(single-object multi-feature) 질의의 한 예이다:

(*color* = '*red*') ∧ (*texture* = '*smooth*') ∨ (*shape* = '*round*').

이러한 형태의 복합 질의를 처리하기 위해 한 부분 시스템은 *color*를 처리하고, 다른 부분 시스템은 *texture*를, 또 다른 부분 시스템은 *shape*를 처리한다고 가정할 수 있다. 그러면 최종 결과는 세 개의 서로 다른 부분 시스템의 부분 결과를 결합하여 형성된다. [6]에서 Fagin은 이러한 종류의 복합 질의 $F(A_1, ..., A_m)$에 대해 부분 시스템 i가 부분 질의 A_i를 평가하고 각각의 부분 결과를 결합하여 전체 질의 F에 대한 k 개의 답을 반환하는 A_0 알고리즘을 제안하였다. 이러한 형태의 부울(Boolean) 질의를 처리하기 위해 Cha와 Chung은 [7]에서 다른 형태의 부분 질의로 부터 나온 부분 결과를 결합하는 알고리즘을 제시하였다. 그런데 [6]과 [7] 같이 부분 질의를 개별적으로 처리하고 부분 결과를 결합하는 형태의 접근법은 비용이 많이 든다. 예를 들어, $F(A_1, ..., A_m)$를 계산하기 위한 A_0 알고리즘의 비용은 $O(N^{(m-1)/m}\ k^{1/m})$이다. 여기서 N은 데이타베이스에 있는 객체의 수이다.

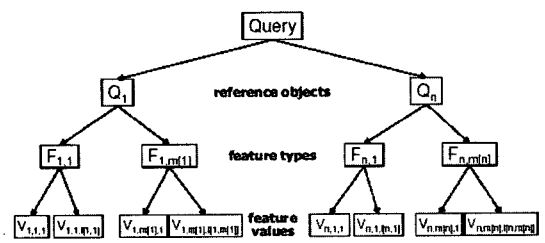

그림 3 복합 질의 모델

본 논문에서는 하나의 효율적인 GB-색인을 통해 이 복합 유사 질의 문제를 푼다. GB-색인은 각 특성(또는 차원)을 독립적으로 처리하여 효율적이고 자연스런 방법으로 복합 유사 질의에서 k 개의 최근접 이웃을 반환한다. 여기서 '효율적'이라는 것은 고차원 색인 기법의 성능을 측정하는 비교의 척도로 사용되는 순차 탐색이나 VA-파일[8]보다 디스크 접근 회수 및 처리 속도 면에서 우수함을 의미하고, '자연스럽다'는 것은 복합 질의 처리를 위해 각 특성을 하나의 특성으로 통합 처리하거나 또는 개별적 특성에 대해 각기 수행한 뒤에 부분 결과를 통합해야 하는 기존의 기법들과 달리, 원하는 특성에 대해서 선택적으로 k-최근접 이웃을 구할 수 있음을 의미한다.

1.1.3 적합성 피드백 문제

적합성 피드백에서는 기본적으로 다중 객체의 복합 유사 질의를 사용하게 되어 분리형 질의를 처리해야 한다. 즉, 적합성 피드백을 위해서는 시스템이 반환하는 k 개의 이미지 중에서 사용자가 긍정적으로 판단하는 m 개의 이미지를 선정하여 이 m 개의 이미지에 대한 k-최근접 질의를 수행해야 하는데 GB-색인은 효율적인 복합 유사 질의 처리와 집계(aggregate) 거리 함수를 통해서 m 개의 이미지를 사용하는 분리형 질의를 효율적으로 수행한다.

2. 관련 연구

최근의 데이타베이스 연구에서, 고차원 데이타를 위한 많은 색인 기법들이 제시되었다. 대부분의 연구들은 트리 구조의 색인 기법의 한계로부터 출발하였고, 차원의 저주를 해결하려고 하였다. 최근에, 많은 응용에서 근사(approximate) 해만으로도 충분하다는 원칙에 기반하여

근사 최근접 이웃을 구하려는 시도가 많이 있다[9-13]. 실제로, 객체를 특성 벡터로 사상시키는 작업 자체도 경험적(heuristic)이므로 근사 최근접 이웃을 찾는 것도 때때로 충분하다. 또한 많은 멀티미디어 응용에서 유사성의 개념도 주관적일 수 있고, 사용자의 질의 개념을 명확히 하는 것도 종종 어렵다[14].

[8]에서 기술하였듯이, 전통적인 데이타 공간 분할에 기반한 다차원 색인 기법들은 검색 공간을 잘라내기 매우 어렵다. 따라서 X-트리[15], SS-트리[16] 같은 대부분의 다차원 색인 기법들은 순차 탐색보다 검색 성능이 우월하지 못하다.

VA-파일[8] 및 LPC-파일[17] 같은 벡터 근사 기반 접근법은 고차원에서는 전통적인 다차원 색인 트리들이 순차 탐색을 능가할 수 없고, 따라서 순차 탐색이 불가피하다는 관점에서 출발한다. 따라서 데이타를 근사(압축)하고 근사된 데이타를 순차적으로 탐색한다. 이 벡터 근사 기반 접근법의 주된 약점은 압축된 데이타를 순차 탐색하므로 압축된 비율 이상의 성능 개선을 기대할 수 없다는 것이다.

차원 축소 기법[9,10]과 ε-근사 최근접 해법[11-13]은 다른 기법과 함께 부차적으로 사용할 수 있다.

GC-트리[18]는 벡터 근사 기법과 트리 기반의 다차원 색인 기법의 장점을 결합한다. 그러나 검색 공간의 제거 비율이 데이타 분포에 크게 좌우된다. VQ-색인[19]은 누적된 질의 역사와 K-평균(K-means) 알고리즘[20]을 사용하여 데이타 집합을 클러스터들로 분할하고, 질의 점이 놓이는 클러스터 내의 점들만 검색한다. 그러나 고차원에서는 데이타 공간의 평균 밀도가 아주 낮아서 클러스터를 찾기가 매우 어렵고 의미가 없을 수 있다[21].

본 저자가 파악하기로는 아직까지 복합 유사 질의를 효과적으로 처리할 수 있는 색인 기법은 거의 없다. 대부분의 기법들은 고정된 수의 차원을 갖는 데이타를 다루고, 따라서 특성 또는 차원의 수의 변화에 적응하지 못한다. 그러나 사용자마다 사용하는 질의 특성이 다르므로 가변적인 수의 특성으로 질의할 수 있는 기능을 제공하는 것이 CBIR 시스템의 효과를 높이는데 중요하다. GB-색인은 가변적인 수의 특성을 갖는 데이타를 효율적으로 다루며, 논리적인 AND/OR로 결합된 특성들을 비트 단위의 AND/OR를 수행하므로 데이타 접근 횟수는 차원에 반비례하고, 이것은 차원이 높아질수록 데이타 공간의 밀도가 낮아지는 현상과 일치한다.

3. GB-색인

고차원에서는 공간의 희박성 때문에 대부분의 점들이 멀리 떨어져 있고, 따라서 GB-색인에서는 근접(proximity) 범위를 제한하기 위해 k-최근접 질의를 범위 질의로 변환한다.

3.1 k-최근접 질의에서 범위 질의로의 변환

고차원 색인과 복합 유사 질의의 색인을 위해 GB-색인은 k-최근접 질의를 범위 질의로 변환한다. 이 변환은 성능 측면 외에도 검색 범위를 어느 정도 제한하는 역할도 수행한다.

k-최근접 질의에서 범위 질의로의 변환은 두 질의의 근본적인 차이 때문에 간단히 수행되지 않는다. k-최근접 질의의 경우, 검색 공간을 미리 결정할 수 없고, 지금까지 찾은 k 번째 최근접 이웃까지의 거리에 기초하여 동적으로 결정되는 반면에 범위 질의의 경우, 질의할 때 이미 질의의 범위가 결정된다. 따라서 k 개의 최근접 이웃을 찾기 위해서는 d-차원의 범위 질의에서 각 차원에서의 범위를 결정해야 한다. 그러나 이 범위를 결정하기란 어렵고, 만약 [22]에서와 같은 기법을 사용하여 질의 범위를 결정한다면 데이타 공간의 대부분을 차지할 만큼 큰 범위가 될 것이다.

범위 질의를 효율적으로 수행하기 위해서, 데이타를 각 차원에서 K 개의 구간(interval)으로 나눈다. 여기서 각 구간은 각 차원에서 관련된 데이타를 포함하는 클러스터를 나타낸다. 따라서 본 논문에서는 구간과 클러스터를 같은 의미로 사용한다. 이렇게 데이타를 분류하는 것은 각 특성 차원에서 유사도에 대한 임계치를 계산하는 효과를 갖는다. 즉, 만약 두 개의 데이타가 같은 구간에 포함된다면 해당 구간에서 유사하다고 간주한다. 구간 내부의 데이타 간의 유사성은 높고, 구간 간의 유사성은 낮도록 구간을 설정하는 것이 바람직하다. 본 논문에서는 데이타를 K 개의 구간으로 분류하기 위해 K-평균 알고리즘[20]을 사용한다. K-평균 알고리즘은 간단하고 효율적인 방법이나 다른 정교한 알고리즘을 채택할 수도 있다.

K-평균 알고리즘에서 K의 선택은 데이타 집합의 크기, 데이타 분포, 차원의 수 등 다양한 성능 파라미터에 영향을 받는다. GB-색인에서는 경험적으로, 즉, 실험 결과에 따라 한 구간에 너무 적은 수의 데이타가 들어가지 않도록 차원에 따라 $8 \le K \le 16$으로 선정하였다.

데이타를 각 차원에서 K 개의 구간으로 분류한 뒤에, 차원 i상의 j 번째 구간 $I_{i,j}$에 대해 비트맵 $b_{i,j}$을 생성한다. 비트맵 $b_{i,j}$는 어느 데이타가 구간 $I_{i,j}$에 있는지를 나타내는 이진 정보를 포함한다. 차원 i와 두 객체 $x = (x_1, x_2, \ldots, x_d)$, $y = (y_1, y_2, \ldots, y_d)$에 대해서, 만약 x_i와 y_i 가 같은 구간 $I_{i,j}$에 속한다면, 두 객체는 차원 i 상에서 유사하다고 간주한다. 차원 i에 대해서, 질의 객체가 놓이는 구간에 함께 놓이는 객체들을 포함하는 집합을 나타내기 위해 기호 S_i를 사용한다.

예를 통해서 질의 범위를 결정하는 과정을 보인다. 차원 i에 대해, 범위 질의의 범위는 질의 객체가 놓이는 구간 $I_{i,j}$이다. 그림 1을 보면, 질의 점은 +로 표시되어 있고, 구간 $I_{x,2}$와 $I_{y,1}$는 각각 차원 x, y에 대한 질의 범위를 나타낸다. 질의 점에 대한 유사 객체의 후보들은 질의 점이 놓이는 구간에 의해 결정된다. 차원 x의 경우, 객체 b와 d가 후보이다, 즉, S_x = {b, d}. 차원 y의 경우, 객체 f, c, h가 후보이다, 즉, S_y = {f, c, h}. 최종 후보를 결정하기 위해, S_x와 S_y의 공통 부분을 추출한다. 만약 S_x와 S_y에 공존하는 객체의 수가 k보다 크거나 같으면 공통 부분에 있는 객체가 k-최근접 질의의 최종 후보가 된다. 그렇지 않을 경우, $I_{i,j}$의 인접한 구간 의 양쪽으로 질의 범위의 크기를 확장한 후에 다시 공통 부분의 객체를 추출한다. 예를 들어, 그림 2에서, S_x와 S_y에 공존하는 객체의 수가 k (예를 들면, 3)보다 적으므로 질의 범위를 차원 x에서 $I_{x,1}$, $I_{x,2}$, $I_{x,3}$를 포함하는 구간으로, 차원 y에서 $I_{y,0}$, $I_{y,1}$, $I_{y,2}$를 포함하는 구간으로 확장한다.

3.2 유사 함수(similarity function)

유사 함수가 유연성을 갖기 위해서는 유사 함수가 각 차원을 독립적으로 다룰 수 있도록 정의해야 한다. 유사 임계치(similarity threshold)는 k-최근접 질의에서 변환된 범위 질의의 구간에 의해 미리 정의된다. 즉, 정의된 질의 범위 내의 객체들은 유사한 범위 내에 있다고 가정한다.

각 차원에서 같은 구간에 있는 객체들이 다른 구간에 있는 객체보다 더 유사하도록 K-평균 알고리즘에 기초하여 데이타를 분류한다. 차원 i에 대해서, j 번째 구간 $I_{i,j}$를 규정하고 그 것의 하한 및 상한 경계 l_{ij}, u_{ij}를 구한다. 차원 i에서 두 객체 x = (x_1, x_2, ..., x_d)와 y = (y_1, y_2, ..., y_d)에 대해 (x_{ij}, y_{ij}) $\in$ [l_{ij}, u_{ij}]가 성립한다. 여기서 첨자 ij는 차원 i의 구간 j상에 해당 객체가 있음을 나타낸다. 이러한 상황에서 차원 i상의 두 객체 x, y에 대한 거리 함수 D_i를 다음과 같이 정의한다:

$$D_i(x_i, y_i) = \begin{cases} \dfrac{|x_i - y_i|}{u_{ij} - l_{ij}} & x_i\text{와 } y_i\text{가 같은 구간에 존재하는 경우} \\ 1 & \text{그렇지 않은 경우} \end{cases}$$

분모의 u_{ij} - l_{ij}는 서고 다른 구간 사이에서 거리를 정규화한다. 두 객체 x = (x_1, x_2, ..., x_d)와 y = (y_1, y_2, ..., y_d)에서 두 객체가 유사한 차원들의 집합은 질의 범위와 겹치는 구간을 갖는 차원들로 구성된다. 따라서, 차원 i에서 만약 x_i, y_i 둘 다 질의 범위 내의 구간에 속한다면, 두 객체는 차원 i에서 유사한 범위에 있다고 정의한다. 두 객체 x, y 간의 전체 거리는 다음과 같이 계산한다:

$$D(x,y) = \left[\sum_{i=1}^{d} w_i D_i^{\,p} \right]^{1/p}$$

여기서 w_i는 사용자에 의해 차원 i에 할당된 가중치이다. 차원 i에 대한 개별적인 거리가 0과 1 사이이므로 전체 거리 값은 0에서 Σw_i사이의 값은 갖는다. 본 논문의 실험에서는 p = 1로 할당하였다. 한 점 x와 다수의 질의 점의 집합 Q = {g_1, g_2, ..., g_m}간의 집계 거리 함수 D_g는 [23]에서와 같이 다음과 같이 정의한다:

$$(D_g(x))^{\alpha} = \frac{1}{\sum_{i=1}^{m} w_i} \sum_{i=1}^{m} w_i (D(x, g_i))^{\alpha}$$

여기서, 만약 α의 값이 크면, 가장 먼 거리가 $D_g(x)$에 대해 가장 큰 영향을 미치고, 반면에 α의 값이 작다면 가장 가까운 거리가 $D_g(x)$에 대해 가장 큰 영향을 미친다. 이 집합 거리 함수는 $\alpha < 0$이 성립하면 퍼지 (fuzzy) OR를, $\alpha > 0$이면 퍼지 AND를 나타낸다. 본 논문에서는 α = -5를 사용하였으며, w_i는 사용자에 의해 차원 i에 할당된 가중치이다.

3.3 비트맵 색인

GB-색인은 고차원 공간에서 한 특정 범위에 있는 객체의 효율적인 탐색 및 각 차원을 독립적으로 다루기 위해 비트맵 색인 기법[24]를 도입한다. 비트맵 색인은 비트 단위의 AND/OR 연산을 통해 원하는 객체를 빠르게 검색한다. 따라서 GB-색인은 이 비트맵 색인을 통해 복합 질의를 빠르고 자연스럽게 처리한다. 예를 들어, 사용자가 특정 차원에만 관심이 있을 경우, 비트맵 색인을 사용한다면 각 차원이 독립적으로 색인되어 있으므로 해당 차원에 대한 색인만 사용할 수 있다. (*texture* = '*smooth*') $\wedge$ (shape = 'round') 같은 복합 조건을 갖는 질의의 경우, 두 조건을 동시에 만족하는 객체는 각각의 조건을 만족하는 객체들을 구한 다음, 효율적인 비트 단위의 AND 연산자를 통해 구할 수 있다. 논리적인 AND/OR를 위한 비트 단위의 연산의 효율성 때문에 복합 질의의 k 개의 최근접 이웃을 구하는 GB-색인은 순차 탐색보다 훨씬 빠르다.

비트맵 색인은 원래 관계형 데이타베이스에서 다수의 속성을 위한 색인 기법으로 설계되었다. 비트맵은 단순히 비트들의 배열이다. 가장 간단한 형태로는, 릴레이션 r의 속성 A에 대한 비트맵 색인은 A가 취할 수 있는 각 값에 대해 하나의 비트맵으로 이루어진다. 각 비트맵은 릴레이션에 있는 객체의 수만큼의 비트로 이루어진다. 만약 i 번째 객체가 속성 A에 대해 값 v_j를 갖는다면, v_j에 대한 비트맵의 i 번째 비트는 1로 결정되고, 비트맵의 모든 다른 비트는 0로 결정된다. 그림 4는 한 릴레이션에 대한 비트맵 색인의 한 예를 보여준다.

비트맵 색인은 두 가지 중요한 장점을 제공한다. 첫째, 질의를 처리하기 위해 효율적인 비트 연산을 제공한다. 복합 질의 처리를 위해서는 다수의 비트맵을 비트 단위의 AND/OR 연산을 통해 결과를 얻을 수 있다. 예를 들어, 그림 4에서 *income-level* L2를 갖는 여성을 선택하는 질의의 경우, *gender* 값으로 F를 갖는 비트맵과 *income-level* 값으로 L2를 갖는 비트맵을 구하여 두 비트맵을 비트 단위의 AND 연산을 수행한 후, 값 1을 갖는 객체를 답으로 얻을 수 있다. AND/OR와 같은 비트 단위의 논리 연산은 하나의 CPU 명령으로 많은 레코드의 논리적 연산을 수행할 수 있으므로 매우 효율적인 연산이다. 또한 비트맵 색인은 전통적인 트리 형태의 색인보다 훨씬 콤팩트하고, 압축 기술의 사용에 적합하다. 비트맵 색인은 각 객체를 1 비트로만 표현하므로 보통 데이타베이스 크기에 비해 아주 작다.

비트맵 색인은 색인에 대한 키 값의 수가 적을 때, 즉, 각 속성이 취할 수 있는 값의 수가 적을 때 유용하다. 그러나 이미지나 비디오 객체의 특성들의 도메인은 크다. 따라서 이러한 멀티미디어 데이타베이스에서의 문제를 처리하기 위해, GB-색인은 각 특성 벡터의 차원의 도메인을 K-평균 알고리즘에 따라 몇 개의 구간으로 분할하고, 각 차원의 각 구간에 대해 하나의 비트맵을 생성한다. 이런 관점에서 우리는 본 논문의 색인을 GB-색인(grid bitmap index: GB-index)이라 명명하였다.

object no.	*name*	*gender*	*city*	*income-level*
0	John	M	New York	L1
1	Diana	F	San Jose	L2
2	Mary	F	Miami	L1
3	Peter	M	San Jose	L4
4	Kathy	F	New York	L3

*gender*에 대한 비트맵
M : 10010
F : 01101

*income-level*에 대한 비트맵
L1: 10100
L2: 01000
L3: 00001
L4: 00010
L5: 00000

그림 4 한 릴레이션에 대한 비트맵 색인의 예

3.4 GB-색인

GB-색인은 각 차원의 분류된 영역에 대한 비트맵 색인의 사용에 기초한다. 다른 두 차원에서의 두 영역의 교집합과 합집합 연산은 각각 비트 단위의 AND, OR 연산을 통해 수행한다.

3.4.1 색인 생성

GB-색인은 다음과 같은 과정을 따라 생성된다:

(1) 각 차원 i, $1 \le i \le d$,에 대해, K-평균 알고리즘을 사용하여 데이타를 K_i 개의 구간으로 분류한다. 차원 i에 대한 j 번째 구간을 $I_{i,j}$로 표시한다.

(2) 각 구간 $I_{i,j}$에 대해, 비트맵 $b_{i,j}$를 생성한다. 구간 $I_{i,j}$의 하한 및 상한 경계$[l_{ij}, u_{ij}]$를 저장한다. 만약 구간 $I_{i,j}$에 p 번째 객체가 존재하면 비트맵 $b_{i,j}$의 p 번째 비트를 1로 세트한다.

(3) 구간 $I_{i,j}$에 있는 객체에 대해, 비트맵 $b_{i,j}$와 함께 차원 i에 대한 실제 좌표를 유지한다.

(4) GB-색인은 단순히 이렇게 생성된 비트맵들의 배열과 각 구간 $I_{i,j}$에 대한 하한 및 상한 경계$[l_{ij}, u_{ij}]$ 및 각 차원 i에 대한 실제 좌표들의 리스트로 구성된다.

3.4.2 k-최근접 검색 알고리즘

특성 벡터 $Q = \{Q_1, Q_2, ..., Q_d\}$, 질의 가중치 $w = (w_1, w_2, ..., w_d)$, 찾고자 하는 객체의 갯수 k 를 입력으로 하는 k-최근접 질의 처리 알고리즘은 다음과 같다:

(1) 질의 Q와 각 차원 i, $1 \le i \le d$,에 대해서, Q_i가 놓이는 구간에 대한 비트맵 b_i를 계산한다.

(2) 비트맵 b의 모든 비트를 1로 세트시킨다.
b에 있는 1-비트의 수를 b_c라 한다.

(3) for $(1 \le i \le d)$ do {
  $b = b$ AND b_i.
  I_l, I_r를 각각 b_i의 왼쪽, 오른쪽에서 인접한 구간이라 한다.
  while $(b_c < k)$ do {
    $b = b$ OR (I_l의 비트맵) OR (I_r의 비트맵).
    I_l를 I_l의 왼쪽에서 인접한 구간이라 한다.
    I_l, I_r를 각각 앞서 계산한 I_l, I_r의 왼쪽, 오른쪽에서 인접한 구간이라 한다.
  }
}

(4) 최종 비트맵 b를 구한다.

(5) b에서 1로 세트된 비트에 해당하는 객체들의 집합 S를 구한다.

(6) S에 있는 객체에 대해서, 질의 객체와의 거리를 계산하고 그 순서에 따라서 k 개의 객체를 반환한다.

위의 k-최근접 질의 처리 알고리즘의 정확성은 다음과 같이 증명할 수 있다.

증명. 알고리즘의 행 (1)을 수행한 결과, 각 b_i에는 차원 i에서 Q와 같은 구간에 있는 객체들에 대한 비트가 1로 세트된다. 즉, 1로 세트된 비트에 해당하는 객체들이 찾고자 하는 k-최근접 이웃의 후보들이다. 행 (2)의 수행을 통해 우선 b의 모든 비트를 1로 세트시켜 놓는다. 알고리즘의 수행이 종료된 시점에서 b에서 1로 세트

된 비트에 해당하는 객체가 최종적인 k-최근접 이웃의 후보가 된다. 행 (3)의 for-루프를 통해 모든 차원에서 Q와 같은 구간에 있는 객체들을 비트맵 b에서 1로 세트시킨다. 그러나 그 구간에서 객체의 수가 k보다 적을 수 있으므로 while-루프를 통해 특정 차원에서는 Q가 놓이는 구간을 확장하여 b에서 1로 세트된 비트의 수가 항상 k보다 크도록 보장한다. 최종적으로 for-루프에서 나올 때는 b에서 1로 세트된 비트의 수가 k 이상이고, 그러한 비트에 해당하는 객체는 질의 Q와 모든 차원의 구간에서 공존한다. 최종적으로 Q와 해당 객체간의 거리를 계산하여 원하는 k-최근접 이웃을 구할 수 있다 (증명 끝).

그런데 GB-색인의 k-최근접 이웃 검색 알고리즘은 k-최근접 이웃 검색을 초사각형(hyper-rectangle) 형태의 범위 질의로 변환하여 k 개의 최근접 이웃을 찾기 때문에, 그림 5에서와 같이 빗금 친 사각형이 질의 범위로 결정되었다고 가정할 때, 점 y가 점 x보다 실제로 질의 Q 에 더 가깝지만 y는 질의 범위에 들지 못하므로 후보 대상에서 제외되는 경우가 발생할 수 있다.

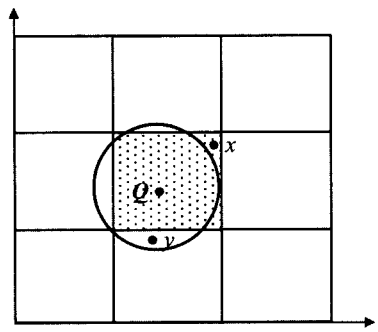

그림 5 질의 변환에 인해 인접한 객체가 제외되는 경우

예제 1. 데이타 공간의 차원을 3, 각 차원에서 구간의 갯수를 3, 데이타베이스에 있는 전체 데이타 객체의 갯수를 8이라고 가정하자. 그림 6에서, 각 차원에서 각 객체가 놓이는 구간에 체크 표시를 하였다. 예를 들어, 객체 o_0는 차원 0, 1, 2에서 구간 1, 0, 1에 각각 놓인다. 해당 구간에 대한 비트맵은 다음과 같다:

$b_{0,0}$: 01000100, $b_{0,1}$: 10101010, $b_{0,2}$: 00010001,

$b_{1,0}$: 10001101, $b_{1,1}$: 00110000, $b_{1,2}$: 01000010,

$b_{2,0}$: 00110000, $b_{2,1}$: 10000101, $b_{2,2}$: 01001010.

두 개의 질의 객체가 o_4와 o_6로 주어지고, o_4와 o_6로부터 3 개의 최근접 이웃을 찾고자 한다고 하자. 두 개의 질의 객체로 부터 다음과 같은 비트맵 색인을 계산할 수 있다:

b_0 = (10101010) OR (10101010) = 10101010

b_1 = (10001101) OR (01000010) = 11001111

b_2 = (01001010) OR (01001010) = 01001010.

그러면 다음 비트 벡터는 질의 객체 o_4와 o_6의 3 개의 최근접 이웃에 대한 후보를 선택하기 위한 최종 비트맵을 나타낸다:

$b = b_0$ AND b_1 = 10001010 (1-비트의 수 = 3)

$b = b$ AND b_2 = 00001010 (1-비트의 수 = 2).

1-비트의 수가 2 개이므로, b_2에 대응하는 구간은 확장되고, 새로운 비트맵 b_2 = 11001111를 구한다. 최종 비트맵은 다음과 같이 계산된다:

$b = b$ AND b_2 = 10001010 AND 11001111 = 10001010.

이것은 이미지 객체 o_0, o_4, o_6가 본 예제의 복합 질의의 후보임을 나타낸다. 집단 거리 함수 D_g를 사용하여 이들 객체에 대한 거리를 계산하고 가장 짧은 거리를 갖는 3 개의 객체를 반환한다.

구간
차원

	0			1			2		
oid	0	1	2	0	1	2	0	1	2
o_0		√		√				√	
o_1	√					√			√
o_2		√			√		√		
o_3			√		√		√		
o_4		√		√					√
o_5	√			√				√	
o_6		√				√			√
o_7			√	√				√	

그림 6 비트맵 색인의 예

3.4.3 삽입과 삭제

비트맵 색인에서의 동적인 삽입과 삭제는 효율적으로 지원된다. 객체의 삽입은 다른 객체의 순서에 영향을 주지 않으며, 단순히 색인 파일의 끝에 객체를 첨가시키면 된다. 객체 a가 GB-색인에 삽입될 경우, 각 차원 i의 각 구간의 비트맵에 새로 삽입되는 a를 위해 하나의 비트씩을 추가한다. 이 때, a가 위치하는 구간에서는 a에 해당하는 비트가 1로 세트되고, 나머지 구간에서는 0로 세트된다. 따라서, 특정한 하나의 객체의 삽입은 매우 간단하고, 따라서 효율적이다.

객체의 삭제도 마찬가지로 매우 간단하다. 각 비트맵에서 단순히 그 객체에 해당하는 비트를 삭제하면 된다. 그러나 실제로 해당 비트의 삭제는 비트맵에 빈 공간을 만들게 되므로 이를 채우기 위해 다른 객체에 해당하는 비트를 이동시키는 것은 비용이 많이 들므로 단순히 해

당 객체에 해당하는 비트를 0로 세트시킨다. 그러나 이 경우, 0-비트가 실제 객체의 값이 0인지 또는 존재하지 않는 객체를 나타내는 것인지 구분이 가지 않으므로, 이를 구분하기 위해 존재 지도(existence map)를 통해 존재하는 객체에는 1-비트를, 그렇지 않는 객체에는 0-비트를 사용한다. 질의 결과에서, 삭제된 객체에 해당하는 비트를 0로 세트시키기 위해 최종 비트맵을 존재 지도와 함께 AND 연산을 수행하면 된다. 이상과 같이 삽입, 삭제 알고리즘이 매우 간단하여 동적인 환경에서 효율적임을 알 수 있다.

4. 성능 평가

GB-색인의 실제적인 효과를 입증하기 위해 본 논문에서는 디스크 접근 실험과 실행 시간 실험을 수행하였고, GB-색인과 순차 탐색 및 VA-파일[8] 간의 성능을 비교하였다. 순차 탐색은 고차원 데이타 공간에서 대부분의 색인 기법보다 성능이 우수하므로 성능 평가를 위한 척도로 많이 사용된다[8]. 모든 실험은 512-MB 메모리의 인텔 펜티엄 IV 1.5 GHz 프로세서 상의 윈도우즈 XP 프로 운영체제 하에서 수행되었다.

실험을 위해 방송국 드라마 비디오 데이타베이스에서 추출한 MPEG-7 기술자를 갖는 12,861개의 이미지와 COIL(Columbia Object Image Library)-100 이미지 데이타베이스의 이미지를 사용하였다. 실험에 사용한 특성 벡터 추출을 위해 지배 색상(dominant color), 동종 질감(homogeneous texture), 경계 히스토그램(edge histogram), 색상 구조(color structure)로 형성되는 MPEG-7 기술자를 사용하였다. 지배 색상 기술자는 한 이미지 또는 한 영역에 있는 각 색상의 비율을 통해 결정한 지배 색상들의 집합이다. 이 기술자를 독립적인 차원들의 특성 공간으로 사상시키기 위해, 각 색상의 비율을 64 개의 빈(bin)으로 구성되는 색상 히스토그램의 빈 값으로 사용한다. 색상 구조 기술자는 이미지의 색상 분포와 지역적 공간 구조에 의해 정의되는 8-비트로 정량화되는 1차원 배열이다. 경계 히스토그램 기술자는 한 이미지의 지역 경계의 분포를 나타낸다. 한 이미지를 4 × 4 개의 부분 이미지로 분할하고 각 부분 이미지에 있는 경계를 5 종류로 분류하여 80 개의 빈으로 이루어진 히스토그램을 형성한다. 동종 질감 기술자는 주파수 채널의 집합으로부터 평균 에너지와 에너지 편차를 사용하여 영역의 질감을 표현한다. 이상의 네 가지 MPEG-7 기술자로 이루어진 특성 벡터는 532 차원의 벡터로 표현되고, 실험을 위해 0에서 1 사이의 값으로 정규화시켰다.

모든 실험에서 찾고자 하는 최근접 이웃의 수는 10, 15, 20으로 하였다. 즉, k = 10, 15, 20이다.

디스크 접근 횟수 실험에 사용된 페이지 크기는 4 KB로 하였다. 각 실험에서 100 개의 532-차원의 복합 k-최근접 질의를 수행하였고, 그 결과를 평균하였다. 다중-객체 질의의 성능을 평가하기 위해 질의 객체의 수 m 은 다음과 같은 값을 사용하였다: m = 1, 3, 5, 7, 9.

4.1 디스크 접근 횟수(단일-객체 다중-특성 질의)

복합 k-최근접 질의에 대해 전체 디스크 접근 횟수를 계산하였다. 그림 7은 k 개의 최근접 이웃을 찾기 위해 수행한 전체 디스크 접근 횟수를 나타낸다. GB-색인에 의해 수행된 디스크 접근 횟수가 VA-파일 및 순차 탐색에 의한 디스크 접근 횟수보다 훨씬 적음을 볼 수 있다. GB-색인에 의한 성능 개선은 특별히 고안된 비트맵 색인에 기인한다. 이 결과는 GB-색인이 고차원 응용에서 복합 유사 질의 처리에 성공적으로 사용될 수 있음을 보인다.

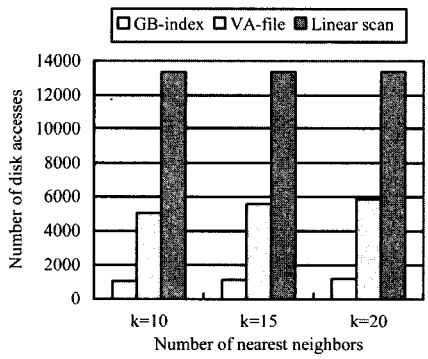

그림 7 단일-객체 다중-특성 질의에 대한 디스크 접근 실험

4.2 실행 시간 실험(단일-객체 다중-특성 질의)

GB-색인의 실제적인 유효성을 증명하기 위해 다수의 실행 시간 실험을 수행하였다. 그림 8은 GB-색인이 VA-파일 및 순차 탐색에 비해 큰 속도 향상을 나타냄

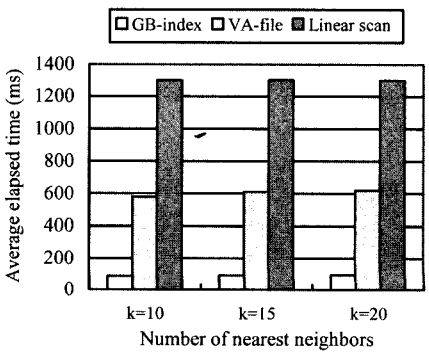

그림 8 단일-객체 다중-특성 질의에 대한 실행 시간 실험

을 보인다. 이러한 실험 결과로 볼 때, GB-색인은 VA-파일 및 순차 탐색과 비교할 때 매우 우수한 성능을 보인다고 결론지을 수 있다.

4.3 다중-객체 다중-특성 질의

그림 9와 10은 다수-객체 질의 실험 결과를 나타낸다: 여기서 사용한 질의 객체의 수는 각각 1, 3, 5, 7, 9 개이고, 특성의 수는 4 개, 전체 데이타 차원은 532이다. 그림 9와 10에서 나타난 것처럼 복합 질의 처리를 위한 디스크 접근 횟수와 실행 시간 모든 면에서 GB-색인은 VA-파일 및 순차 탐색에 비해 매우 우수한 성능을 보인다.

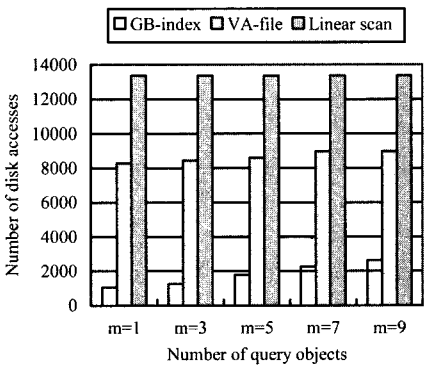

그림 9 다중-객체 다중-특성 질의에 대한 디스크 접근 실험(k = 15)

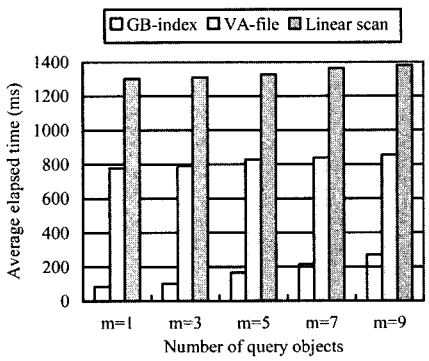

그림 10 다중-객체 다중-특성 질의에 대한 실행 시간 실험(k = 15)

4.4 실험 결과에 대한 논의

그림 7-10에 표시한 것처럼 디스크 접근 회수와 실제 실행 시간 비교에서 GB-색인은 순차 검색에 비교하여 10 배 이상, VA-파일에 비해 4-5 배 정도의 우월성을 보인다. VA-파일은 데이타를 근사하여 저장하고 근사된 전체 데이타를 순차 접근으로 읽어야 하므로 데이타베이스의 크기에 큰 영향을 받는다. 아울러, 질의 처리의 두 번째 단계에서 k-최근접 이웃의 후보들에 대한 실제 데이타를 읽어야 한다. 이 단계에서는 각 데이타를 임의 접근 해야 하기 때문에 하나의 데이타 객체에 대한 접근도 한 번의 디스크 임의 접근에 해당되어 그 비용이 비싸다.

GB-색인에서는 각 차원을 $8 \le K \le 16$ 개의 구간으로 구분하고 각 구간마다 해당 구간에 속하는 데이타 객체를 비트맵에 의해 1 비트로 표시한다. 따라서 질의 객체의 k-최근접 이웃의 후보는 주어진 질의 객체 부근의 각 차원에 대한 비트맵을 비트 단위의 AND 연산한 결과로 1로 세트된 객체들이며 이 연산은 간단하고도 굉장히 빠르다. 1로 세트된 후보 객체에서 질의 객체까지의 거리를 계산하여 우선 순위를 결정하면 되므로 GB-색인의 검색 성능은 VA-파일에 비해 훨씬 빠르다.

5. 결 론

본 논문은 고차원 이미지 데이타베이스에서 복합 유사 질의를 위한 GB-색인이라는 새로운 비트맵 색인 기법을 제시하였다. GB-색인의 설계 목표는 (1) 고차원 이미지 데이타베이스에서 k-최근접 질의의 효율적인 처리와 (2) 적합성 피드백을 갖는 복합 유사 질의의 효과적인 지원의 두 가지이다. 이 목표를 이루기 위해 각 특성 차원을 독립적으로 처리할 수 있는 유사 함수를 개발하였고, 이 유사 함수에 기반하여 GB-색인을 개발하였다. GB-색인의 비트맵 색인은 VA-파일과 순차 탐색에 비해 k-최근접 질의 처리를 위한 디스크 접근 회수와 실행 속도 면에서 높은 성능 향상을 보였다. 이러한 성능 향상은 복합 특성을 가지고 관련된 객체를 찾기 위해 효율적인 비트 단위의 AND/OR 연산을 수행하기 때문이다. 지금까지 복합 유사 질의를 처리하기 위한 연구는 거의 없었는데 이 논문에서 GB-색인은 매우 효율적이고 자연스러운 방법으로 복합 유사 질의를 처리함을 보여주었다. 나아가, GB-색인은 특성의 개수 변환, 즉, 데이타 공간의 차원의 변환에 쉽게, 유연성있게 대처할 수 있으므로 가변적인 개수의 특성을 갖는 데이타의 색인에 효과적으로 사용될 수 있다. 아울러, 삽입, 삭제를 위한 알고리즘이 매우 단순하여 동적인 환경에서의 사용에도 적합하다.

참 고 문 헌

[1] Y. Rui et al., "Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval," *IEEE Trans. Circuits and Video Technology*, 8(5), 644-655, 1998.

[2] K.S. Beyer et al., "When is nearest neighbor

meaningful?" *Proc. ICDT*, 217-235, 1999.
[3] A. Hinneburg, C.C. Aggarwal, and D.A. Keim, "What is the nearest neighbor in high dimensional spaces?," *Proc. VLDB*, 506-515, 2000.
[4] C.C. Aggarwal and P.S. Yu, "The IGrid Index: Reversing the Dimensionality Curse for Similarity Indexing in High Dimensional Space," *Proc. ACM SIGKDD*, 119-129, 2000.
[5] K. Böhm, M. Mlivoncic, H.-J. Schek, and R. Weber, "Fast Evaluation Techniques for Complex Similarity Queries," *Proc. VLDB*, 211-220, 2001.
[6] R. Fagin, "Combining Fuzzy Information from Multiple Systems," *Proc. ACM PODS*, 216-226, 1996.
[7] G.-H. Cha and C.-W. Chung, "Object-Oriented Retrieval Mechanism for Semistructured Image Collections," *Proc. ACM Multimedia*, 323-332, 1998.
[8] R. Weber, H.-J. Schek, and S. Blott, "A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces," *Proc. VLDB*, 194-205, 1998.
[9] K. Chakrabarti and S. Mehrotra, "Local Dimensionality Reduction: A New Approach to Indexing High Dimensional Spaces," *Proc. VLDB*, 89-100, 2000.
[10] K.V.R. Kanth, D. Agrawal, and A. Singh, A., "Dimensionality Reduction for Similarity Searching in Dynamic Databases," *Proc. ACM SIGMOD*, 166-176, 1998.
[11] S. Arya et al, "An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions," *JACM*, 45(6),. 891-923, Nov. 1998.
[12] P. Indyk and R. Motwani, "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality," *Proc. ACM STOC*, 604-613, 1998.
[13] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, "Efficient Search for Approximate Nearest Neighbor in High Dimensional Spaces," *Proc. ACM STOC*, 614-623, 1998.
[14] W.-C. Lai, C. Chang, E. Chang, K.-T. Cheng, and M. Crandell, "PBIR-MM: Multimodal Image Retrieval and Annotation," *Proc. ACM Multimedia*, 421-422, 2002.
[15] S. Berchtold, D.A. Keim, and H.-P. Kriegel, "The X-tree: An index structure for high-dimensional data," *Proc. VLDB*, 28-39, 1996.
[16] D. White and R. Jain, "Similarity indexing with the SS-tree," *Proc. ICDE*, pp. 516-523, 1996.
[17] G.-H. Cha, X. Zhu, D. Petkovic, and C.-W. Chung, "An Efficient Indexing Method for Nearest Neighbor Searches in High-Dimensional Image Databases," *IEEE Trans. on Multimedia*, 4(1), 76-87, March 2002.
[18] G.-H. Cha and C.-W. Chung, "The GC-Tree: A High-Dimensional Index Structure for Similarity Search in Image Databases," *IEEE Trans. on Multimedia*, 4(2), 235-247, June 2002.
[19] E. Tuncel, H. Ferhatosmanoglu, and K. Rose, "VQ-index: An index structure for similarity searching in multimedia databases," *Proc. ACM Multimedia*, 543-552, 2002.
[20] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proc. 5th Berkeley Symp. Math. Statist, Prob.*, 1:281-297, 1967.
[21] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," *Proc. ACM SIGMOD*, 94-105, 1998.
[22] S. Berchtold, C. Boehm, D.A. Keim, and H.-P. Kriegel, "A Cost Model for Nearest Neighbor Search in High-Dimensional Data Space," *Proc. ACM PODS*, 78-86, 1997.
[23] L. Wu, C. Faloutsos, K. Sycara, and T.R. Payne, "FALCON: Feedback Adaptive Loop for Content-Based Retrieval," *Proc. VLDB Conf.*, 297-306, 2000.
[24] P.E. O'Neil and D. Quass, "Improved Query Performance with Variant Indexes," *Proc. ACM SIGMOD*, 38-49, 1997.

차 광 호
정보과학회논문지 : 데이타베이스
제 32 권 제 3 호 참조