클러스터링 및 연속적 I/O를 이용한 이미지 데이타 검색 연구

김진옥[0], 황대준
성균관대학교 전기전자 및 컴퓨터공학부
jinny[0]@ece.skku.ac.kr, djhwang@skku.ac.kr

A study on the searching of images via clustering and sequential I/O

Jinok Kim[0], DaeJoon Hwang
Sungkyunkwan University
School of electrical and computer engineering

요 약

본 연구에서는 멀티미디어 데이터 검색에 클러스터링과 인덱싱 기법을 같이 적용하여 유사한 이미지끼리는 인접 디스크에 클러스터하고 이 클러스터에 접근하는 인덱스를 구축하여 검색이 빠르게 이루어지는 유사 검색방법을 제시한다. 이 연구에서는 트리 유사 구조의 인덱스 대신 해싱 방법을 이용하며 검색시 I/O시간을 줄이기 위해 오브젝트를 가진 클러스터 위치를 찾는데 한번의 I/O를 사용하고 이 클러스터를 읽기 위해 연속적인 파일 I/O를 사용하여 클러스터를 찾는 비용을 최소화한다. 클러스터인덱싱 접근은 트리 유사 구조와 임의 I/O를 사용한 내용기반의 이미지 검색보다 효율적인 검색 적합성을 보이며 연속적 I/O를 통해 검색 비용을 낮춘다.

1. 서 론

내용 기반의 이미지 검색 기술[1,2,3]에서는 이미지를 고차원 특징 벡터로 정규화 하여 고차원 공간의 근접지역에서 주어진 이미지의 벡터와 가장 유사한 벡터를 검색한다. 유사질의를 실행하기 위한 가장 단순한 방법은 모든 이미지를 읽고 질의 이미지까지의 거리를 계산하여 최근접 이미지를 선택한다. 하지만 이 방법의 실행시간은 데이터의 차원(D), 데이터집합의 크기(N), 알고리즘의 복잡도($O(ND)$)와 비례하기 때문에 검색공간을 줄임으로써 실행시간을 단축하는 트리구조(R-tree[6],R*-tree[7],SR-tree[8],K-d-b tree [9], TV-tree[10],X--tree[11],M-tree[12])가 제안되었다. 하지만 최근 연구결과[4,5]는 트리구조가 차원의 저주 문제를 야기시킨다는 것을 지적하고 있다. 근접 블록을 찾기 위해 인덱스구조를 왔다 갔다 하는 시간이 지나치게 많이 소요되며 유사 이미지를 찾는 과정에서 근접 블록 수가 데이터차원과 함께 지수적으로 커지기 때문에 성능은 역시 지수적으로 떨어진다는 것이다. 전통적인 트리구조는 데이터 차원이 높을 때 유사 오브젝트를 디스크에 랜덤적으로 분산 배치한다. 이에 따라 트리 인덱스구조는 데이터의 랜덤적 위치와 지수적으로 커지는 I/O로 인해 고차원 공간에서 유사한 검색을 실행하는데 비효율적이다. 본 연구에서는 클러스터링과 인덱싱을 혼합한 새로운 유사도 검색 방법을 제안한다. 클러스터링과 인덱싱은 다양하게 연구되었지만 이미지 유사도 검색의 개별적인 방안으로 제안되고 있다.

클러스터인덱싱 스키마는 특징공간을 작은 셀 형태의 하위영역으로 나눈다. 나뉜 셀은 유사한 레코드를 클러스터링하고 클러스터를 인덱싱하는데 이용한다. 클러스터링 단계에서 각 셀에 위치한 이미지 수를 기록하고 이미지가 포함된 주변 셀을 같은 클러스터에 통합한다. 디스크 그리드의 해상도를 변경함으로써 클러스터를 다단계로 구축할 수도 있다. 즉 그리드가 더 섬세할수록 셀 크기는 작아지고 클러스터의 결과도 더 상세해 진다. I/O의 효율성을 개선하기 위해 가장 촘촘하게 입도된 밀집 클러스터를 개별적인 연속적 파일로 저장하고 성글게 입도된 희소 클러스터를 가능한 한 많이 촘촘한 클러스터와 같은 실린더 그룹에 저장한다. 클러스터를 형성한 후 알고리즘은 클러스터를 인덱싱하는 매핑 테이블을 만든다. . 유사도 질의에 답하기 위해 먼저 질의 이미지의 특성 벡터를 해쉬하여 셀 ID를 구한 다음 셀 ID를 이용한 매핑 테이블로 질의 이미지가 위치한 클러스터를 탐색한다. 유사도 질의에 사용하는 I/O의 수는 보통 두가지이다. 한가지는 클러스터를 찾는 I/O이고 다른 한가지는 클러스터에서 읽는 연속적 파일의 I/O이다.

이와 같은 방법으로 클러스터인덱싱 스키마가 검색을 효율적으로 처리하여 찾고자 하는 이미지가 웹에 있으면 이미지를 포함하는 클러스터가 형성되어 이미지를 찾아낸다.

이 논문에서는 클러스터의 수와 크기에 영향을 미치는 계수를 제시하고 계수를 조율하는 절차를 설명하며 고차원 공간에서 빠른 검색을 지원하는 동시에 저장공간을 유지하는데 필요한 인덱스를 구축하는 방안을 제시한다. 데이터베이스 크기에 대해서는 선형적인 인덱스구조의 크기를 보존하는 힙 구조를 이용한다..

2. 관련연구

최근접 검색 기술이 고차원 공간에서 유사도 검색을 구현하는데 이용되어 왔다. 그러나 차원의 저주 때문에 최근 연구에서는 개략적으로 유사도 검색을 처리하는 방안을 제안하고 있다. White와 Jain [4]은 질의 대상이 위치하는 데이터의 버켓만을 돌려주는 방법을 제안했다. 이 방법은 빠르지만 재현율(recall)은 낮다. Kleinberg[13]은 $O((D\log^2 D)+(N+\log^3 N))$ 과 $O(N+D\log^3 N)$ 시간 안에 돌아가는 두 가지 근접 알고리즘을 제안했다. 이 두 가지 다른 접근 알고리즘은 차원을 줄이고 병렬 자원을 사용하여 차원의 저주를 처리하는 방법을 제안했다. 이 방법들은 아주 높지 않은 고차원 응용에 적용할 수 있지만 고차원의 이미지 검색 문제를 해결하기는 어렵다.

QBIC[14], Virage[15]등 주어진 이미지와 유사한 이미지를 찾는 전통적인 내용기반의 이미지 검색 방법이 있지만 이 시스템들은 전통적인 트리 구조를 이용하여 구현되었으므로 차원의 저주 문제를 야기시킬 수 있다. CLARANS[16], BIRCH[17], DBSCAN[18], CLIQUE[19]와 같은 클러스터링 연구가 데이터베이스 분야에서 이루어지고 있다. 이 기술들은 대형 데이터집합에서 클러스터를 확인하는 데 높은 성공율을 보인다. 그러나 이 방법들은 데이터 검색과 추출을 효과적으로 처리하지 못한다. 본 논문의 클러스터인덱싱 접근은 디스크에 유사이미지를 클러스터링 함으로써 차원의 저주를 피한다 또한 클러스터인덱싱 접근이 유사도 검색을 대략적으로 수행하지만 데이터 클러스터율 통해 재현율을 높인다.

3. 클러스터링

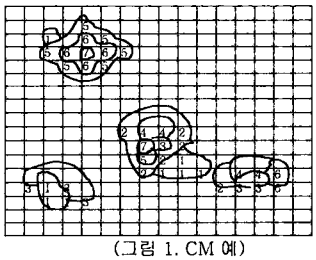
이 장에서는 클러스터링과 인덱싱 스키마를 설명한다. 스키마의 주요 아이디어는 유사 오브젝트를 추출하기 위한 디스크 지연도(latency)

를 최소화하도록 디스크에 유사 데이타끼리 클러스터하고 클러스터를 인덱스 하는데 해싱을 이용하여 클러스터 탐색 비용을 최소화한다. 접근방법은 세가지 단계로 구성된다.

1. i 번째 차원을 2^k 개 영역으로 나누고 최소 비트수를 사용하여 특성벡터가 속한 셀을 인코딩한다.
2. 클러스터내로 셀을 통합한다. 셀은 각각 다른 모양의 클러스터를 구축하기 위한 최소 블록으로서 차원을 나눈 하위영역이 작을수록 셀은 더 작아진다. 클러스터는 연속적인 파일형태로 디스크에 저장된다.
3. 클러스터를 참조하는 인덱스 구조를 만든다. 셀은 최소 주소단위이다. 인코딩 스키마는 오브젝트를 셀 ID로 해쉬하고 I/O 1회에 속하는 클러스터를 추출한다

3.1 CM 클러스터링 방법

고차원 공간에서 클러스터링을 효율적으로 수행하기 위해 클러스터링 생성(CM) 알고리즘을 제안한다. 그림 1은 2D 상에서 같은 크기로 나눈 그리드에 분포한 포인트(오브젝트)를 클러스터링 한 결과를 보여준다.



(그림 1. CM 예)

CM 알고리즘은 다음 방법으로 이루어진다

1. CM은 먼저 각 셀의 높이(오브젝트 수)를 기록한다
2. CM은 포인트 집중도가 가장 높은 셀에서 시작한다. 이 셀은 최초 클러스터의 최고점이다. (그림 1의 예에서는 7로 표시된 셀에서 시작)
   CM은 한번에 한 단위씩 내려온다. 셀은 다음 세가지 조건 중의 하나에 속해 있는데셀은 어떤 클러스터에도 인접하지 않거나 오직 한 클러스터에만 인접하거나 한 개 이상의 클러스터에 인접해 있다.
3. CM이 취하는 방법은
   (a)만약 셀이 어떤 클러스터에도 인접하지 않으면 셀은 새로운 클러스터를 생성하는 중심이 된다
   (b)셀이 한 개 클러스터에만 인접하면 셀을 클러스터에 조인한다
   (c)셀이 한 개 이상의 클러스터에 인접하면 CM 알고리즘은 어떤 클러스터에 셀이 속하는지 또는 셀이 속한 클러스터끼리 통합되는 것을 결정하기 위해 셀 결정 알고리즘을 적용한다.

CM은 셀의 높이가 임계값이 되면 끝난다. 어떤 클러스터에도 속하지 않는 셀은 외곽 클러스터에 통합되고 한 개의 연속적 파일로 저장된다
클러스터의 입도(Granularity)는 네 가지 계수로 제어된다

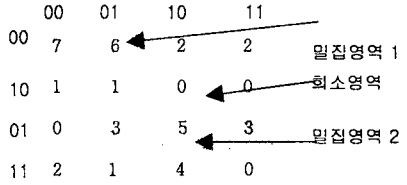
- D : 데이터차원 또는 오브젝트 특징
- b : 각 차원을 인코딩하는데 사용하는 비트의 수
- N : 오브젝트의 수
- θ : 수평계수

셀의 수는 D, θ 계수에 의해 결정되어 $2^{D\times b}$ 로 쓰인다. 그래서 셀의 평균 포인트 수는 $N / 2^{D\times b}$ 이다. 두 인자는 원하는 포인트 밀도를 결정한다. 낮은 포인트 밀도는 많은 수의 셀을 만들고 많은 수의 작은 클러스터를 생성하기 때문에 좋지 않다. 반면 높은 포인트 밀도는 숫자는 적지만 대형 클러스터를 만듦으로 역시 좋지 않다. b 값의 변화는 그리드 해상도와 클러스터 수,크기에 영향을 미친다. θ 값 역시 클러스터 수와 크기를 결정한다.

4. 유사도 검색

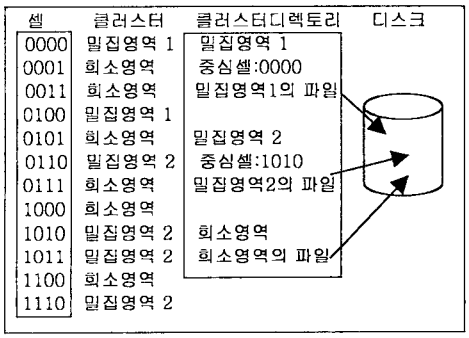
질의 오브젝트가 주어지면 먼저 매핑 테이블을 이용하여 오브젝트를 클러스터에 매핑한다. 그림 2는 2D 예를 보여준다. 각 셀은 x 축으로부터 시작해 y 축으로 가는 이진코드로 표현한다.

세가지 색상 특징치, 모멘트, 대비, 연관성, 분산, 혼잡도의 5가지 질감



(그림 2. 클러스터 예)

예를 들어 셀 0001은 x 축의 좌측 첫째 부분과 y 축의 위로부터 두번째 부분의 결합이다. 그림 3은 인덱스 구조 예를 보여준다. 그림의 좌측은 셀을 클러스터에 매핑하는 테이블이다. 클러스터를 찾은 다음 클러스터가 저장된 연속적 파일을 찾기 위해 클러스터 디렉토리를 이용한다. 클러스터를 형성한 후의 마지막 단계는 클러스터에 빨리 접근할 수 있는 인덱스를 구축하는 것이다. 오브젝트를 클러스터에 사상하기 위해 매핑 테이블을 생성하고 클러스터에 대한 정보를 기록하기 위해 클러스터 디렉토리를 구축한다.



(그림3. 인덱스 구조)

만약 셀이 한 개 이상의 클러스터에 인접해 있으면 셀 결정 알고리즘이 셀이 속할 클러스터를 결정한다. 셀이 인접한 클러스터의 최고점 약간 아래 높이라면 (최고점의 1/3) 이 클러스터를 통합한다.만약 다른 클러스터들의 근접 셀들이 같은 높이라면 오브젝트 수가 적은 클러스터들을 통합한다. 그렇지 않으면 랜덤하게 셀을 인접 클러스터에 연결한다.
그림 2에서 θ 값은 3이다. 3 이하의 셀은 희소 영역으로 클러스터링한다. 여기서 2개의 밀집영역과 1개의 희소 영역으로 3개 클러스터가 만들어진다. 밀집영역 1은 0000과 0100셀로 구성되고 밀집영역 2는 0110, 1010,1110, 1011셀로 구성된다. 클러스터 디렉토리는 3가지 클러스터 종류로 만들어진다. 밀집영역 1의 중심과 2의 중심은 0000과 10100이다. 클러스터를 저장하고 있는 파일의 이름 역시 디렉토리에 기록된다. 질의 이미지가 주어지면 먼저 x 와 y 속성을 4비트 코드로 양자화하고 그 다음 오브젝트가 위치한 클러스터를 매핑 테이블에서 찾는다. 만약 질의 대상이 밀집영역에 있으면 디스크에서 순차적으로 밀집영역에 속하는 디스크 블록을 읽고 거리함수를 이용하여 포인트를 순위매김한 후 그 결과를 보여준다. 예를 들어 질의 오브젝트가 0000 셀에 해쉬되면 밀집영역 1의 13개 오브젝트를 정렬순서로 찾아 보여준다 첫번째 시도에서 원하는 데이터를 찾지 못한 이용자를 위해 양방향 질의 기능을 제공한다. 찾는 오브젝트가 희소영역에 있으면 거리함수가 이를 찾는 동시에 근접 중심셀을 통해 밀집지역의 오브젝트를 찾아 이용자에게 제시하여 선택하도록 한다.

5. 평가

평가를 수행하기 위해 3,000개의 이미지 데이터베이스를 구축했다. 검색을 수행하기 위해 세 단계를 거쳐 이미지의 특징벡터를 추출한다. 이미지 특징벡터 추출과정은 내용기반 이미지 검색시스템에서 주로 적용하는 방법을 이용한다.[20,21,22]. 먼저 정규 크기와 형식으로 이미지를 바꾼다. 그리고 이미지에 Daubeches Wavelet transform을 적용하여 계수를 추출한다. 마지막으로 이미지의 특징벡터로 선택한 HSV 명도, 채도, 색도 특징치, 영역 크기 정보, 푸리에 기술자를 이용한 모양 및 위치정보 특징치를 특징공간에 웨이블릿 계수로 저장한다.

그런 다음 데이터 집합 을 적절하게 클러스터링하기 위해 클러스터링 알고리즘을 적용하고 데이터집합에서 패턴을 찾도록 b 와 θ 계수를 조정한다. 실험에서는 b 에 2를 θ 에 1을 적용하여 전체 3만개 오브젝트에서 16개 오브젝트를 갖는 25개 클러스터를 얻었다. 이용자가 첫번째 시도에서 원하는 결과를 찾을 수 있는 가능성을 p 라고 하고 마지막 시도한 클러스터에서 읽는 I/O시간을 T_{read} , 질의 정제 I/O시간을 T_{refine} 라고 한다. 검색 시간 T_{search} 은 다음과 같이 모델링한다.

$$T_{search} = (1-p)T_{refine}+T_{read} \quad (1)$$

T_{search} 를 최소화하기 위해서는 I/O시간을 최소화하고 p 를 최대화한다. p 값은 질의 오브젝트가 속한 클러스터를 정확하게 찾는 가능성이다. 검색비용은 한 개 연속적 파일 I/O 비용과 비슷하며 클러스터의 평균 크기에 달려 있다.

5.1 연속적 I/O 대 랜덤 I/O

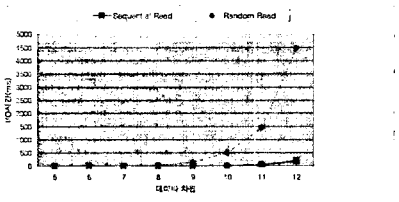
클러스터인덱싱 구조는 전통적 트리 유사구조보다 많은 데이터를 읽기 때문에 정확도는 떨어진다. 하지만 본 논문에서 제안한 바와 같이 I/O가 연속적으로 이루어지면 정확도는 높아진다. 또한 많은 데이터를 읽으면 재현율이 높아진다. 즉 검색할 때 I/O가 메모리 거리 계산보다 더 빈번하게 병목효과를 일으키기 때문에 연속적 방법으로 많은 데이터를 읽는 것이 랜덤적 방법보다 효과가 높다. 연속적 방법으로 디스크를 검색할 때 검색시간, 회전시간, S 량의 데이터를 전달하는 시간이 필요하다. 검색시간 함수 $\gamma(d)$ 는 주어진 검색거리 d 를 계산하는 시간이다. T_{rot} 은 평균 회전시간, S/TR 은 TR 전달율에 데이터 S 를 전달하는 시간이라 정의하면 연속적인 I/O 시간은

$$T_{read} = \gamma(d)+T_{rot}+S/TR \quad (2)$$

이며 랜덤적인 I/O 시간은 다음과 같다. S/B 의 B 는 블록의 크기이다.

$$T_{read} = S/B(\gamma(d)+T_{rot})+S/TR \quad (3)$$

그림 4에서는 데이터 차원 D 의 값이 다를 때 연속적 I/O 대 랜덤 I/O의 효과를 비교한다.



(그림 4. 연속적 I/O와 랜덤 I/O 시간)

그림의 x 축은 5부터 12까지의 데이터 차원 수이다. y 축은 3^D 블록 수 검색에 필요한 IO시간이다. 랜덤 I/O는 연속적 I/O보다 더 비용이 높다는 것이 명백하다. 예를 들어 1,000개의 랜덤 디스크 블록을 읽는시간이 28,000개 연속적 블록을 읽는 것과 동일하며 10,000개 랜덤 블록을 읽는 시간이 300,000개 연속적 블록을 읽는 것과 같다. 연속적 파일이 많은 데이터를 읽기 때문에 질의 오브젝트에 가깝지 않은 포인트를 포함함으로써 낮은 정확도(Precison)를 보이지만 이 논문의 제안 클러스터링 접근방법은 높은 차원의 멀티미디어 데이터를 검색하는데 전통적 인덱스 구조보다 더 빠른 I/O 시간을 기록한다.

6. 결론

이 논문에서는 고차원 데이터의 유사도 검색을 수행하는 새로운 방법으로 데이터를 클러스터링하고 이 클러스터들을 검색하는 방안을 제안했다. 이 접근방법은 디스크로부터 관련 정보를 연속적으로 추출하고 클러스터링함으로써 I/O 시간을 줄이고 연관된 클러스터를 연속적으로 검색해 감으로써 질의내용을 계속 바꿔가며 검색하는 이용자의 피드백을 포함하는 방안을 제공한다. 이 연구의 주된 연구결과를 요약하면 다음과 같다.

- 스토리지 클러스터에 데이터 클러스터를 사상하는 방안을 제안하고 이렇게 만들어진 클러스터에 접근하는 인덱스를 구축하였다
- 클러스터를 형성하는 CM알고리즘을 제안했다.
- 질의 이미지에 클러스터를 연결시키는 알고리즘을 이용하여 유사한 질의를 효과적으로 처리하는 방안을 제시했다.

참조

[1] P. Aigrain, H. Zang and D. Petkovic." content based Representaiton and Retrieval of Visual Media: A State-of-the-Art Review" Multimedia Tools and Applications, vol3, pp179-202,1996

[2] T.Hermes,C.Klauck,J.Krey and J, Zhang" Image Retrieval for information systems" in Proc. SPIE Vol24(20): Storage and Retrieval for Image and Video Databases" pp394-405,February 1995

[3] Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In Proceeding of International Conference Visual Information System, 1999.

[4] D. A. White and R. Jain. Algorithms and strategies for similarity retrieval. Stanford Technical Report, August, 1998.

[5] R. Weber, H. Schek and S. Blott. A quantitative anlaysis and performance study for silimarity-search methods in high-dimensional spaces. Proceedings of the 24th VLDB, pages 194-205,1998

[6] A. Guttman. R-tree; a dynamic index structure for spatial searching. Proceedings of ACM Sigmod. June 1984

[7] N. Beckmann. H. P. Kriegel, R. Schneider and B. Seeger. The r*-tree: an efficient and robust acces method for points and rectangles. Proceedings of ACM Sigmod May 1990.

[8] N. Katayama and S. Sotoh. The sr-tree: An index structure for high-dimensional nearest neighbor queries. Proceedings of ACM SIGMOD, May 1997.

[9] J. T. Robinson. The k-d-b-tree: A search structure for large multidimensional dynamic indexes. Proceedings of ACM SIGMOD, April, 1981.

[10] K. L. Lin, H. V. Jagadish and C. Faloutsos. The tv-tree: an index structure for high-dimensional data. VLDB Journal,3(4),1994

[11] S. Berchtold. The x-tree_ An index structure for high-demensional data. Proceedigns of the 22nd VLDB, August 1996.

[12] P. Ciaccia, M. Patella and P. Zezula, M-tree: An efficient access method for similarity search in metric spaces. Proceedings of the 23rd VLDB, August 1997.

[13] J. M. Kleinberg. Two algorithms for nearest-neighbor search in hgh dimensions. Proc. Of 29th ACM Symposium on Theory of Computing, 1997.

[14] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, and et al. Query by image and video content: the QBIC system. IEEE Computer, 28(9):23-32,1995.

[15] A. Hampapur, A. Gupta, B. Horowitz, C. Fuller, J. R. Bach M. Gorkani, and R. C. Jain. Virage Inc., Virage Video Engine. In Proc. SPIE Vol. 3022 : Storage and Retrieval for Image and Video Databases. pp.188-198, Feb. 1997.

[16] R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. Proceedings of the 20th VLDB, September, 1994.

[17] T. Zhang,R. Ramakrishnan and M. Liny. Birch: An efficient data clustering method for very large databases. Proceedings of Sigmod, June 1996.

[18] M. Ester, H. P. Kriegel, J. Sander and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the 2nd International Conference on Knowledge Discovery in Databases and Data Mining, August 1996.

[19] R.Agrawal, J.Gehrke, D.Gunupolus and P. Raghavan. Automatic subspace clustering of high diemensional data for data mining applications. Proceedings of ACM sigmod, June 1998

[20] G. Lu and A.Sajjanhar" Region based shape representation and similarity measure suitable for content based image retrieval" Multimedia systems, pp165-174,1999

[21] 고병철,이해성,변혜란 " 영역기반 영상 검색을 위한 FRIP 시스템" 정보과학회 논문지 : 컴퓨팅의 실제 제7권 제3호, pp 260-272, 20001

[22] R. Weber, H. Schek and S. Blott. A quantitative anlaysis and performance study for silimarity-search methods in high-dimensional spaces. Proceedings of the 24th VLDB, pages 194-205,1998