수평 분할 방법을 이용한 병렬 CBF(Cell-Based Filtering) 기법의 설계

김남기° 장재우
{ngkim° , jwchang}@dblab.chonbuk.ac.kr

Design of Parallel CBF(Cell-Based Filtering) Scheme using Horizontally-Partitioned Method

Nam-Gi Kim° Jae-Woo Chang
Dept. of Computer Engineering, Chonbuk National University

요 약

기존의 CBF 기법은 데이타의 차원이 증가함에 따라 검색 성능이 급격히 저하되는 'Dimensional Curse' 문제를 해결하기 위해 제안되었다. 그러나, 데이타의 양이 증가하고 차원이 증가할수록 검색 성능이 선형적인 감소를 보인다. 따라서, 본 논문에서는 CBF 기법의 성능 향상을 위해 멀티 디스크 환경을 기반으로 하는 병렬 CBF 기법을 제안한다. 제안하는 병렬 CBF 기법은 멀티 디스크 환경하에서 CBF가 지니는 특성을 이용하여 시그니쳐와 특징 벡터 데이타의 수평 분할 방법을 사용한다. 이를 통해, 제안하는 기법은 디스크 개수에 비례하여 선형적인 검색성능 향상을 가져온다.

1. 서 론

최근 멀티미디어 데이타베이스 응용에서 내용기반 검색에 대한 관심이 부각되고 있다. 멀티미디어 내용기반 검색은 데이타베이스에 저장된 대량의 멀티미디어 객체들 중에서 내용을 기반으로 사용자가 원하는 객체를 찾는 검색이다. 이는 멀티미디어 객체로부터 특징벡터를 추출하여 데이타베이스에 저장한 후, 사용자가 검색하고자 하는 멀티미디어 객체의 특징벡터와 가장 유사한 값을 찾는 방법이다. 그러나 기존의 색인 구조들은 추출된 특징 벡터의 차원이 증가함에 따라 검색 성능이 급격히 저하된다. 이러한 차원의 증가에 따라 발생하는 기존의 고차원 색인 기법의 비효율성을 극복하기 위해 CBF(Cell-Based Filtering)기법이 제안되었다[1].

CBF 기법에서 검색 성능은 시그니쳐 파일과 특징 벡터 파일을 탐색하기 위한 디스크 접근 횟수에 의존한다. 한편, 데이타의 양이 증가하고, 차원이 증가할수록 전체 시그니쳐 파일의 크기와 특징 벡터 파일의 크기는 선형적으로 증가하므로 검색 성능을 저하시키는 결과를 가져온다.

따라서, 본 논문에서는 CBF 기법의 성능향상을 위해 멀티 디스크 환경을 기반으로, 데이타를 각 디스크에 분산시켜 저장 및 검색하는 병렬 CBF 기법을 제안한다. 아울러, 시그니쳐 파일은 전체 탐색되어야 하고, 특징 벡터 파일은 시그니쳐 파일의 탐색을 통해 얻어진 후보 특징 벡터만 부분적으로 탐색되어지는 특성을 고려하여 시그니쳐와 특징 벡터 데이타를 수평 분할 방법을 이용하여 분산시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 CBF 기법을 살펴보고, 3장에서는 수평 분할 방법을 이용한 병렬 CBF 기법을 제안한다. 4장에서는 본 논문에서 제안한 방법을 분석적으로 성능 평가를 수행하며, 마지막으로 5장에서는 결론 및 향후 연구 방향을 제시한다.

2. 셀-기반 필터링(CBF) 기법

기존 셀 기반 필터링 기법(CBF)은 고차원 색인 기법들이 차원이 증가함에 따라 성능이 급격히 저하되는 Dimensional Curse 문제를 해결하고자 제안되었다. 셀 기반 필터링 (CBF : Cell-Based Filtering)기법은 객체들의 특징 벡터들에 대한 순차 탐색을 수행하기 전에, 각각의 특징 벡터에 대한 시그니쳐를 탐색하여 필터링 함으로써 탐색 성능을 향상시킨다. 또한, 특징 벡터에 대한 시그니쳐 뿐만 아니라 셀 중심에서 객체까지의 거리 정보를 이용함으로써, 필터링 효과를 증대시킨다. 시그니쳐는 데이타 각각의 차원별 특징 벡터의 요약 정보로써벡터 데이타 공간을 균등한 셀로 나누어 만들고 셀의 정보를 포함하는 셀의 대표값이 된다[2].그림1은 N개의 데이타가 저장될 때의 셀 기반 필터링 기법의 전체 구조이다.

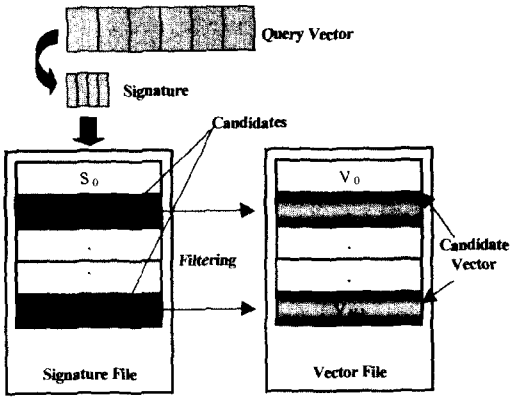

그림 1. 셀-기반 필터링 기법의 전체 구조

질의 벡터가 주어지면, 시그니쳐 생성 알고리즘을 이용하여 질의 시그니쳐로 변환한 후, 이를 이용하여 시그니쳐 파일에 저장된 각각의 시그니쳐들을 순차적으로 탐색하여 필터링을 수행한다. 아울러, 각 시그니쳐 정보와 함께 저장된 거리 정보를

사용함으로써 보다 효율적인 필터링을 수행한다. 이러한 필터링을 통해 선택된 각 후보 시그니쳐의 실제 특징 벡터들을 데이타 파일에서 탐색함으로써 최종적으로 주어진 질의를 만족하는 특징 벡터를 검색한다. 시그니쳐와 거리 정보에 의한 필터링 효과가 클수록 검색 성능은 우수하게 된다.

3. 수평 분할 방법을 이용한 병렬 CBF 기법

CBF 기법에서 데이타 객체로부터 추출된 특징 벡터와, 특징 벡터로부터 생성된 시그니쳐 데이타의 크기는 전체 데이타 객체에 비해 훨씬 적다. 그러나 데이타의 양이 증가하고, 데이타 객체의 차원이 증가할수록 시그니쳐 데이타와 특징 벡터 데이타의 양은 선형적으로 증가하므로 탐색 성능이 선형적으로 감소한다. 이를 위해, 본 논문에서는 기존 CBF기법에 병렬 데이타베이스 개념을 도입하여 멀티 디스크를 사용하는 병렬 CBF기법을 제안한다.

기존의 CBF 기법의 색인 구조는 데이타 객체로부터 추출된 시그니쳐와 특징 벡터를 저장하고 있는 시그니쳐 파일과 특징 벡터 파일을 하나의 물리적 디스크에 저장한 후에 검색한다. 그러나, 병렬 CBF 기법은 물리적으로 분리된 여러 디스크에 시그니쳐 데이타와 특징 벡터 데이타를 분산시켜 저장하기 때문에 CBF 기법을 병렬 CBF 기법으로 확장하기 위해서는 데이타의 분산 문제를 고려해야 한다. 기존의 병렬 데이타 분산 방법에는 수평 분할 방법과 수직 분할 방법이 있다. 수평 분할 방법은 데이타를 디스크 수에 따라 일정 비율로 나눈 후, 각 디스크에 분산시켜 저장하는 방법이고[3], 수직 분할 방법은 각각의 데이타즉, 시그니쳐와 벡터를 디스크 수만큼의 일정 크기의 프레임으로 나눈 후 각 디스크에 분산시켜 저장하는 방법이다[4]. 한편 두 가지 데이타 분할 방법을 병렬 CBF 기법에 적용하기 위해서는 먼저 CBF의 특성을 고찰하는 것이 필요하다.

특성 1. 주어진 질의에 대한 검색을 위해서는, CBF의 시그니쳐 파일의 전체 데이타가 순차적으로 탐색되어야 한다.

CBF에 질의가 주어지면, CBF는 시그니쳐 파일 전체를 탐색하므로, 시그니쳐 파일을 수직 분할 방법을 이용하여 분산, 저장할 경우, 시그니쳐 개수 만큼의 디스크 I/O 횟수가 필요하고, 아울러, 분할된 데이타를 병합하는 처리 시간이 추가로 필요하다. 그러나 수평 분할 방법을 사용할 경우, D개의 물리적 디스크를 사용하고, 전체 시그니쳐 개수가 N이라 하며, 데이타가 고르게 분포되어 있다고 가정하면, 디스크 I/O 횟수는 각 디스크당 약 N/D 번으로 줄어들고, 수직 분할 방법에서의 데이타 병합을 위한 추가비용은 필요하지 않다.

특성 2. 주어진 질의에 대한 검색을 위해서는, 시그니쳐 파일의 탐색을 통해 얻어진 후보 특징 벡터에 대해서만 CBF의 특징벡터 파일을 부분 탐색한다.

CBF에서 특징 벡터 데이타는 시그니쳐 검색 후 추출된 후보셀에 해당하는 특징 벡터, 즉, 특징 벡터 전체 데이타중 일부분만을 탐색한다. 수직 분할 방법으로 데이타를 디스크에 분산시켜 저장할 경우, 검색되어야 할 벡터 데이타의 수가 N이라면 N번의 디스크 I/O 횟수가 발생하게 되고, 또한 데이타 병합 처리시간이 필요하다. 그러나, 수평 분할 방법으로 데이타를 분산, 저장할 경우, 후보 특징 벡터가 임의의 위치에 저장되어 있다고 가정하면, 약 N번의 디스크 I/O 횟수가 필요하며, 데이타 병합에 드는 처리비용은 존재하지 않는다.

위의 두 특성을 통해서, 본 논문에서 제시하는 병렬 CBF 기법은 시그니쳐 파일과 특징 벡터 파일을 모두 수평 분할 방법을 사용해서 분산, 저장한다. 그리고 데이타를 각 디스크에 분산시켜 저장할 때 데이타를 디스크수에 따라 모듈러(Modelar) 함수를 이용하여 저장한다.따라서, 같은 ID의 시그니쳐와 특징 벡터 데이타는 한 디스크에 함께 저장된다. 본 논문에서 제안하는 병렬 CBF 기법의 데이타 분산, 저장 구조는 그림 2와 같다. 여기서, N은 전체 데이타 수이고, D는 디스크 개수이고, n은 평균 디스크당 데이터수 $\lceil N/D \rceil$이다.

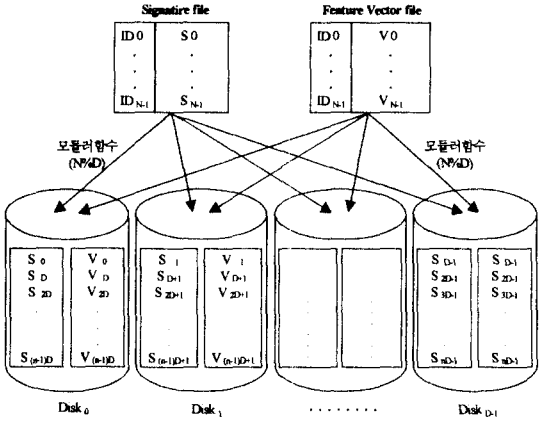

그림 2. 수평 분할 방법을 이용한 병렬 CBF 기법의 데이타 분할, 저장 구조

4. 분석적 성능 평가

본 논문에서 제안한 병렬 CBF기법에 대해 분석적인 성능평가를 수행한다. 디스크 I/O 측정을 위해 랜덤 데이터와, Hadamard 알고리즘[5]을 이용하여 만들어진 Hadamard 리얼 데이타 각각 10, 20, 50, 80 차원의 각 20만건의 데이타로 실험한다. 본 논문에서 제시한 기법을 이용하여 K-최근접 질의에 대한 성능 평가를 수행한다. 실험에 쓰인 인자(Parameter)는 표 1과 같다.

표 1. 실험 인자

인자	설 명
L_S	The length of a signature data (bits)
N	The number of Vector data
b	The number of bits for a signature per dimension
d	The number of dimensions
P	Page size (64 kbits)
D	The number of disks (4, 8, 10, 16)
r	The length of radius an object in a cell and its central point
C_i	The number of candidate cells per disk i
R_S	The number of disk I/O for accessing signature file
R_V	The number of disk I/O for candidate feature vectors
R	The number of total disk I/O

식(1)은 주어진 질의에 대하여, 검색을 위한 디스크 I/O 횟수를 계산하기 위한 식이다. 전체 디스크 I/O 횟수는 R로 표현한

다. Rs는 시그니처 파일을 탐색하기 위한 디스크 I/O 횟수이고, Rv는 특징 벡터 파일을 탐색하기 위한 I/O 횟수이다.

$$R = R_s + R_v \quad \cdots\cdots\cdots\cdots\cdots\cdots \text{식(1)}$$

$$R_s = \left(\frac{L_s \times N}{P}\right) \Big/ D \qquad L_s = (b \times d \times N) + r$$

$$R_v \approx \max(C_i) \qquad (\forall i, 0 \le i < D)$$

그림 3은 랜덤 데이타에 대한 실험 결과이다. 데이타가 80차원이고 디스크 수가 4개 일때, 디스크 I/O는 550번이 발생하고, 8개일때는 278번, 16개 일때는 142번이 발생함을 보인다. 따라서, 물리적 디스크 수가 증가할수록 시그니처 파일과 특징 벡터 파일을 탐색하는 디스크 I/O 횟수는 선형적으로 감소함을 알 수 있다.

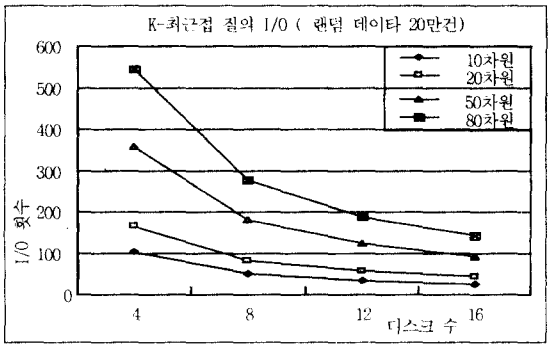

그림 3. 랜덤 데이타에 대한 K-최근접 질의 디스크 I/O 횟수

그림 4는 리얼 데이타에 대한 실험 결과이다. 데이타가 80차원이면, 디스크 수가 4개 일 때, 디스크 I/O는 587번이 발생하고, 8개일때는 294번,16개 일때는 148번이 발생함을 보인다. 랜덤 데이타에 비해서 디스크 I/O 횟수는 다소 증가하나 이는 시그니처를 검색한 후의 후보셀의 수가 더 많기 때문이며, 물리적 디스크수가 증가할수록 마찬가지로 디스크 I/O 횟수가 선형적으로 감소한다.

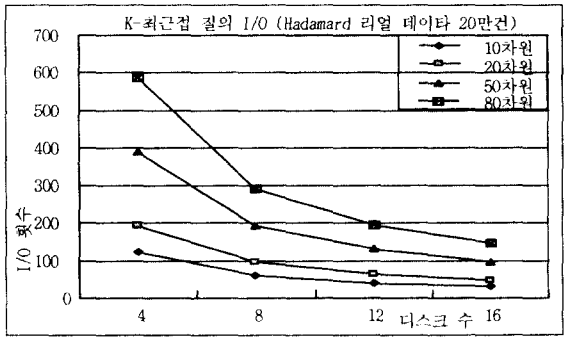

그림 4. Hadamard 리얼 데이타에 대한 K-최근접 질의 디스크 I/O 횟수

데이타 특성에 따른 각각의 물리적 디스크에 저장된 데이타의 분포 정도를 위해 편중도(S_D)를 계산하며, 다음의 식(2)로 계산된다.

$$S_D = \max(C_i) / avg(C_i) \quad (\forall i, 0 \le i < D) \quad \cdots\cdots \text{식(2)}$$

편중도 측정은 각 10, 20, 50, 80차원별로 20만건의 랜덤 데이타와 각 15, 20차원별 20만건의 애니메이션(Animation) 데이타를 사용한다. 표 2는 데이타간의 특성 차이에 의해 랜덤 데이타의 편중도가 애니메이션 데이타의 편중도보다 근소한 차이로 우위를 보이지만, 두 종류의 데이타 모두 거의 고르게 분산되어 있음을 보인다. 따라서, 수평 분할 방법을 사용하는 것이 타당함을 알 수 있다.

표 2. 데이타 종류에 따른 데이타 편중도

랜덤 데이타 편중도

	디스크수 4	디스크수 8	디스크수 12	디스크수 16
10 차원	1.0090	1.0441	1.0200	1.0639
20 차원	1.0074	1.0171	1.0414	1.0622
50 차원	1.0324	1.0752	1.0530	1.0836
80 차원	1.0257	1.0388	1.0493	1.0466

애니메이션 리얼 데이타 편중도

	디스크수 4	디스크수 8	디스크수 12	디스크수 16
15 차원	1.0404	1.0760	1.0829	1.1056
20 차원	1.0220	1.0536	1.0597	1.1483

5. 결론 및 향후 연구

기존 CBF (Cell-Based Filtering)기법은 기존의 트리에 기반한 색인 구조의 Dimensional Curse 문제를 해결하기 위해 벡터 공간을 일정한 크기의 셀로 분할하여 시그니처 정보를 유지하고 이를 이용하여 검색을 수행함으로써 검색 성능을 향상시키는 방법이다. 본 논문에서는 멀티 디스크 시스템 환경에서 수평 분할 방법을 이용한 병렬CBF 기법을 제안하고, 이를 통해 검색 성능을 향상시켰다.

향후 연구로는 구현을 통한 실험을 수행하는 것이며, 아울러, 제안하는 병렬 CBF 기법이 고차원 색인 구조에 적합함을 보이기 위해 클러스터 시스템에 적용하는 것이다.

참고문헌

[1] 장재우,한성근,김현진, " 셀 기반 필터링 방법을 이용한 고차원 색인기법" ,정보과학회 논문지:데이타베이스 제28권2호,pp.204-216,2001.

[2] Roger Weber, Stephen Blott, "An Approximation-Based Data Structure for Similarity Search", Technical report Nr. 24,ESPRIT project HERMES(NO.8141),October 1997.

[3] Jeong-gi Kim, Jae-Woo Chang, "Horizontally divided signature files on a parallel machine architecture", Journal of system Architecture Vol. 44, pp.723-735,1998.

[4] 김정기,유경민,장재우," 수직 분할 병렬 요약화일 기법의 설계 및 성능평가" ,정보과학회논문지(B) 제26권1호,pp.66-79, 1999.

[5] K.-I.Lin, H. V. Jagadish, and C. Faloutsos, "The TV-tree: An Index Structure for High-Dimensional Data", VLDB Journal, Vol.3, NO.4,pp.517-542, 1994.