수평 분할 방법을 이용한 병렬 CBF 기법의 성능평가

박승봉[0] 장재우

전북대학교 컴퓨터공학과

(sbpark[0], jwchang)@dblab.chonbuk.ac.kr

Performance Analysis of a Parallel CBF Scheme using Horizontally-Partitioned Method

Seong-Bong Park[0] Jae-Woo Chang

Dept. of Computer Engineering, Chonbuk National University

요 약

기존의 색인 기법들은 차원의 수가 증가할수록 검색 성능이 급격히 저하되는 문제를 지니고 있다. 이 문제를 극복하기 위하여 CBF 기법이 제안되었다. 그러나 CBF 기법은 데이터 양이 증가함에 따라 검색 성능이 선형적으로 감소하는 문제가 존재한다. 이를 해결하기 위해 다수의 디스크를 수평 분할 방법을 이용하여 디클러스터링(declustering)을 하는 병렬 CBF 기법이 제안되었다. 본 논문에서는 수평 분할 방법을 이용한 병렬 CBF (Parallel CBF) 기법을 삽입시간, 범위 질의 검색시간, k-최근접 질의 검색시간, 데이터의 편중도 측면에서 성능 평가를 수행한다. 아울러, 병렬 CBF 기법을 기존 CBF 기법과 성능 비교를 수행하며, 이를 통해 병렬 CBF 기법이 기존 CBF 기법보다 우수한 검색 성능을 나타냄을 보인다.

1. 서 론

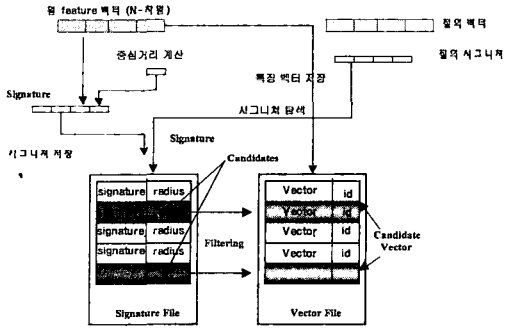
데이터웨어하우징에 사용되는 데이터의 n-차원 속성이나 멀티미디어 데이터베이스에서 사용하는 멀티미디어 객체의 n-차원 특징 벡터들은 모두 고차원 데이터이며, 이러한 고차원 데이터를 효율적으로 검색하기 위해서는 고차원 색인 기법이 요구된다. 제안된 대부분의 색인 기법들이 차원의 수가 증가할수록 검색 성능이 급격히 저하되는 '차원 저주(dimensional curse)' 문제를 가지고 있다[1]. 이러한 문제를 해결하기 위해 VA-File과 CBF 기법이 제안되었다. VA-File 기법은 근사 정보를 사용하여 필터링을 수행하며[2], CBF 기법은 VA-File 기법의 근사정보 외에 셀에 적합한 최대거리 및 최소거리를 새롭게 정의하여 시그니쳐를 구성함으로써 보다 좋은 검색 성능을 제공한다[3].

그러나, CBF기법은 데이터의 양이 증가할수록 선형적으로 검색성능이 감소하는 문제가 있어, 이를 해결하기 위해 다수의 디스크를 수평 분할 방법을 이용하여 디클러스터링(declustering)을 하는 병렬 CBF 기법이 제안되었다 [4]. 본 논문에서는 수평 분할 방법을 이용한 병렬 CBF (Parallel CBF) 기법을 삽입시간, 범위 질의 검색시간, k-최근접 질의 검색시간, 데이터의 편중도 측면에서 성능 평가를 수행한다. 아울러, 기존 CBF 기법과 병렬 CBF 기법와의 성능 비교를 수행한다.

본 논문의 구성은 다음과 같다. 2장에서는 병렬 CBF 기법에 대하여 기술하고, 3장에서는 병렬 CBF 기법의 실험을 통한 성능 평가 결과를 기술하고, 마지막으로 4장에서는 결론 및 향후 연구 방향을 기술한다.

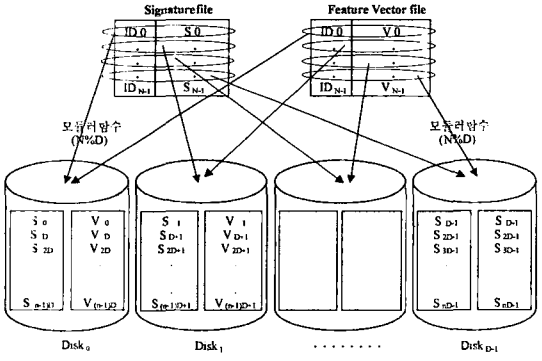
2. 수평 분할 방법을 이용한 병렬 CBF 기법

CBF 기법은 객체들의 특징 벡터들에 대한 순차 탐색을 수행하기 전에, 각각의 특징 벡터에 대한 시그니쳐를 탐색하여 필터링함으로써 탐색 성능을 향상시킨다. 이러한 특징벡터를 시그니쳐로 변환하는데 있어서[5], 셀(Cell)로 표현된 벡터의 정보와 셀 중심간의 거리 정보를 이용하여 필터링 효과를 증대시킴으로써 검색성능을 향상시킨다. [그림 1]은 CBF 기법의 구조를 나타낸다.



[그림 1] CBF 기법의 구조

병렬 CBF 기법은 기존 CBF 기법의 성능향상을 위해 멀티 디스크 환경을 기반으로, 데이터를 각 디스크에 분산시켜 저장 및 검색한다.



[그림 2] 수평분할 방법을 이용한 병렬 CBF 구조

아울러, 시그니쳐 파일은 전체 탐색되어야 하고, 특징 벡터 파일은 시그니쳐 파일의 탐색을 통해 얻어진 후보 특징 벡터만 부분적으로 탐색되어지는 특성을 고려하여 시그니쳐와 특징 벡터 데이터를 수평 분할 방법을 이용하여 분산시켜 저장한다[6]. [그림 2]는 수평 분할 방법을 이용한 병렬 CBF 기법을 나타내며, 물리적으로 분리된 다수의 디스크에 시그니쳐 데이터와 특징 벡터 데이터를 병렬 분산시켜 저장하고 있다.

3. 성능 평가

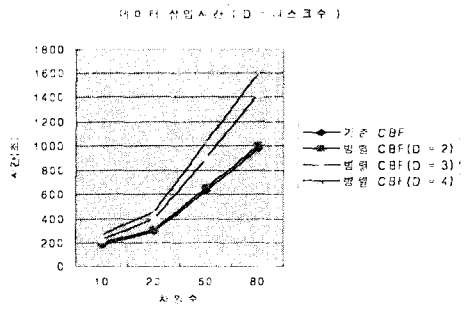
본 절에서는 병렬 CBF 기법을 구현하여 성능 평가를 수행한다. 구현된 시스템 환경은 [표 1]과 같다. 실험 데이터는 200만건의 Synthetic 랜덤 데이터 셋 각각 10차원, 20차원, 50차원, 80차원을 사용하였다. 성능 비교 대상은 기존 CBF 기법과 병렬 CBF 기법이다. 또한 성능 평가는 데이터 삽입, 범위 질의 검색, k-최근접 질의 검색에 필요한 시간을 측정하여 수행한다.

[표 1] 실험적 성능평가 시스템 환경

OS	Windows 2000 Server
CPU	1 GHZ Dual
disk	SCSI HDD(18GB * 4)
Main Memory	512 MB
Compiler	Visual C++ 6.0

3.1 삽입 시간

삽입 시간은 200만건의 Synthetic 랜덤 데이터 각각 10차원, 20차원, 50차원, 80차원을 병렬 CBF 기법을 사용하여 삽입하는데 필요한 시간이다([그림 3]).



[그림 3] 데이터 삽입 시간

10차원의 경우 기존 CBF 기법은 삽입시간으로 140초가 소요되며, 병렬 CBF의 경우 디스크가 2개, 3개, 4개일 때, 각각 153초, 164초, 175초로 디스크의 개수가 증가할수록 데이터 삽입 시간도 점차적으로 증가함을 알 수 있다. 또한 80차원의 경우 기존 CBF 기법이 912초 정도의 시간이 소요되며, 병렬 CBF 기법은 디스크의 개수에 따라 각각 940초, 1080초, 1140초 정도로 역시 디스크의 개수가 증가할수록 데이터 삽입 시간이 증가하고 있다.

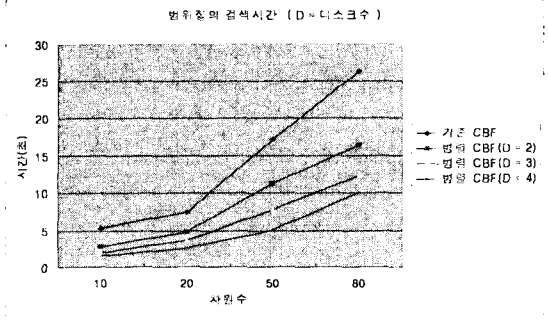
기존 CBF 기법에 비해 병렬 CBF 기법이 삽입시간 측면에서의 성능이 더 떨어지는 것을 알 수 있으며, 그 이유로는 병렬 CBF가 각각의 디스크당 스레드를 생성하는 시간과, 디스크를 번갈아 가며 데이터를 자례로 삽입함으로써 발생하는 스레드간 문맥(context) 교환 시간, 데이터를 읽기 위해 주어진 데이터 파일을 중복해서 접근함으로써 생기는 부가적인 시간, 데이터 삽입시 버퍼를 사용하지 않아 데이터를 삽입할 때 각각의 디스크에 병렬적으로 접근하는 대신 순차적으로 접근함으로써 생기는 부가적인 시간 때문이다.

3.2 범위 질의 검색 시간

범위 질의는 주어진 질의 포인트로부터 일정한 거리 영역 안에 포함된 객체를 찾는 검색이다. 병렬 CBF 기법의 성능 평가를 위하여 물리적으로 독립된 2개, 3개, 4개의 디스크를 각각 사용한다. 범위 값은 전체 데이터 중에서 0.1%의 데이터를 검색할 수 있는 값을 범위 값으로 사용한다. [그림 4]는 범위 질의 검색 시간을 기존 CBF 기법과 병렬 CBF 기법에서 디스크가 4개인 경우를 각 차원별로 비교한 것이다. 여기서 성능향상 지수는 기존 CBF 기법에 비해 병렬 CBF의 성능 향상치를 수치적으로 나타낸 것으로써 다음 식(1)에 의해 구할 수 있다.

$$\frac{1}{PT/CT} \times 100 \quad \text{--식(1)}$$

여기서 PT는 병렬 CBF 기법의 성능 측정치(시간)이며, CT는 기존 CBF 기법의 성능 측정치이다. 데이터의 차원수가 10차원일 때 188%정도의 성능향상을 보이며 이는 이론적 최대치 200% 에 거의 도달하는 수치이다. 한편 디스크가 D=3,4 일 경우, 각각 251%, 311%의 성능향상을 보이고 있다. 디스크의 개수가 증가할수록 이론적 최대치 300%, 400%에는 미치지 못하지만 디스크의 개수에 비례하여 검색 성능을 크게 향상되는 것을 알 수 있다.



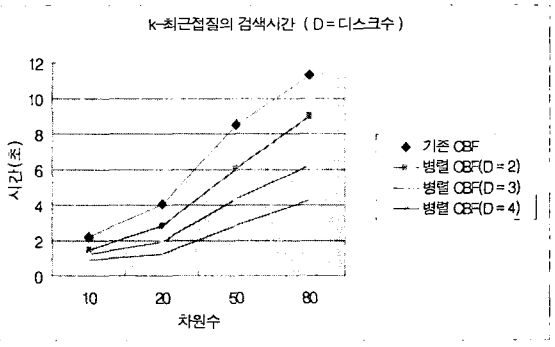
[그림 4] 범위 질의 검색 시간

3.3 k-최근접 질의 검색 시간

k-최근접 질의는 주어진 질의 벡터로부터 가장 가까운 k개의 객체를 찾는 검색이다. 본 성능평가에는 k값을 100으로 정하여 검색 시간을 측정한다. [그림 5]는 기존 CBF 기법과 병렬 CBF 기법의 디스크가 2개, 3개, 4개의 경우에 대해 각각 10차원, 20차원, 50차원, 80차원에서 측정된 k-최근접 질의 검색 시간을 나타낸다.

[그림 5]에서 알 수 있듯이 디스크의 개수가 증가할수록 k-최근접 질의 검색 성능도 향상됨을 알 수 있다. 예를 들어 차원의 수가 80차원인 경우 기존 CBF 기법이 11.32초 정도의 시간이 소요되며, 병렬 CBF 기법은 디스크 수가 2, 3, 4 에 따라, 9.01초, 6.23초, 4.26초가 소요됨을 볼 수 있으며, 범위질의의 경우와 비교해 볼 때, 범위질의 만큼의 성능향상을 가져오지 못하고 있

다. 이러한 검색 결과를 보이는 이유는 첫째, k-최근접 질의의 특성상 CBF 기법을 통해 실제 데이터 파일인 벡터 파일을 검색하게 되는 횟수가 3차에 걸친 필터링을 통해 k값에 근접하므로 디스크의 수가 늘어나도 최종적으로 벡터파일을 검색하는 횟수는 상대적으로 작은 값이므로 많은 차이를 보이지 않는다. 둘째, k-최근접 질의의 각 필터링 단계마다 만들어지는 후보리스트 및 결과리스트를 각 스레드가 공유하기 때문에 생기는 부가적인 시간이 범위 질의 만큼의 성능향상을 가져오지 못하기 때문이다.



[그림 5] k-최근접 질의 검색 시간

3.4 데이터의 편중도

데이터 디클러스터링을 이용한 병렬 CBF 기법에서는 디스크의 불필요한 접근이나 데이터의 재분산이 일어나지 않도록 주어진 데이터를 다수의 디스크에 얼마나 균등하게 분산시키느냐가 주요한 관건이 된다. 즉, 주어진 질의에 대한 병렬 질의 처리시 실행 밀집 현상(execution skew)이 일어나지 않도록 하는 것이 주요한 목표가 된다. 따라서 본 논문에서 제안하는 수평 분할 디클러스터링을 이용한 병렬 CBF 기법은 질의 처리를 수행할 때, 실행 밀집 현상(execution skew)이 얼마나 일어나는 지를 측정하기 위해서 데이터 편중도를 계산한다. 즉, 편중도(SD)는 다음과 같이 식(2)를 통해 계산되며, 디스크 i 당 수행되어야 하는 I/O 횟수/시간(C_i)의 최대치를 평균 값으로 나눈 값이다.

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

[표 2]는 데이터의 편중도를 나타낸다. 표에 나타난 바와 같이 10차원이고, 디스크가 2, 3, 4개일 때 데이터 편중도는 각각 1.15, 1.30, 1.31을 나타내어 데이터의 분포가 1에 가까워 데이터가 각 디스크에 고르게 분포되어 있음을 알 수 있다. 차원의 수가 80차원일 때는 디스크가 2, 3, 4에 따라 각각 1.23, 1.52, 1.70을 나타내고 있다. 10차원에 비해 데이터의 분포가 고르지 못함을 나타내는데, 차원의 수가 8배 증가함으로써 삽입하는 데이터의 크기가 증가하기 때문에 나타나는 현상이다. 하지만 80차원일 때에도 데이터의 분포는 비교적 고르게 분산됨을 알 수 있다.

[표 2] 데이터 편중도

차원수 \ 디스크수	2	3	4
10	1.1504	1.3022	1.3063
20	1.1571	1.3041	1.4062
50	1.3152	1.5453	1.5183
80	1.2318	1.5193	1.7046

4. 결론 및 향후 연구

본 논문에서는 수평 분할 방법을 이용한 병렬 CBF 기법을 Windows 2000 환경에서 구현하고, 삽입시간, 범위 질의 검색시간, k-최근접 질의 검색시간, 데이터의 편중도 측면에서 성능 평가를 수행하였다. 아울러, 병렬 CBF 기법을 기존 CBF 기법과 성능 비교를 수행하였다. 성능 평가 결과 이론적으로 가질 수 있는 성능향상 최대치에는 다소 못 미치지만, 병렬 CBF 기법이 범위 및 k-최근접 질의에 대해 기존 CBF 기법과 비교하여 물리적으로 독립된 디스크 개수가 비례하여 거의 선형적인 성능 향상이 이루어짐을 보였다.

향후 연구로는 병렬 CBF 기법을 클러스터 시스템 환경에 적용하여 추가적인 성능평가를 수행하는 것이다.

참고문헌

[1] S. Berchtold, C. Bohm, D. Keim, and H.-P. Kriegel, "A Cost Model for Nearest Neighbor Search in High-Dimensional Data Space", ACM PODS Symposium on Principles of Databases Systems, Tucson, Arizona, 1997.

[2] R. Weber, H.-J. Schek, S. Blott, "A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces," Proceedings of International Conference on Very Large Data Bases, pp.24-27, 1998.

[3] S.-G. Han and J.-W. Chang. "A New High-Dimensional Index Structure Using a Cel -based Filtering Technique", In Lecture Notes in Computer Science 1884 (Current Issues in Databases and Information Systems), Springer, pp. 79-92, 2000.

[4] 김남기, 장재우 " 수평 분할 방법을 이용한 병렬 CBF(Cell-Based Filtering) 기법의 설계", 한국정보과학회 추계 학술발표 논문집(I), 제 28권 2호, pp. 70-72, 2001.

[5] C. Faloutsos, "Design of a Signature File Method that Accounts for Non-Uniform Occurrence and Query Frequencies", ACM SIGMOD, 165-170, 1985.

[6] J.-K. Kim and J.-W. Chang, "A Horizontally Divided Signature File on Parallel Machine Architecture," Journal of System Architecture, Vol. 44, No. 9-10, pp.723-735, Jun. 1998.