SAN 환경에 적합한 고차원 색인 구조 설계

박춘서°, 신재룡, 송석일, 복경수, 유재수, 신범주*
충 북 대 학 교 정 보 통 신 공 학 과
*한국전자통신연구원, 인터넷 서비스 연구부
{parkcs, jrshin, prince, ksbok, yjs}@pretty.chungbuk.ac.kr
*bjshin@etri.re.kr

Design of High-dimensional Index Structures for SAN Environment

Choon Seo Park°, Jae Ryong Shin, Seok-Il Song, Kyoung Soo Bok,
Jae Soo Yoo, Bum Joo Shin
Department of Computer and Communication Engineering,
Chungbuk National University
*Internet Service Lab., Electronics and Telecommunication Research Institute

요 약

SAN(Storage Area Network)이 최근 대용량 데이터를 효율적으로 관리하기 위한 차세대 저장 장치로 각광받고 있다. 이 SAN에는 이미지, 동영상, 지도, 캐드 데이터와 같은 대용량의 고차원 특징을 갖는 데이터들이 저장되어 관리 될 것이다. 따라서 SAN 환경에서 이들을 보다 빠르고 정확하게 검색할 수 있는 효율적인 고차원 색인구조가 필요하다. SAN 환경은 저장 장치를 공유하는 형태의 병렬 환경이라 볼 수 있다. 이 논문에서는 SAN의 병렬성을 충분히 이용해서 고차원 데이터를 색인 할 수 있는 방법을 제안한다. 제안하는 고차원 색인 구조는 하나의 노드를 여러 디스크에 분산시켜 팬-아웃을 증가시키고 트리의 높이를 줄임으로서 검색 성능을 향상시킨다. 또한 범위 질의와 K-최근접 질의 수행시 병렬성을 최대화하는 방법을 제안한다.

1. 서론

최근 지리 정보 시스템, 내용기반 이미지 검색 시스템, 멀티미디어 데이터베이스 등에서 고차원 색인구조는 매우 중요한 비중을 차지한다. 현재까지 제안된 고차원 색인 구조들은 TV-트리, X-트리, SS-트리, SR-트리, CIR-트리와 같은 데이터 분할을 사용하는 색인 구조와 KDB-트리, hB-트리, LSDh-트리, BANG 파일, GRID 파일과 같이 공간분할을 사용하는 색인 구조들이 있다. 또한, 이들의 혼합 형태의 색인구조로 Hybrid-트리가 존재하며 이 외에도 LS(Locality Sensitive)해싱 기법을 사용하는 색인구조와 VA-파일와 IQ-트리처럼 요약 기법을 사용하는 색인 구조도 존재한다.

그러나, 고차원 색인구조의 원초적 문제인 차원의 저주현상(Dimensionality Curse)과 데이터의 수가 많아지면서 색인구조의 검색 성능이 현저히 저하된다는 문제점을 완전히 해결하는 색인구조는 존재하지 않는다. 또한 고차원 색인구조를 구성하는 데이터의 양이 매우 크기 때문에 이들을 모두 하나의 사이트에 저장한다는 것이 불가능할 수도 있다. 이런 문제를 해결하기 위해 병렬 고차원 색인구조 또는 병렬 질의 처리에 대한 연구가 90년대 중반부터 진행되었다.

최근 들어 데이터의 크기가 폭발적으로 증가하고 그 형태가 다양화되면서 SAN이라는 개념이 등장하게 되었다. SAN이란 연결된 서버에 관계없이 원거리에 분산된 저장 장치를 SAN에 연결된 네트워크 노드들이 공유할 수 있도록 해주는 초고속 통신망이다. 따라서 저장 장치는 호스트서버와의 주종관계에서 벗어나 여러 개의 서버와 공유되며 호스트서버에 의존하지 않고, 저장장치간에 공유가 가능하도록 연결되어 디스크나 테이프 장비의 공동사용, 복제기능과 고가용성을 위한 클러스터링 등을 가능하게 하는 장점을 갖는다. 그러나, SAN에는 이미지, 동영상, 지도, 캐드와 같은 고차원 데이터가 저장 관리 될 것이므로 이들을 빠르고 정확하게 검색할 수 있는 고차원 색인구조가 필요하다. 따라서 이 논문에서는 SAN 병렬 환경에 적합한 효율적인 병렬 고차원 색인 구조를 설계하고, 실험을 통해 다른 병렬 고차원 색인 구조와 비교하여 성능의 우수성을 입증 하고자 한다.

2 관련 연구

기존의 고차원 색인 구조에 대한 연구는 지난 십여 년 동안 매우 활발히 진행되었다. 앞에서 언급했듯이 기존에 제안된 고차원 색인 구조들을 분류해 보면 데이터 분할을 사용하는 색인 구조와 공간분할을 사용하는 색인 구조들로 크게 나누어 볼 수 있다. 또한, 이들의 혼합 형태의 색인구조와 기타 해싱 기법을 사용하는 색인구조, 또는 요약 기법을 사용하는 색인 구조도 존재한다.

공간 분할을 사용하는 색인구조들은 공간을 서로 겹치지 않도록 분할하여 표현한다. 이들의 특징은 비 단말 노드의 엔트리의 크기가 차원과 독립적이다. 하지만 차원이 증가할수록 데이터가 존재하지 않는 영역이 증가하고 하향 연쇄 분할(Downward cascading split)의 빈도수가 증가하여 저장공간 활용률이 현저히 떨어진다. 데이터 분할을 사용하는 색인 구조들은 MBR(Minimum Bounding Region)형태로 비 단말 노드의 엔트리를 표현하기 때문에 Dead Space가 없다. 또한 겹침을 허용하기 때문에 하향 연쇄분할 같은 문제는 발생하지 않는다. 하지만 MBR 표현 방식 때문에 차원이 증가할수록 비 단말 노드의 팬-아웃은 떨어지며 겹침 영역이 증가하게 된다. 이런 두 방법의 장점을 혼합한 방식이 바로 Hybrid-트리 이다. 여기에서는 공간분할방식의 비단말 노드 엔트리의 크기가 차원에 독립적이라는 특성과 MBR로 엔트리를 표현하며 겹침을 허용하여 하향 연쇄 분할을 피하고 있다. 하지만 완벽하게 비 단말 노드의 엔트리 크기가 차원과 독립적이라고 말할 수 없다.

그리고, 기존에 제안된 병렬 다차원 색인 구조는 크게 1P-nD의 구조와 nP-nD의 구조로 나누어 볼 수 있다. 여기서 P는 프로세서를 의미하고 D는 디스크를 의미한다. 그리고 n은 프로세서 또는, 디스크의 개수를 의미한다.

1P-nD 구조는 하나의 처리기에 다중 디스크가 연결되어 있는 병렬 환경을 말한다. 이 구조에서는 다중 디스크의 병렬 입출력을 통해서 다차원 색인 구조의 성능을 개선하고 있다. 하지만 프로세서와 다중 디스크들 사이에 단 하나의 채널이 존재하므로 데이터를 주 기억공간으로 적재하는 작업은 직렬로 처리된다. 이 부분에서 병목현상이 발생할 수 있으며 색인 구조의 성능 향상에 한계가 있게 된다. 이에 대표적인 병렬 색인구조들은 MXR-트리, Parallel X-트리, PML-트리 등이 있다.

nP-nD는 다중 처리기와 다중 디스크가 존재하는 병렬 환경이다. NOW(Network Of Workstation)와 같은 환경이 그 한 예가 될 것이다. 이런 병렬 환경에서 구축된 병렬 색인 구조들은 처리기와 디스크 입출력의 병렬성을 모두 이용할 수 있어서 보다 색인 구조의 성능을 향상시킬 수 있다. 이에 대표적인 병렬 색인 구조들로는 M-R트리, MC-R트리, DSVM(Distributed Shared Virtual Memory)상에서의 Parallel R-트리, GPR-트리, Parallel VA-화일 등이 있다.

3. 제안하는 병렬 고차원 색인구조

3.1 요구사항

기존의 병렬 다차원 색인 구조에 대한 연구를 분석해 보면 크게 1P-nD와 nP-nD의 구조로 나누어 볼 수 있다. 1P-nD의 경우는 병렬 디스크 입출력을 통해 색인구조의 성능 향상을 꾀하는 방법들이 대부분이다. 그러나 이 구조는 다중 디스크들과 처리기 사이에 채널이 하나이기 때문에 이 부분에서 병목현상이 발생할 수 있기 때문에 성능 개선에 한계가 있다. 따라서 처리기와 디스크 입출력의 병렬성을 모두 취할 수 있는 nP-nD형태의 구조가 더 바람직하다. 즉, 디스크 입출력의 병렬성 외에도 여러 처리기가 질의를 처리할 때 서로 메시지를 주고받으면서 처리기의 병렬성도 얻을 수 있다. 이때, 여러 처리기가 서로 메시지를 주고받을 때 그 메시지의 양이 최소가 될 수 있도록 해야 할 것이다.

인덱스 구조상의 요구 조건으로는 한번 색인이 구축된 후에 발생하는 데이터의 삽입에 대해서도 동적으로 색인이 구성될 수 있는 색인 구조이어야 한다. 또한, SAN에 부착된 디스크가 제거되거나 또는 새로운 디스크가 추가될 때 색인이 재구성될 수 있도록 해야 한다. 대부분의 기존 병렬 다차원 색인 구조들에서는 범위(Range) 질의만을 고려하고 있다. 하지만 이 연구에서 고려해야 하는 것은 고차원 색인 구조이다. 고차원 색인 구조에서는 K-최근접(Nearest Neighbor) 질의 역시 중요한 질의 형태중 하나이다. 이 K-최근접 질의는 알고리즘상 병렬로 질의를 처리하는데 있어 제약이 따르게 된다. 따라서, K-최근접 질의를 효과적으로 병렬 처리할 수 있는 새로운 질의 처리 알고리즘을 고려해야 한다. 또한 다중 디스크에 여러 가지 형태로 색인 구조를 배치할 수 있다. 이때 되도록 같은 데이터가 중복되지 않으면서도 성능을 최대한 얻을 수 있는 형태로 색인구조를 배치해야 할 것이다. 이런 병렬 환경에서 구축된 병렬 색인 구조들은 처리기와 디스크 입출력의 병렬성을 모두 이용할 수 있어서 보다 색인 구조의 성능을 향상시킬 수 있다.

3.2 제안하는 병렬 고차원 색인 구조

SAN환경하에서 일반적으로 이미지 검색에 사용하는 서버에 대한 비용 보다 디스크의 비용이 적다고 볼 수 있다. 따라서, 서버의 개수보다 디스크 수가 더 많이 존재한다고 볼 수 있다. [그림 1]과 같이 이러한 환경에서 디스크들을 서버의 개수만큼 디스크 그룹으로 형성하고 각 서버가 디스크 그룹을 관리한다. 서버는 탐색 결과를 통합 관리하는 주 서버가 존재하고, 나머지를 부 서버로 구분한다.

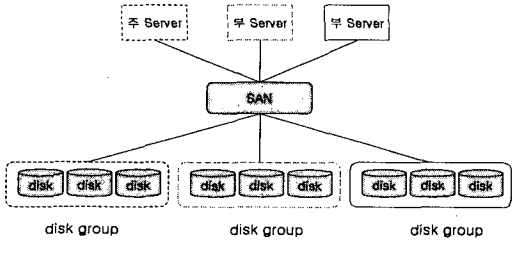

그림 1 전체 시스템 구성도

[그림 2]는 각 디스크 그룹내의 트리 구조를 보여 준다. 각 노드의 구조는 각 디스크에 있는 자식 엔트리를 하나의 MBR로 나타내고 각 디스크에 할당되어 있는 자식 엔트리를 가리키는 포인터를 유지한다. [그림2]에서 노드의 구조를 살펴보면 루트 노드의 첫 번째 엔트리의 자식 노드는 디스크 A,B,C에 있는 첫 번째 블록을 가리키게 되고 이 엔트리들을 통합해서 하나의 노드 1을 구성하게 된다. 즉, 하나의 노드는 각 디스크에 분산되게 된다. 이렇게 구성함으로써 트리의 팬-아웃을 증가시켜 트리의 높이를 줄임으로서 검색 속도를 향상시킬 수 있다. 그리고 비 단말 노드의 엔트리의 수는 디스크 수가 증가함에 따라 급격히 감소하게 되므로 고차원 데이터를 색인 하는데 적합하다.

고차원 색인 구조에서는 차원이 증가함에 따라 디스크 접근수가 급격히 증가하게 된다. 따라서, 병렬 구조에 있어서 중요한 성질인 최소 활성 디스크(Minimum Load) 보다는 모든 디스크를 동시에 접근하는 균등 활성 디스크(uniSpread) 성질을 극대화해서 동시에 많은 노드를 접근하는 트리 구조가 필요하게 되었다. 제안된 구조를 통해 항상 균등 활성 디스크의 성질을 최대화할 수 있게 되어 범위 질의 및 K-최근접 질의시 병렬로 입출력을 처리하게 된다.

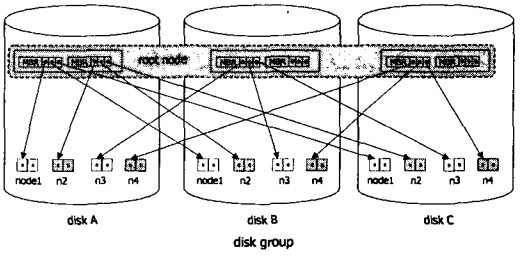

그림 2 각 디스크 그룹에서의 트리 구조

3.3 삽입

전체 SAN 환경에서 엔트리가 삽입되는 과정을 [그림 3] 과 같이 나타낼 수 있다. 엔트리가 하나씩 삽입되면 디스크 그룹에 라운드로빈 방법으로 하나씩 할당하게 된다. 고차원으로 갈수록 분산을 계산하기 위한 비용이 많이 들고 성능은 다른 방법과 비교했을 때 차이가 많이 나지 않기 때문에 계산에 대한 비용이 적게 들고 구현하기 쉬운 라운드 로빈 방법을 사용한다[9]. 삽입시 그룹 내에서 페이지 할당 방법은 해당 노드의 페이지가 할당되지 않은 디스크에 할당한다. 모든 디스크에 페이지가 할당되었을 때 또 하나의 페이지가 삽입되면 노드가 오버플로우 발생으로 노드가 분할한다. 고차원 색인 구조에서 하나의 페이지의 저장 공간 활용률이 약 60-70% 정도이기 때문에 하나의 노드를 접근할 때 모든 디스크를 접근하는 병렬성을 최대화할 수 없다. 그러므로 여러 노드를 병렬로 입출력을 하기 위한 방법이 필요하게 되었다. 여러 노드를 병렬로 입출력을 하기 위해서는 분할된 노드를 다른 디스크에 할당하여 병렬로 접근할 수 있게 해야 한다. 따라서 각 디스크의 페이지수가 가장 적은 디스크부터 해당 노드의 페이지가 할당되게 하였다. 이렇게 함으로써 특정 디스크에 페이지가 집중되는 것을 방지할 수 있다. 분할된 페이지를 다른 디스크에 분할시킴으로써 고차원 색인 구조에 있어서 가장 기본적인 질의 형태인 범위 질의에 있어서 여러 노드를 병렬로 읽을 수 있게 된다.

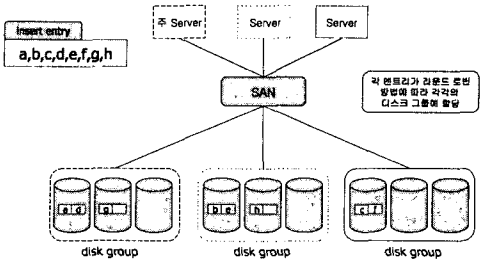

그림 3 전체 SAN 환경에서 삽입 과정

3.4 검색

고차원 색인 구조에 있어서 가장 기본적인 검색 형태인 범위 질의 과정을 살펴보면 루트 노드에 접근해서 범위에 포함되는 엔트리를 선택해서 디스크 관리자에게 다음 읽을 노드를 관리하게 된다. 각 디스크 관리자는 병렬로 각각의 하위 노드를 읽게 된다. [그림 4]의 범위 질의 탐색의 예제를 보여 주고 있다. 루트 노드에 접근해서 엔트리를 1-8까지 읽고 범위에 해당하는 엔트리 2,4,7을 선택하고 각 엔트리의 자식 노드에 해당하는 포인터를 디스크 관리자에게 전달된다. 엔트리 2의 포인터 A3, E1의 의미는 A 디스크의 세 번째 블록을 의미하고 E1은 E 디스크의 첫 번째 블록을 의미한다. 즉, 단말 노드 2를 가리키게 된다. 엔트리 2,4,7에 대한 포인터를 디스크 관리자가 관리를 하고 A 디스크에서 읽을 블록은 3과 5블록이고 B 디스크에서는 3번째 블록을 읽게 된다. 디스크 관리자는 각 디스크에서 읽을 블록에 대한 정보를 관리하게 된다. 단말노드에 2, 4, 7에 접근할 때 각 하나의 노드에 대해서 병렬로 접근하게 되면 디스크 접근수가 3이므로 루트 노드에 대한 접근 수를 포함한 총 접근 수는 4가 되게 된다. 여기서 접근 수를 줄이기 위해 디스크 관리자에 의해서 각각의 디스크에서 A3,B3,C3,D4,E1의 블록을 병렬로 읽게 되고 다음으로 디스크 A에서 A5와 디스크E의 E2을 동시에 병렬로 읽게 되어서 단말노드에 접근 시간은 2가 되고 루트 노드를 포함한 접근 수는 3이 되므로 노드 하나에 대해서 병렬로 접근할 때 보다 접근 수가 작게 된다.

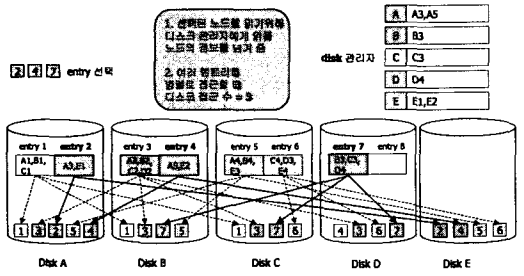

그림 4 범위 질의 탐색

다음으로 고차원 색인구조에 있어서 범위 질의와 함께 자주 발생하는 K-최근접 질의에 대해서 살펴보면 본 연구에서는 네 가지 방법을 제시했다. 첫 번째 방법으로 범위 질의 형태는 K-최근접 질의 방법보다 노드의 입출력시 병렬성을 최대화 할 수 있다는 것을 이용한다. K-최근접 질의가 들어오면 우선 [그림 5]와 같이 주 서버에서 부분적인 K-최근접 질의 검색을 수행한다. 여기서 부분적인 K-최근접 질의 검색이란 기존의 방법으로 K-최근접 질의 탐색을 수행하고 최종의 K 개의 결과를 찾는 것이 아니라, 기존의 K-최근접 질의 탐색시에 첫 번째 K-결과를 얻으면 K 번째에 해당하는 객체와의 유사도를 구하여 범위 질의 형태로 변환한다. 즉, 완전한 K-최근접 검색을 하는 것이 아니라, 부분적인 K-최근접 검색을 하고 K 번째 결과를 범위질의 형태로 전환하고 범위 질의를 다른 부 서버에 전달한다. 부 서버에는 병렬성을 최대화해서 검색 결과를 주 서버로 전달하고, 주 서버에는 각 서버의 결과를 통합하여 유사도 순으로 정렬해서 최종 K 개의 결과를 얻는다. 이 방법은 범위 질의 형태로 변환해서 병렬성을 최대화하는 장점이 있다. 그러나 주 서버에서 얻은 범위질의의 크기가 크면 필요 없는 연산을 많이 할 수가 생기게 되는 단점을 가지게 된다. 두 번째 방법은 첫 번째 방법이 주 서버에서 부분적인 K-최근접 질의 검색을 하기 때문에 범위 질의 형태로 변환했을 때 범위 질의의 크기가 너무 크면 탐색 효율이 떨어 질 수 있다. 이러한 단점을 줄이기 위해서 부분적인 K- 최근접 질의 검색을 주 서버에서만 하는 것이 아니라, 주 서버와 부 서버에서 동시에 부분적인 K-최근접 질의 검색을 해서 주 서버 전달하고 주 서버에서는 각 서버에서 전달된 범위 질의 중에서 가장 범위가 작은 질의를 각 서버에 전달하고 첫 번째 방법처럼 범위 질의를 수행하게 된다. 두 번째 방법은 첫 번째 방법의 문제점을 조금 개선 할 수 있으나, 완전히 해결하지 못 한다. 세 번째 방법으로는 각 서버에 K-최근접 질의가 들어오면 각 서버에서 부분적인 K-최근접 질의를 수행하고, 첫 번째 K 결과를 찾으면 K번째에 대해서 범위 질의 형태로 변환하여, 다음 단계에서는 범위 질의를 수행하여 다시 K 결과를 얻고, K 번째 객체가 바뀌게 되면 범위 질의로 다시 변환하여 질의의 크기를 줄여 범위 질의를 수행하게 된다. 최종 K 결과를 찾을 때까지 수행하여 결과 K 개를 찾으면 주 서버로 전달한다. 주 서버에는 모든 서버에 찾은 결과를 통합하여 최종 결과를 얻는다. 이 방법은 트리의 높이가 클 때 유용하다. 마지막으로 네 번째 방법은 질의가 들어오면 각각의 주 서버와 부 서버에서 K-최근접 질의 방법을 수행한다. 여기서는 하나의 노드에 대해서만 병렬로 입출력을 수행하게 된다. 이 방법은 앞의 세 가지 방법에 비해 구현이 간단하고 범위 질의 형태로 변환하지 않기 때문에 여러 노드에 대해서 병렬성을 얻을 수 없지만 1-3 방법에서 범위 질의 변환시 필요 없는 연산을 하는 단점은 발생하지 않는다.

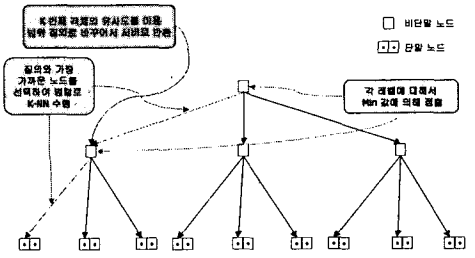

그림 5 주 서버에서 부분적인 K-최근접 질의 탐색 방법

4. 결론

이 논문에서는 SAN 환경에 적합한 병렬 고차원 색인 구조를 제안하였다. 제안된 병렬 고차원 구조에서 각 서버는 디스크 그룹을 관리하고, 각 디스크 그룹에 엔트리를 할당하는 방법은 가장 간단한 라운드 로빈 방법을 사용한다. 하나의 노드를 여러 디스크에 분산시켜 팬-아웃을 증가시키고, 트리의 높이를 줄임으로서 검색 성능을 향상시킨다. 각 디스크 그룹 내에서 노드 분할 방법은 각 레벨에서 페이지의 수가 가장 작은 디스크에 할당을 해서 노드를 다른 디스크에 분산시켜 병렬성을 최대화한다. 질의 처리시 범위 질의 방법은 여러 노드를 병렬 입출력을 수행하는 방법을 제안한다. 그리고 K-최근접 질의 방법은 각 서버에서 K-최근접 질의를 수행하는 방법과, 범위 질의의 병렬성을 이용하여 K-최근접 질의의 성능을 향상시키기 위한 방법을 제안했다. 향후 시뮬레이션 결과를 통해서 고차원 색인 구조의 우수성을 보이며, 고차원 구조의 성능 개선 요소를 찾고, 이를 통해 제안한 고차원 색인 구조의 성능을 더욱 더 개선한다.

5. 참고 문헌

[1] 이석희, 유재수, 조기형, 허대영, "CIR-Tree : 효율적인 고차원 색인 기법", 한국정보과학회 논문지(B), 한국정보과학회, 제26권 제6호, pages 724-734, Jun 1999.

[2] K.I. Lin, H. Jagadish, and C. Faloutsos. "The TV-tree : An Index dStructure for High Dimensional Data", VLDB Journal, Vol 3, pp.517-542, 1994.

[3] S. Berchtold, D. A. Keim, H-P. Kriegel, " The X-tree : An Index Structure for High-Dimensional Data", Processing of the 22nd VLDB Conference.

[4] K. Chakrabarti and S. Mehrotra. "The Hibrid Tree : An Index Structure for High-Dimensional Feature Spaces." Proc. of ICDE, 1999

[5] Ibrahim Kamel and Christos Faloutsos, "Parallel R-trees," ACM SIGMOD'92, 6 1992, California

[6] N. Koudas, C. Faloutsos, and I. Kamel, "Declustering R-trees on Multi-Computer Architectures," Technical Research Report ISR 1994, 1994

[7] Bernad Scnnitzer and Scott T.Leutenegger, "Master-Client R-trees: A New Papallel R-tree Architecture," IEEE, 1999

[8] Botao Wang, Hiroyuki Horinokuchi, Kunihiko Kaneko, and Akifumi Makinouchi, " Parallel R-tree Search Algorithm on DSVM," IEEE, 1999.

[9] Roger Weber, "Parallel VA-File," VLDB 1999