클러스터를 이용한 공간데이타 디클러스터링

곽지숙, 김학철, 이기준
부산대학교 전자계산학과
e-mail : {jskwak,hckim,lik}@quantos.cs.pusan.ac.kr

Declustering Spatial Objects by Clustering for Parallel Disks

Ji-Sook Kwak, Hak-Chul Kim and Ki-Joune Li

Department of Computer Science, Pusan National University

요 약

지리정보시스템과 같은 공간 데이터베이스에서 다루는 데이터는 대용량이며, 사용자의 다양한 질의에 따라 빠르게 접근 될 수 있어야 한다. 그런데 이때 성능의 대부분이 디스크 접근시간에 의해 영향을 받으므로 접근시간을 줄이는 기술이 필요하다. 이는 다수의 디스크 공간에 데이터를 분산하여 저장하는 디클러스터링 방법을 사용함으로써 효과적인 성능 향상을 기대할 수 있다. 효과적인 디클러스터링 방법은 주어진 질의에 대하여 동시에 접근될 가능성이 있는 공간 객체를 다른 디스크에 각각 저장함으로써 한번에 접근하는 병렬성을 높일 수 있다. 그러나 하나의 디스크에 할당 가능한 공간 객체들을 서로 다른 디스크에 할당하는 것은 오히려 성능의 저하를 초래할 수 있다. 이러한 두 가지 조건을 동시에 만족하기 위해서는 공간 객체들을 클러스터링 한 후, 클러스터 단위로 디스크로 할당하는 것이 효과적이다. 이전에 제시된 디클러스터링 방법들은 이러한 요소를 고려하지 않았다. 이에 본 논문에서는 주어진 공간 객체들에 대해서 일정한 크기의 클러스터를 만들고 클러스터 단위로 디클러스터링 하여 효율적인 성능 향상을 보이는 새로운 방법에 대해서 제시하고자 한다. 또한 이전에 제시되어졌던 여러가지 디클러스터링 방법들과의 비교실험을 통해, 본 논문에서 제시한 방법이 가장 효과적인 방법임을 밝히고자 한다.

1. 서론

지리정보 시스템에서 다루는 데이터는 공간 객체로써, 형태가 복잡하다는 뿐만 아니라 데이터 크기가 크기 때문에 보조 기억장치에 저장되어 있으며 질의 처리를 위해서 많은 디스크 입출력을 필요로 한다. 따라서, 이러한 빈번한 디스크 입출력은 시스템의 전체 성능을 저하시키는 요인이 된다. 이를 해결하기 위해서는 우선적으로 공간 데이터를 주어진 다수개의 병렬 디스크상에 효과적으로 분산시켜 저장해야 하는데, 지금까지 이에 대한 연구가 많이 이루어졌다[DG92, FB93, MAS96, MS96] .

효과적인 디클러스터링 방법은 주어진 질의에 대하여 동시에 접근될 가능성이 있는 공간 객체는 다른 디스크에 저장함으로써 입출력 병렬성을 높일 수 있다. 하지만, 하나의 디스크에 할당 가능한 공간 객체들을 서로 다른 디스크에 할당하는 것은 오히려 성능의 저하를 초래할 수 있다. 이러한 두 가지 조건을 동시에 만족하기 위해서는 공간 객체들을 클러스터링 한 후, 클러스터 단위로 디스크에 할당하는 것이 효과적이다. 이전에 제시된 디클러스터링 방법들은 이러한 요소를 고려하지 않았다. 이에 본 논문에서는 동시에 접근될 가능성이 높은 공간 객체의 단위를 클러스터의 개념을 도입하여 클러스터 단위로 다수개의 디스크에 디클러스터링 하는 새로운 방법을 제시하였다. 여기서 사용한 클러스터링 방법은 이전에 제시되었던 방법들 중에서 효율적인 SMTin [KKL97]을 사용하였다. 또한 기존의 공간 색인 방법들 중에서 R*-tree를 이용하여 디클러스터링 하는 방법도 함께 제시하고 비교하였다.

본 논문의 구성은 다음과 같다. 2 장에서는 기존에 연구된 디클러스터링 방법에 대해서 살펴보고 3 장에서는 본 논문에서 사용한 클러스터링 방법에 대해서 소개하도록 한다. 4 장에서는 제시한 방법을 이용하여 디클러스터링하는 방법을 소개하고 5 장에서는 본 논문에서 제시한 디클러스터링 방법과 이전에 제시된 그리드 기반의 디클러스터들과 비교실험을 통한 성능 측정 결과를 보이도록 한다. 마지막으로 6 장에서는 결론 및 향후 연구과제에 대해서 제시하도록 한다.

2. 관련 연구

지금까지 데이터의 분산을 통한 병렬 입출력을 위한 연구는 많이 이루어졌다.

먼저 가장 간단한 방법으로 라운드 로빈 방법, 범위방법, 해쉬 분할 방법들이 있다[DG92]. 그리고 다수의 속성을 가지고 디스크에 데이터를 할당하고 있는 그리드를 기반으로 하는 방법들이 있다. 이 방법은 각 속성값의 도메인을 일정 크기로 분할한 후, 각 그리드에 속하는 객체들은 같은 디스크에 저장한다. 그리고 여기서 인접하고 있는 그리드는 가능하면 다른 디스크에 저장하는 것을 원칙으로 한다. 가장 간단한 방법인 DM(Disk modulo method)은 범위 질의시 디스크 수로 나눈 나머지 값에 따라 디스크에 디클러스터링 한다[FB93]. 다음으로 FX(Fieldwise exclusive-or method)는 랜덤수 일반화 방식으로 가능한 고른 분산을 위하여 exclusive-or 값으로 계산해서 필요한 비트만을 가져와서 디스크에 디클러스터링한다[FB93] ECC(Error correcting code method)는 패리티 체크 방정식에 의해서 나온 값으로 고르게 디스크에 디클러스터링 한다[FB93] 마지막으로 HCAM(Hilbert-curve Allocation method)은 공간 포화곡선 중에서 공간 지역성을 가장 잘 표현하는 힐버트 곡선의 순서를 유지하며, 디스크에 라운드 로빈방식으로 디클러스터링한다[FB93] 이 힐버트 곡선은 하나의 그리드를 한번씩만 순회하며, 근접한 것들끼리 순서화 되어 있으므로, 다른 방법들보다 가장 좋은 성능을 보장하게 된다.

그러나 이러한 방법들의 가장 큰 문제는 그리드 파일에서 발견되는 것과 마찬가지로 균등한 분포를 가지는 공간 객체들의 응용에서는 좋은 성능을 보여주나, 비균등한 분포를 가지는 경우에는 성능이 매우 떨어진다. 더욱이 대부분의 응용분야에서 공간 객체들은 비균등한 분포이므로, 이 방법을 일반적으로 사용하기는 힘들다. 그러므로 본 논문에서는 클러스터를 통한 공간 데이터의 디클러스터링 알고리즘을 새로이 고안함으로써 비균등한 데이터 분포에도 효과적인 디클러스터링 방법을 제시하였다.

3. 클러스터링 방법

공간상으로 가까운 객체들은 질의 조건을 동시에 만족할 확률이 높다. 따라서 이러한 공간 객체들의 집합을 하나의 단위로 관리하는 것이 질의 처리시 유리하다. 본 논문에서는 공간 객체들을 클러스터링 하는 방법으로 [KKL97]에서 제시한 SMTin 방법을 이용하여 기존에 제시된 많은 공간 색인 방법들 중 클러스터링의 성능이 가장 뛰어난 것으로 평가되는 *R**-tree 의 단말노드를 이용하여 클러스터를 형성하였다.

SMTin 은 공간 데이터에 대한 클러스터를 생성할 때 주어진 공간 데이터에 대하여 딜로니 삼각분할 방법을 적용시켜 불규칙 삼각망 데이터를 생성한 뒤에 이 불규칙 삼각망 데이터를 이용하여 클러스터링을 수행한다. 불규칙 삼각망은 모든 공간 데이터의 연결성과 연결되어 있

는 두 공간 객체는 주어진 공간 데이터 집합 내에서 가장 가까운 두 객체임을 의미하는 인접성의 특징을 가진다 그러므로 생성된 클러스터링의 집합은 디클러스터링을 위한 전단계로 우리가 원하는 조건에 부합하는 단위가 될 수 있다.

4. 클러스터들의 디클러스터링

이 장에서는 앞에서 제시한 클러스터링을 이용하여 디클러스터링 하는 방법에 대해서 소개하도록 한다. 공간적 근접성을 유지하면서 클러스터를 형성하고 난 다음에는 인접한 클러스터들을 가능하면 다른 디스크에 할당하는 것이 중요하다 또한 하나의 클러스터에 속하는 데이터의 크기는 디스크 블록크기와 같아야 하는데, 형성된 클러스터가 디스크 블록크기보다 클 경우 하나의 클러스터를 찾기 위해서 여러 번의 디스크 접근을 해야 하는 문제점이 발생할 수 있다 3 장에서 제시한 SMTin 방법은 클러스터의 특성을 잘 반영하지만, 공간적으로 일정 거리 이내에 있는 객체는 같은 클러스터에 포함되므로 하나의 클러스터 내의 데이터 크기는 블록크기를 초과하는 경우도 있다 그러므로 본 논문에서는 클러스터 내의 데이터의 수가 블록 크기를 초과할 경우에는 블록 크기만큼 데이터를 묶어서 새로운 클러스터를 생성하도록 수정하였다 또한 범위 값을 기본으로 하는 SMTin 클러스터링 방법에서는 어떤 클러스터에도 포함되지 않고 단독으로 존재하는 예외자가 존재하게 된다. 이러한 예외자를 효과적으로 저장 관리하는 것은 디클러스터링을 하는 방법에서 성능에 많은 영향을 주게 된다 그러므로, 본 논문에서는 예외자의 수가 하나의 디스크 블록 크기를 초과하지 않도록 조정한 후, 1 번 디스크의 첫 번째 블록에 지정하였다 예외자 블록은 질의 처리시에는 항상 메모리에 유지하고 예외자 블록을 먼저 조사한 후, 질의 조건을 만족하는 다른 블록을 디스크로부터 병렬적으로 적재하는 방법을 사용하였다 그것은 SMTin 의 클러스터링 방법의 특성상 공간상으로 가까운 거리에 있는 객체 순으로 클러스터에 포함하고 있기 때문이다 초기에 공간 데이터의 클러스터를 할 때, 클러스터링을 위한 클러스터 생성임계값(CCT Cluster Construction Threshold)을 조정하여 예외자의 수가 하나의 디스크 블록 크기보다 작도록 클러스터를 형성한다 그리고 단일 공간 객체의 수가 블록 크기를 초과하는 클러스터에 대해서는 재분배를 통해서 클러스터의 크기를 조정한다 클러스터들 만들고 난 후에는 클러스터 단위로 디스크에 저장하게 되는데, 저장하는 순서는 클러스터를 포함하는 최소경계 사각형(MBR)의 중심점의 hilbert 값으로 정렬하여 라운드 로빈 방식으로 디스크에 할당한다. 구체적인 방법은 알고리즘 1 과 같다

```
Algorithm Declustering 1
Input  set of points
Output  set of clusters pair of (C_i, disk number)
Make Initial clusters C_1, C_2, . . , C_n
While number of outlier is lager than N(B)   // 블록당 총 갯수
   Recalculate CCT and make clusters
For each clusters Ci ( I = 1 = n)
      If  N(C_i) > N(B)   Redistribute(C_i)
For each final clusters C_1, C_2,  , C_m
   Calculate center of cluster mbr Ci(x, y)
   Assign disk Hilbert-value (Ci(x,y))  mod N(D)
```

< 클러스터를 통한 디클러스터링 알고리즘 1 >

또한 범위 값을 기본으로 하는 SMTin 클러스터링 방법에서는 어떤 클러스터에도 포함되지 않고 단독으로 존재하는 예외자가 존재하게 된다 이러한 예외자를 효과적으로 저장 관리하는 것은 디클러스터링을 하는 방법에서 성능에 많은 영향을 주게 된다. 그러므로, 본 논문에서는 예외자의 수가 하나의 디스크 블록 크기를 초과하지 않도록 조정한 후, 1 번 디스크의 첫 번째 블록에 저장하였다 예외자 블록은 질의 처리시에는 항상 메모리에 유지하고 예외자 블록을 먼저 조사한 후, 질의 조건을 만족하는 다른 블록을 디스크로부터 병렬적으로 적재하는 방법을 사용하였다

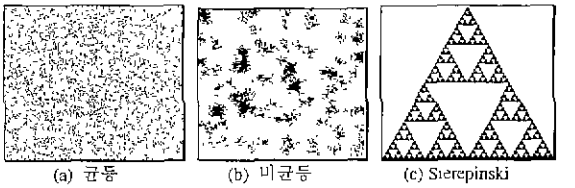
두번째로 제안 한 것은 효과적인 공간 색인 기법인 R*-tree 색인에 의한 데이터 디클러스터링 방법이다 즉, R*-tree 색인에 의해 데이터 객체를 삽입하여 단말 노드를 형성한 다음, 단말 노드의 중심점을 힐비트 값으로 정렬하여 라운드 로빈 방식으로 디스크를 할당하는 방식이다. R*-tree 방식에서는 예외자는 존재하지 않으므로 이를 고려하지 않아도 된다 구체적인 방법은 알고리즘 2 와 같다

```
Algorithm Declustering2
Input  set of points
Output  set of  clusters
Insert to R*-tree
For each leaf node N_1, N_2,  N_n
   Calculate center of node mbr Ni(x, y)
      Assign disk Hilbert-value(Ni(x,y))  mod N(D)
```

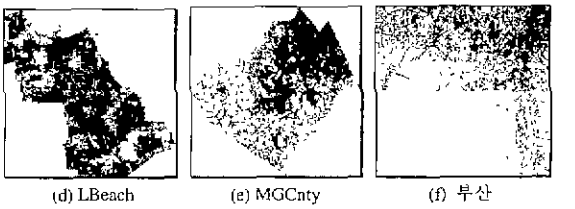
< R*-tree 색인에 의한 데이터 디클러스터링 알고리즘 2 >

5. 실험결과

이 장에서는 기존에 제시된 많은 디클러스터링 방법들 중 일반적인 방법인 그리드 기반의 디클러스터링 방법과 본 논문에서 제시한 새로운 방법들을 실험을 통해 비교하였다. 디클러스터링 방법의 성능에 영향을 주는 요소는 데이터의 분포와 질의 크기, 디스크수, 페이지 크기 등이 있을 수 있다 먼저, 실험을 위해서 본 논문에서는 다양한 분포를 가지는 임의의 데이터를 생성하였으며, 실제 데이터도 함께 사용하였다. 다음의 그림 1 과 표 1 은 실험에 사용한 데이터의 분포와 특성을 정리한 것이다



(a) 균등 (b) 비균등 (c) Sierepinski



(d) LBeach (e) MGCnty (f) 부산

< 그림 1 테스트 데이터 분포 >

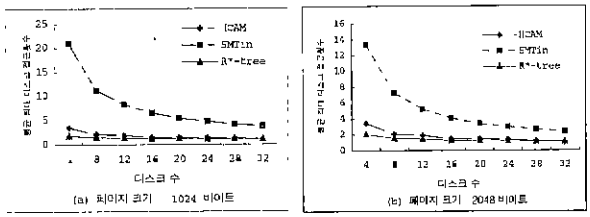
분류	이름	점 개수	설명
가상 데이터	균등	10,000	
	비균등	10,000	
	Sierepinski	9,841	
실제 데이터	LBCnty	36,548	road intersections, Long Beach Cty, CA
	MCnty	27,282	road intersections, Montgomery Cty, MD
	부산	10,213	DXF 부산지역의 feature 명을 나타내는 텍스트

< 표 1. 테스트 데이터 >

질의가 주는 영향을 측정하기 위해서 데이터 영역에 집중된 영역질의와 데이터 공간에 균등하게 분포하는 영역 질의를 임의로 생성하였다 또한 영역 질의 크기에 따른 성능 측정을 위해서는 각 변의 길이가 해당 도메인의 1~3%, 3~10%, 10~20% 크기에 해당하는 영역 질의를 생성하여 이를 각각 작은 질의, 중간 질의, 큰 질의로 명하였다 또한 제시한 방법의 성능 측정을 위해서 주어진 다수개의 병렬 디스크에 대해서 질의가 주어졌을 때, 가장 많은 디스크를 접근하는 횟수를 기준으로 성능 비교를 하였다

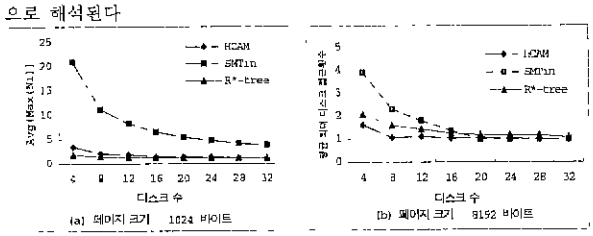
본 논문에서는 먼저 이전에 연구되어졌던 그리드 기반의 DM, FX, HCAM 과 같은 디클러스터링 방법들을 구현하여 실험하였다 그 결과 가장 좋은 성능을 보이는 HCAM 과 본 논문에서 제시한 SMTin 을 이용한 디클러스터링 방법 및 R*-tree 색인에 의한 디클러스터링 방법을 실험하여 그 성능을 비교하였다 그림 2 는 데이터 영역에 균일하게 분포하는 데이터에 대해서 중간 질의를 주고, 페이지 크기를 고정한 다음 디스크 수를 증가 시켰을 때의 성능 비교를 나타낸 그래프이다. 균등하게 분포하는 데이터에 대해서는 HCAM 방법이 좋은 성능을 나타냄을

알 수있다. 이는 데이터 영역 전반에 걸쳐서 그리드가 골고루 형성되기 때문이다 R*-tree 방법은 모든 경우에 대해서 비교적 고른 성능을 나타내었으며 SMTin 방법은 페이지 크기가 작을 경우에는 성능이 현저히 떨어지는 반면, 페이지 크기가 커짐에 따라 효율적인 성능 향상을 보임을 알 수 있다. 이것은 페이지 크기가 작을 경우에 하나의 클러스터에 포함되는 객체의 수가 블록 크기를 초과하여 재분배할 때, 클러스터의 특성을 상실하여 일어나는 현상이다



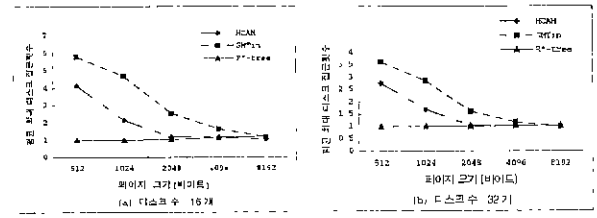
< 그림 2 균등 분포 데이터의 페이지 크기에 따른 성능평가 >

그림 3 은 비균등한 데이터에 대해서 데이터 영역에 집중된 중간 질의 영역에 대해서 디스크수와 페이지 크기의 변화에 따른 성능 측정을 나타낸 것이다 HCAM 방법과 SMTin 방법은 페이지 크기가 커짐에 따라서 성능의 향상이 있는 반면 R*-tree 를 이용한 디클러스터링 방법은 일정 크기 이상에서는 오히려 성능이 저하되는 것을 알 수 있다. 이는 페이지 크기가 증가함에 따라 R*-tree 의 노드 조건을 만족시키기 위해서 공간상으로 멀리 떨어져 있는 객체들도 같은 노드에 포함하기 때문으로 해석된다



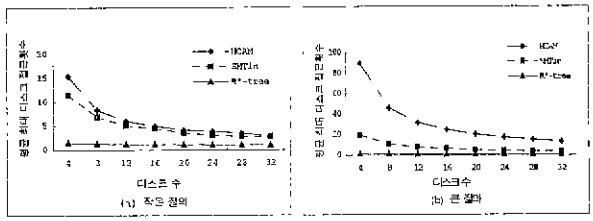
< 그림 3 비균등 분포 데이터의 페이지 크기에 따른 성능평가 >

그림 4 는 Sierepinski 비균등 분포 데이터에 대해서 삼각형 주위에 집중되어 있는 질의에 대해서 디스크수가 각각 16, 32 일 때, 페이지 크기의 변화에 따른 성능 평가를 나타낸 것이다. HCAM 와 SMTin 은 페이지 크기가 증가함에 따라 평균 최대 디스크 접근횟수가 줄어드는 반면, R*-tree 방법은 증가하였다 이는 R*-tree 의 노드 조건을 만족하기 위해서 공간상으로 멀리 떨어져 있는 객체도 같은 노드에 저장함으로써 노드간의 겹침이 많이 생기기 때문이다 또한 SMTin 방법은 하나의 블록 크기보다 많은 객체가 하나의 클러스터에 포함될 경우에 재분배해야 하는데, 이 때 새로 생성되는 클러스터는 공간적 근접성을 잘 보장하지 못하기 때문으로 해석된다

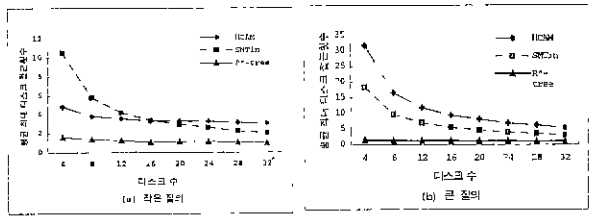


<그림 4 Sierepinski 비균등 데이터의 페이지 크기에 따른 성능평가 >

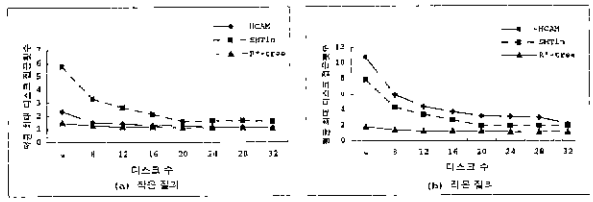
아래 그림 5, 6, 7 은 실제 데이터에 대해서 질의 사각형의 크기가 각각 데이터 영역의 1~3%, 10~30%에 해당하는 데이터 영역에 균일하게 분포하는 질의에 대해서 페이지 크기가 4096 바이트일 때, 디스크수의 변화에 따른 성능 측정을 나타낸 것이다 이는 앞에서 실험한 가상 데이터 중 비균등한 데이터 분포와 비슷한 결과를 나타내었다 사용한 실제 데이터는 대부분 넓은 지역에 대규모의 클러스터를 이루고 있음을 알 수 있다. (그림 1 참조) 이러한 데이터 분포는 SMTin 을 이용한 디클러스터링 방법에서는 효과적으로 클러스터를 만들지 못한다 대부분의 데이터가 하나의 클러스터를 형성하기 때문에 효과적인 재분배 알고리즘을 적용하지 못 할 경우 클러스터를 포함하는 최소 경계 사각형의 겹침 현상이 많이 발생하게 된다 그러므로 질의 처리시 더 많은 디스크 접근을 필요로 하게 된다 이러한 경우는 그림 6 의 (a)와 그림 7 의 (a)에서 알 수 있듯이 HCAM 방법이 더 효과적임을 알 수 있다. 하지만 질의 사각형이 클 경우에는 SMTin 이 HCAM 보다 성능이 뛰어남을 알 수 있다. 이는 전체 데이터를 고려했을 경우에는 SMTin 이 HCAM 보다 데이터의 분포 상태를 더 잘 반영하기 때문으로 보인다



< 그림 5 LBCnty 실제 데이터의 디스크 수에 따른 성능평가 >



< 그림 6. MGCnty 실제 데이터의 디스크 수에 따른 성능평기 >



< 그림 7 부산지역 실제 데이터의 디스크 수에 따른 성능평가 >

6. 결론 및 향후 연구

본 논문에서는 대용량의 공간 데이터의 디스크 접근 횟수를 줄이기 위해 클러스터 단위로 디클러스터링 하는 효율적인 방법에 대해서 제시하고 실험을 통해 평가하였다 디클러스터링을 위한 클러스터는 SMTin 방법과 R*-tree 의 단말 노드를 사용하였으며, 이 클러스터를 포함하는 최소 경계 사각형의 중심점을 힐버트 곡선값에 따라 정렬하여 라운드 로빈 방식으로 디스크를 할당하였다 실험에 의하면 이러한 방법은 클러스터를 고려하지 않은 디클러스터링 방법에 비해서 디스크 접근을 적게 하있으며, 특히 데이터 영역에 집중된 질의에 대해서 좋은 성능을 나타내었다

향후 연구과제는 SMTin 방법에서 하나의 클러스터에 블록크기보다 많은 객체들이 포함될 때 효과적으로 재분배하는 방법에 대한 연구와 다른 클러스터링 방법을 적용한 비교실험이 필요하다

참고문헌

[DG92] D DeWitt and J Gray Parallel database systems The future of high performance database systems Communication of the ACM, June 1992

[FB93] C Faloutsos and P Bhagwat Declustering using fractals In the 2nd International Conference on parallel and DIS, pages 18-25, January 1993

[KF92] I.Kamel and C Faloutos Parallel R-Trees, In Proceeding of ACM SIGMOD, 1992

[KKL97] I.S Kang, T W Kim and K J Li, A Spatial Data Mining Method by Delaunay Triangulation, In the 5th Proceeding on ACM-GIS, pages 35-39, 1997.

[MAS96] B Moon, A Acharya, and J Saltz Study of scalable declustering algorithms for parallel grid files In proceedings of IPPS, April 1996

[MS96] B. Moon, and J H Saltz, Scalability Analysis of Declustering Methods for Multidimensional Range Queries, In Proceedings of IEEE TKDE, 1996