# 디클러스터링된 공간 데이타베이스에서의 다중 공간 질의 처리

박영민°, 전봉기, 서영덕, 홍봉희
부산대학교 컴퓨터공학과

# Multiple Spatial Query Processing in Declustered Spatial Databases

**Park Youngmin°, Jun Bonggi, Seo Youngdeok, Hong Bonghee**
**Dept. of Computer Engineering, Pusan National University**

## 요 약

다중 공간 질의는 동시에 2개 이상 수행되는 영역 질의로 정의되며 인터넷 기반 지도 보기 응용의 주요 연산이 되므로, 질의 처리 속도의 향상을 위해서 병렬로 처리되어야 하고 디스크 입출력 비용을 최대한 줄일 필요가 있다. 그런데 다중 공간 질의는 디스크 입출력 비용을 개선하기 위해 다중 CPU/다중 디스크 구조 상에서 디클러스터링을 수행하더라도 디스크 암의 탐색이 발생하는 문제점이 있다.

이 논문에서는 디클러스터링된 공간 데이타베이스에서 다중 공간 질의를 처리할 때 발생하는 문제점인 질의 간 암의 탐색을 분석하고, 해결 방안으로 질의 간 위치 관련성과 질의 처리 이력을 이용한 질의 스케줄링 기법을 제안하고 구현하였다. 실험을 통한 성능 평가 결과, 질의 스케줄링을 수행할 경우 디스크 입출력 비용을 줄일 수 있어 다중 공간 질의 처리시의 성능을 개선할 수 있는 것으로 나타났다.

## 1. 서론

다양한 응용 분야에서 사용되고 있는 GIS(지리정보시스템)에서의 공간 질의의 처리 속도 개선은 해당 응용 분야의 성능 향상에 중요한 부분을 차지하고 있다. 공간 질의 중 많이 사용되는 영역 질의(region query)의 처리 속도 개선이 중요한데, 최근 많이 사용되는 클라이언트-서버 구조의 인터넷 기반 지도 보기 응용에서는 중요성이 더욱 강조되고 있다[3].

인터넷 기반 지도 보기 응용은 다수의 클라이언트가 단일 서버에 저장된 대용량의 기본도에서 자신의 관심 영역을 읽어와서 표시해 주는 응용으로써, 클라이언트의 수가 수십 개에서 수백 개에 이르는 특징이 있다. 이러한 지도 보기 응용의 특성상 서버는 다수의 클라이언트가 동시에 수행한 공간 질의들을 빠른 시간 안에 처리한 후 그 결과를 제공해야 한다. 즉 [3]에서 정의한 다중 공간 질의를 효율적으로 처리할 필요가 있다. 이 논문에서는 다중 공간 질의를 처리할 때의 문제점을 제시한 후, 이를 효과적으로 해결하기 위한 알고리즘을 제시하고 구현한 성능 평가 결과를 제시한다.

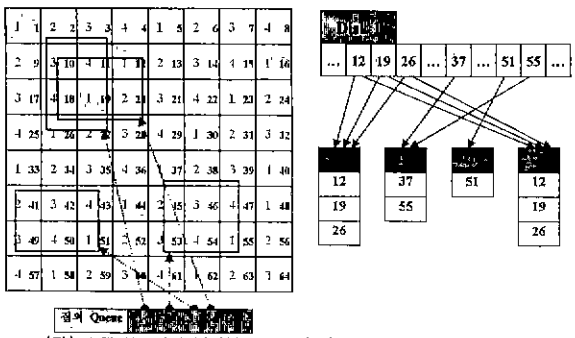
이 논문의 구성은 다음과 같다. 2장에서 다중 공간 질의의 문제점을 분석하고, 3장에서는 다중 공간 질의의 특징과 문제점을 해결하기 위한 방법을 제시한다. 그리고 4장에서 구현한 성능 평가 결과를 기술한 후 5장에서 결론 및 향후 연구로 끝을 맺는다.

## 2. 다중 공간 질의의 문제점

다중 공간 질의는 정의에서 볼 수 있듯이 '**다수의**' 공간 영역 질의가 '**동시에**' 수행되고 질의들을 빠른 시간 내에 처리해야 한다[3]. 따라서, 단일 CPU를 기반으로 한 서버보다는 다중 CPU를 기반으로 한 병렬 질의 처리기 마땅직하다. 다중 공간 질의는 공간 영역 질의의 집합으로 볼 수 있는데, 영역 질의의 처리 시간을 분석해 보면 CPU가 차지하는 비중보다 디스크 입출력이 차지하는 비중이 크다. 따라서 다중 공간 질의를 수행할 때도 디스크 입출력 비용을 줄여야 한다.

다중 공간 질의를 다중 CPU/단일 디스크 구조를 가진 서버 상에서 처리할 때는 질의 처리를 위해 하나의 디스크에 접근하므로, 디스크 병목 현상이 발생해서 질의 처리 성능이 크게 저하되는 문제점이 있다. 공간 데이타의 양이 방대할 경우 병목 현상이 더욱 심화되므로, 다중 공간 질의를 처리하려면 단일 디스크 구조보다 다중 디스크 구조를 채용하는 것이 효율적이다. 다중 디스크 상에서의 공간 데이터 배치 기법인 디클러스터링(declustering)은 분산 저장한 공간 데이타에 대한 병렬 디스크 입출력을 가능하게 해 주므로 디스크 병목 현상을 해소할 수 있는 장점이 있다.

이 논문에서 사용하는 고정 그리드 공간 색인을 기반으로 하는 디클러스터링 기법으로 CMD, Z-Order, HCAM 등을 들 수 있는데[1], 일반적으로 CMD 기법과 HCAM 기법이 성능이 안정되어 있고 우수하다고 알려져 있다[2].




(가) 수행되는 영역 질의들 (나) 질의에서 필요한 디스크1의 셀

그림 1. 질의 간 암의 탐색의 예시

본 연구는 1997년도 한국학술진흥재단 대학부설연구소과제 연구비에 의하여 연구되었음

그림 1은 4개의 디스크에 CMD 디클러스터링 기법을 사용해 공간 데이타를 분산 배치한 환경에서의 질의 처리 예시이다. 영역 질의들이 질의 처리 큐(queue)에 들어간 순서대로 처리된다고 가정할 때 그림 1의 (가)와 같이 질의가 들어오면, 디스크 1에서는 질의 Q1을 처리하기 위해 12, 19, 26번 셀을 읽어 들이게 된다. 다음 질의 Q2를 처리하려면 37, 55번 셀을 읽어야 하므로 디스크 임의 탐색(random seek)이 발생하게 되고, 마찬가지로 질의 Q3과 Q4를 처리할 때도 임의 탐색이 발생한다.

그림 1과 같이 다중 디스크 기반 구조에서 디클러스터링 기법을 도입해도, 다수의 질의를 처리할 때 각 질의가 연관성 없이 지역 디스크에 대해서 동시에 수행되므로, 질의 처리 중에 디스크 임의 탐색이 발생하게 된다. 이를 '**질의 간 임의 탐색**'이라고 정의한다. 디스크 입출력 시간 중 디스크 탐색 시간이 가장 크므로, 질의 간 임의 탐색이 발생하면 디스크 입출력 시간이 증가해 질의 처리 속도가 저하된다. 또한 질의 Q1과 Q4처럼 유사하거나 같은 지역을 처리해야 하는 경우는 디스크 캐쉬를 효율적으로 이용할 수 없고, 디스크 입출력이 중복되므로 질의 처리가 더욱 늦어진다.

질의 처리 단계 이전인 디클러스터링 수행 시에는 동시에 수행되는 영역 질의들 간의 위치나 관련성을 알 수 없는 문제점이 있다. 따라서 다중 공간 질의를 수행할 때 발생하는 질의 간 임의 탐색은 다른 방법으로 해결할 필요가 있는데, 이 논문에서는 3장에서 질의 간 임의 탐색 문제를 해결할 수 있는 방법을 제시한다.

## 3. 다중 공간 질의의 처리 기법

이 장에서는 다중 공간 질의의 특징을 분석하고, 질의 간의 공간 및 시간 관련성을 이용하여 다중 공간 질의를 효율적으로 처리하는 방법을 제안한다.

다중 공간 질의는 다음과 같은 특징을 가지고 있다.

**[특징 1] 전체 질의 대상 영역 중 질의가 집중되는 지역이 발생할 수 있다.** – 예를 들어 도시의 중심가나 상가 밀집 지역처럼, 다수의 사용자가 공통적으로 관심을 가지는 영역이 발생할 수 있다.

**[특징 2] 질의들을 빠른 시간 안에 처리해야 한다.** – 모든 질의는 응용의 특성상 적절한 처리 한계 시간 이내에 수행되어야 한다.

**[특징 3] 이전에 처리했던 질의들의 정보를 질의 처리 중에 유지할 수 있다.** – 질의 처리 도중에 지금까지 수행한 질의의 위치 정보 등을 알 수 있다.

### 3.1. 질의 간 위치 관련성

다중 공간 질의의 [특징 1]을 이용한 '**질의 간 위치 관련성(positional relationship)**'은 [수식 1]과 같이 정의하였다.

$$PR_k = \sum_{i=0}^{n-1} Overlap\_Area(Q_p, Queue[0..(n-1)]) \quad \cdots \quad [수식\ 1]$$

질의 간 위치 관련성은 n개의 영역 질의들이 서로 겹치는 면적의 합으로써, n개의 질의 사이의 겹치는 면적이 클수록 높으며, 겹치지 않거나 하나의 선만 만나는(meet) 경우는 0으로 정한다. 위치 관련성은 새로운 질의가 큐에 들어올 때마다 계개산하며, 질의 간 위치 관련성이 높은 질의들을 질의 처리 우선순위가 높게 설정한다.

질의 간의 위치 관련성이 크다는 말은 해당 질의가 서로 인접한다는 뜻으로, 같은 지역의 객체를 사용할 확률이 높다는 의미이다. 근접한 지역에 대해 집중되는 영역 질의를 같이 처리하면, 객체를 찾을 때 필요한 디스크 탐색의 중복을 줄이면서 질의를 처리할 수 있다. 그림 1을 예로 들면, 질의 Q1과 Q4간의 질의 간 위치 관련성은 다른 질의들에 비해 크고 질의 처리에 필요한 셀들(12, 19, 26번 셀)이 같으므로, Q1을 처리한 후 Q4의 우선순위를 높여 한번에 처리해 줄 수 있다.

### 3.2. 질의 간 시간 관련성

다중 공간 질의는 '공간적 관련성'인 질의 간 위치 관련성 이외에도 [특징 2]를 위해 '시간적인 측면'을 고려해야 한다. 시간적인 측면을 고려하는 이유는 질의를 공평하게 처리하기 위함이다. 즉 공간적 관련성만 고려할 때, 다른 질의와의 위치 관련성이 낮아서 처리 순서가 계속 늦어지는 질의(다른 질의와 공간적으로 멀리 떨어져 있는 질의)가 발생하는 문제점을 해결하는 것이 목적이다.

질의를 처리할 때의 시간 우선순위는 [수식 2]와 같이 정의하였다.

$$TS_k = \sum_{i=0}^{k-1} Q_i \quad \cdots \quad [수식\ 2]$$

$TS_k$는 k번째 질의의 시간 우선 순위로써 0에서 $T_{MAX}$까지의 값을 가질 수 있고, $Q_i$는 질의 대기 시간을 반영하기 위해 큐 안의 질의들에 할당하는 상수이다. 질의가 큐 안에 오래 유지될수록 $TS_k$가 증가하게 되어 $TS_k$를 질의 처리 우선순위에 반영할 수 있다.

### 3.3. 질의 처리 이력

질의 처리 이력(query processing history)이란 질의 처리를 담당한 프로세서가 생성하는 정보로써, 다중 공간 질의의 [특징 3]을 이용한 것이다. 질의 처리 이력은 자신이 처리한 영역 질의의 위치와 자신에 관한 정보로 구성된다. 마스터-슬레이브(Master-Slave) 구조상에서 질의를 처리한다고 가정하면, 슬레이브들은 질의 처리 이력을 질의 처리 결과와 함께 마스터에게 전송하게 되고, 마스터는 슬레이브로부터 질의 처리 이력들을 수집하게 된다. 마스터는 질의 처리 이력을 활용하여 각 슬레이브들에게 이전에 처리했던 지역과 유사한 지역의 질의를 할당할 수 있으므로, 슬레이브는 유사한 지역에 대한 질의를 연속적으로 처리하게 된다. 그러므로 디스크 캐쉬의 적중률이 증가하게 되어, 디스크 입출력 비용을 줄일 수 있는 효과를 기대할 수 있다.

### 3.4. 질의 스케줄링 기법

질의 간 위치 관련성과 질의 간 시간 관련성 및 질의 처리 이력을 고려해서 질의 스케줄링을 수행하면, 질의 큐 내의 영역 질의들의 처리 순서를 재조정해서 유사한 지역에 대한 질의들을 연속적으로 처리할 수 있다 따라서 질의를 처리할 때 읽는 객체의 중복을 최소화할 수 있고, 디스크 캐쉬의 적중률을 높일 수 있는 장점이 있다.

그런데, 질의 스케줄링은 질의 처리 시간을 단축하는 것과 동시에 질의를 공평히 처리해야 하므로 공간적 측면과 시간적 측면을 모두 고려할 필요가 있다. 즉 [수식 2]로 계산한 시간 우선순위에 대해 [수식 1]로 계산한 공간 우선순위의 비율을 조정해야 하는데, 시간 우선순위에 대한 비율을 조정한 공간 우선순위를 [수식 3]으로 정의하였다.

$$SS_k = (\max(PR_k) / T_{MAX}) \times SF \quad \cdots \quad [수식\ 3]$$

[수식 3]은 비율 조정을 위한 상수 $SF$의 시간 우선순위의 임계값 $T_{MAX}$를 이용해서, 위치 관련성($PR_k$)과 시간 우선순위의 비율을 맞춰 조정한 공간 우선순위($SS_k$)를 계산하는 것이다 $SF$는 질의 처리 시에 시간 우선순위와 공간 우선순위의 비중을 고려해서 결정한다.

처리할 질의를 선택하기 위해, 마스터는 이전에 처리한 질의의 큐 안의 질의를 대상으로 질의 간 시간 우선순위와 공간 우선순위를 계산해 그 합을 구한다. 가장 높은 우선순위를 가진 질의를 선택하면, 마스터는 유사한 지역의 질의 처리 이력을 가진 슬레이브에게 해당 질의를 할당한다. 각 슬레이브들은 질의 결과와 함께 질의 처리 이력을 마스터에게 전송한다. 질의 처리 큐는 우선순위 큐(priority queue)를 사용해서, 우선순위가 가장 높은 질의를 먼저 처리한다

시간 우선순위와 공간 우선순위를 모두 고려한 질의 스케줄링 알고리즘은 그림 2와 같다.

```
queue_item get_next_query (query_queue[], prev_query[])
        query_queue[] : 질의 queue(질의 MBR 의 집합)
        prev_query[] : 이전에 수행되었던 query 들
        queue_item  다음에 수행할 질의
        begin procedure
                // spatial_scheduling
                for (queue_item in query_queue[])
                begin
                        for (query in prev_query[])
                        begin
                                area = positional_relationship (query_item, query),
                                sum_area[] += area,
                        end for
                end for
                temporal_scheduling (query in query_queue[]);
                adjust_scale (query_time[], sum_area[]);
                set_priority (sum_area[], queue_length[]),
                queue_item = max_area_query in query_queue[],
                return queue_item;
        end procedure get_next_query()
```

그림 2 질의 스케줄링 알고리즘

그림 2의 질의 스케줄링 알고리즘을 이용해서 질의 간 임의 탐색을 최소화하는 예시는 그림 3과 같다.

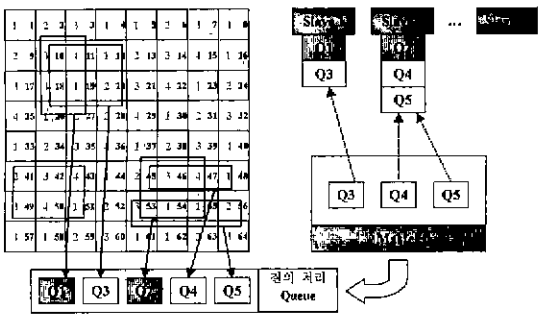



그림 3. 질의 간 임의 탐색의 해결 예시

슬레이브 1이 질의 Q1을 처리한 이력이 있고, 슬레이브 2가 질의 Q2를 처리한 이력이 있다고 할 때, 새로운 질의 Q3, Q4, Q5가 들어왔다고 하자. 마스터는 각 슬레이브의 질의 이력을 조사한 후 처리해야 할 질의들과의 위치 관련성을 계산하여 Q3을 슬레이브 1에 할당하고, Q4와 Q5를 슬레이브 2에 할당하도록 질의 스케줄링을 수행한다. 각 슬레이브들은 인접한 지역들에 대한 질의를 연이어 처리할 수 있으므로 디스크 캐쉬의 적중률이 높아져 질의 처리 속도를 개선할 수 있게 된다.

## 4. 구현 및 성능 평가

이 논문의 구현을 위해서 다중 디스크/다중 CPU 구조를 가진 Parsytec사의 CC16 병렬 컴퓨터를 사용하였다. CC16은 비공유 메모리를 가진 16개의 CPU로 구성되어 있고, 4개의 디스크를 가지고 있다 프로그램은 CC16에서 동작하는 병렬 프로그래밍 라이브러리인 EPX 라이브러리와 C 언어를 사용하여 작성하였다. 질의 처리 구조는 마스터-슬레이브 구조를 사용했다.

성능평가에 사용한 공간 데이타는 Sequoia 2000 벤치마크 데이타[4] 중 37개 계층으로 구성된 다각형 데이타로써, 원래 데이타를 변경한 3~100개 사이의 점으로 표현되는 65,000여 개의 다각형 객체들로 이루어져 있다

전체 질의 영역에 대해 각각 1-2%(Small), 2-4%(Medium), 4-8%(Large)의 크기를 차지하는 질의 500개를 임의로 생성한 후 질의를 수행시켰을 때, 질의 스케줄링의 수행 여부에 따른 캐쉬의 적중률은 그림 4와 같다. 질의 처리에 필요한 다른 비용들-CPU 비용 및 메시지 전송 비용-은 기의 비슷하므로 측정에서 제외하였다. 그리고 실제 운영체제 내부의 디스크 캐쉬의 상태를 직접 조사할 수 없어서, 메모리상에 LRU 방식의 객체 캐쉬를 구현한 후 객체 캐쉬의 적중률을 조사하였으며, 캐쉬의 크기는 객체 100개로 지정하였다.

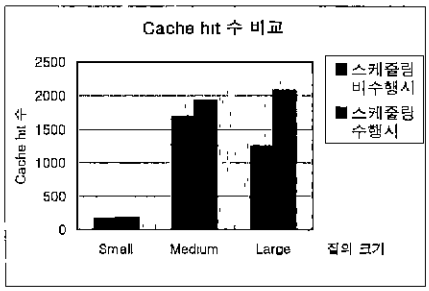



그림 4. 질의 스케줄링 유무에 따른 캐쉬 적중률

실험 결과에서 볼 수 있듯이, 질의 스케줄링을 수행하게 되면 유사한 질의 지역 내의 객체들에 대해 연속적으로 접근할 수 있으므로 디스크 캐쉬의 적중률이 증가하게 되고, 불필요하게 중복되는 디스크 입출력을 방지할 수 있으므로 질의 처리 속도가 개선된다는 것을 알 수 있다.

## 5. 결론 및 향후 연구

이 논문에서는 디클러스터링된 공간 데이타베이스에서의 다중 공간 질의의 특징 및 문제점을 분석한 후 해결책을 제시하였다. 다중 공간 질의는 동시에 수행되는 2개 이상의 공간 영역 질의로, 질의 처리 시간의 대부분을 차지하는 디스크 입출력 비용을 줄이기 위해서 다중 디스크 구조에서 디클러스터링을 수행해도 질의 간 임의 탐색이 발생하게 된다. 질의 간 임의 탐색은 영역 질의가 동시에 서로 다른 위치에 수행될 때 질의를 처리하는 도중 발생하는 디스크 임의 탐색으로, 질의 처리 시간이 증가하는 문제점이 있다

이를 해결하기 위해 질의 간의 위치 및 시간 관련성과 질의 처리 이력을 이용한 질의 스케줄링 기법을 도입하여, 중복되는 객체에 대한 디스크 입출력을 줄이고 디스크 캐쉬의 적중률을 높임으로써 질의 간 임의 탐색을 최소화하였다. 구현 후 실험을 통한 성능 평가 결과, 캐쉬의 적중률이 증가해서 질의 처리 성능이 개선되는 것으로 나타났다.

향후 연구로는 디스크 입출력 비용을 더욱 감소시킬 수 있는 캐쉬 알고리즘과 다중 공간 질의의 특징을 반영할 수 있는 개선된 디클러스터링 기법의 연구를 수행할 예정이다.

## 참고문헌

[1] "Disk Allocation Methods for Parallelizing Grid Files", M. Coyle, S. Shekhar, Y Zhou, Int'l Conf. on Data Engineering, pp.243-252, 1994.

[2] "Scalability Analysis of Declustering Methods for Multidimensional Range Queries", B. Moon, J. H. Saltz, IEEE TKDE Vol 10, No 2, pp 310-327, 1998

[3] "다중 공간 질의 처리를 위한 병렬 공간 객체 파일 서버의 설계", 박영민, 서영덕, 전봉기, 홍봉희, 한국정보과학회 '99 봄 학술 발표 논문집, 제 26권 1호, pp.134-136, 1999.

[4] "The Sequoia 2000 Benchmark", M. Stonebraker, J. Frew, K. Gardels, J. Meredith, ACM SIGMOD Conference, pp.2-11, 1993.