이동체 데이터베이스에서 시공간 근접성을 고려한 디클러스터링 정책[1]

홍 은석○, 서 영덕, 홍 봉희
부산대학교 컴퓨터공학과
{eshong, ydseo, bhhong}@pusan.ac.kr

Declustering Policies Using Spatial-Temporal Proximity in Moving Objects DataBases

Eunseok Hong○, Youngduk Seo, Bonghee Hong
Dept. of Computer Engineering, Pusan National University

요 약

이동체 데이터베이스에서 이동체 궤적의 양은 엄청나게 많아서 기존의 단일 디스크 기반에서는 특정 영역의 질의에 대한 빠른 응답과 처리율의 향상을 볼수 없다. 따라서 고성능 질의 처리를 위한 시스템의 성능 향상을 위해서는 병렬 처리 기법의 도입이 필요하다. 이런 병렬 처리 기법 중, 기존의 디클러스터링 방법에서는 시간이 지남에 따라 연속적으로 보고되는 이동체 특성을 고려하지 않고 있다. 그러므로 대용량 이동체 데이터에 대하여 고성능 질의 처리를 위한 새로운 디클러스터링 방법이 필요하다.

이 논문에서는 대용량 이동체 데이터베이스에 대한 고성능 질의 처리를 위한 새로운 디클러스터링 정책을 제시하였다. 이동체 데이터의 MBB중 공간 좌표의 근접성만을 고려하여 하나의 SemiAllocation Disk 값을 설정하고 그 값과 시간 도메인을 다시 고려하여 근접성을 계산함으로써 디클러스터링을 할 수 있다. 또한 디스크별 Load Balancing을 고려하여 보다 정확한 디클러스터링 효과를 가지도록 하였다. 이와 같이 이동체의 특성을 고려한 새로운 디클러스터링 정책으로 시스템의 성능을 향상 시킬 수 있다.

1. 서론

이동체 데이터는 시간이 지남에 따라 연속적으로 보고되는 특징이 있기 때문에 그 양은 엄청나게 많다. 따라서 이동체 데이터를 처리할 대용량 이동체 데이터베이스 시스템이 필요하게 되었다. 기존의 단일 디스크 기반의 시스템에서는 이러한 방대한 양의 데이터를 처리함에 있어 한계를 보일수 밖에 없었고 새로운 병렬 처리의 방법이 필요하게 되었다. 이러한 문제를 해결하는 병렬 처리 방법중에 하나가 다중 디스크 기반의 디클러스터링 방법이다. 엄청나게 많은 양의 이동체 데이터를 적절히 디클러스터링 함으로써 새로운 환경의 질의 처리에 빠른 응답시간과 결과를 얻을 수 있는 것이다.

기존의 이동체 데이터에 대한 디클러스터링 방법은 round-robin이나 공간 근접성을 고려한 방법등이 있다. 이러한 방법들은 이동체 데이터의 시간, 공간 도메인 모두를 고려하지 않는다. 그래서 적절한 데이터의 디클러스터링 처리가 이루어지지 않는 문제점이 있다.

이 논문에서는 이동체 데이터에 대한 영역 질의시 빠른 응답과 시스템 향상을 위해 이동체 근접성을 이용한 디클러스터링 방법을 제시한다. 제시된 방법은 이동체 데이터의 공간좌표의 근접성을 고려하여 나온 결과에 시간 도메인을 다시 고려하여 디클러스터링 하는 방법이다. 이 방법을 이용함으로써 이동체 데이터의 시공간 모두를 고려한 디클러스터링을 할 수 있다.

이 논문의 구성은 다음과 같다. 먼저 2장에서는 관련 연구를 소개하고 3장에서는 디클러스터링의 기존 방법 적용시의 문제를 기술한다. 4장에서는 이동체 데이터의 디클러스터링 정책 몇가지를 제시하고 5장에서는 결론 및 향후 연구를 기술한다.

2. 관련 연구

이 논문에서는 기존의 디클러스터링 방법들 중에서 관계형 데이터베이스가 아닌 공간 데이터베이스를 대상으로 한다. 공간 데이터베이스에서 가장 heuristic한 디클러스터링 방법으로는 round-robin 방법이 있다. 이 방법에서는 삽입되는 데이터의 상호 연관성과는 상관없이 보고되는 차례대로 각 디스크로 할당하는 방법이다. 그리고 수학적 heuristic 한 방법으로는 근접성(Proximity)[1] 정책을 사용하는 Parallel R-tree[1] 방법과 시간에 대한 값만을 이용하는 T-Proximity[2]이 있다.

[1] 본 연구는 한국과학재단 2003년도 지역대학우수과학자 지원연구(R05-2003-000-10360-0)의 지원으로 수행되었음

[1]에서는 기존의 객체와 새로 삽입되는 객체간의 근접성(Proximity)를 조사하여 낮은 근접성을 가지는 객체들을 서로 같은 디스크에 저장하는 방법이다. 이것은 특정 영역의 질의가 들어올 때 동시에 검색되는 객체를 서로 다른 디스크에 둠으로써 디클러스터링의 기본요소인 MinLoad와 UniSpread를 만족하고자 하는 방법이다. [2]의 방법에서는 보고되는 이동체의 시간 도메인 사이의 근접성을 고려하여 디클러스터링 하는 방법이다.

이러한 기존의 방법을 이동체 데이터에 대하여 그대로 적용하면 시간 도메인에 대한 고려가 없거나 시간만을 적용하기 때문에 정확한 디클러스터링이 이루어 지지 않는 문제점이 있다.

3. 문제 정의

이동체 데이터에 대해 기존의 Proximity[1] 방법을 적용하면 이동체의 특성인 시간 도메인을 고려하지 않는 문제점이 있다. 다차원에 대한 논의는 있지만 시간을 하나의 도메인으로 보지 않고 있다. 기존의 이러한 방법 적용은 이동체 데이터에 대하여 적절한 디클러스터링이 이루어 지지 않는다.

이동 객체의 3차원 MBB 데이터 중에서 공간 좌표에 공간 근접성을 적용하는 것을 다음과 같이 정의한다.

정의 1. Spatial Proximity (SP) : 보고 되는 이동체 데이터의 공간 좌표와 기존의 공간 데이터 사이의 근접성

이동체 데이터는 시간, 공간 모두를 고려한 디클러스터링이 이루어져야 한다. 그러나 기존의 방법대로 시간 도메인에 대한 고려가 없거나 공간 도메인에 대한 고려가 없는 디클러스팅 방법으로는 한 디스크에 데이터가 집중되는 등의 시공간 데이터에 대한 적절한 분산효과를 볼 수 없다. 이로 인하여 특정 영역에 대한 질의가 수행될 때 하나의 디스크에만 접근하는 Load Balancing 문제가 발생할 수 있는것이다.

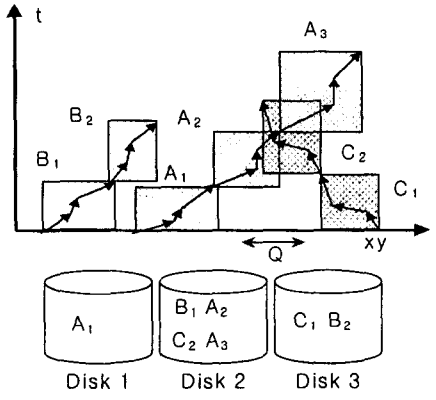

그림 1. 시공간 디클러스터링

예를 들어, 그림 1과 같이 이동체 데이터가 보고 된다고 가정하자. 만약 A_3 데이터가 마지막으로 보고 된다면 이 데이터는 자신의 형제 노드들과의 공간 근접성인 SP를 검사하게 된다. 이때 A_3는 B_1객체의 데이터와 가장 낮은 근접성을 보인다. 즉 A_3 데이터는 B_1 데이터와 가장 멀리 떨어져 있으며 Q와 같은 특정 영역의 질의에 대하여 동시에 읽혀질 가능성이 낮다는 것이다. 그러므로 A_3는 B_1이 저장되어 있는 디스크2로 저장될것이다. 이와 같은 방법으로 모든 데이터를 저장하여 그림 1과 같이 저장되어 있다고 가정한다.

이때 Q와 같은 특정 영역에 대해 질의가 들어 온다면 읽혀질 데이터는 A_2, C_2, A_3가 된다. 그러나 이 데이터들은 하나의 디스크에 있으므로 세번의 디스크 I/O가 발생한다. 이러한 결과는 디클러스터링의 MinLoad, UniSpread 목적에 맞지 않는 결과로 전체 시스템의 늦은 응답과 낮은 처리율을 보이게 된다. 그러므로 시간 도메인에 대한 고려없이 기존의 방법을 그대로 적용하는것은 문제점을 안고 있다.

4. 이동체 데이터의 디클러스터링

이 장에서는 앞장에서 보인 문제점에 대하여 어떻게 효과적인 이동체 디클러스터링을 할 것인지 설명한다.

4.1 SemiAllocation Disk

이동체 데이터 중에서 객체간의 공간 근접성만을 고려하여 나온 가상의 디스크를 다음과 같이 정의 한다.

정의 2. SemiAllocation Disk (SD) : 이동체 데이터의 MBR에 대하여 Spatial Proximity 방법을 적용하여 설정된 가상 디스크

즉 SD는 이동체 데이터의 공간 좌표의 근접성만 고려된 가상의 디스크이다. 이때의 SD값을 시간 도메인과 함께 하나의 도메인 축으로 생각하고 시간과 SD로 근접성을 다시 계산한다.

4.2 SD축 선정 정책

앞 절에서 정의한 SD는 공간좌표의 근접성만을 고려하였기 때문에 불균등하게 디클러스터링 되는 문제점이 발생할 수 있다. 그러한 문제점을 미리 해결하기 위해 각 디스크에 저장될 데이터의 양으로 SD축을 선정한다.

그림 2는 데이터 분포에 따른 SD축의 변화를 보여주고 있다. 하나의 디스크를 기준으로 다른 디스크의 데이터양을 표시하고 그에 맞추어 SD축을 동적으로 구성하는것이다.

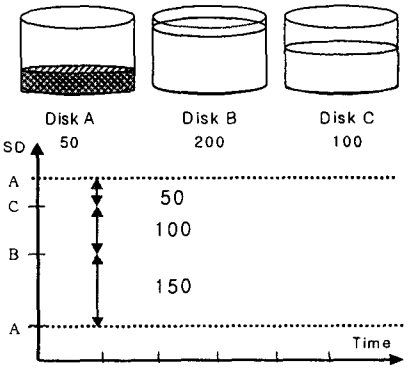

그림 2. 데이터 분포에 따른 Load

예를들어 그림 2를 보면 Disk C를 기준으로 Disk B의 데이터 양이 200만큼이 있다. 이때 Disk C와 Disk B의 데이터 차이는 100이 되며 이 값이 SD축을 형성할때 Disk A 와 Disk B의 간격을 결정하게 되는 것이다. 이렇게 함으로써 특정 디스크에 데이터의 집중을 막고 효과적인 디클러스터링이 되도록 설정 하는것이다.

4.3 SDT-Proximity

앞 절에서 정의한 SDT 좌표에서 근접성을 적용할 이동체 대상을 Point로 할 것인가 Line으로 할 것인가에 따라 다음과 같이 다른 접근방법이 있다.

4.3.1 Point

SDT 좌표에서 근접성을 비교할 대상을 하나의 Point로 본다면 그림 3과 같이 표현된다. 검사할 데이터는 그 이동 객체의 이전의 데이터와는 무관하며 SDT 좌표에 보고되는 당시의 데이터만으로 다른 객체들과 근접성을 비교하게 된다.

그림 3에서 처럼 C_2가 마지막 보고된 SD라면 C_2를 기준으로 C_2의 형제노드의 데이터끼리 근접성을 비교하는 방법이다. 이렇게 함으로써 데이터의 삽입 성능에 향상을 볼수 있다.

4.3.2 Line Segment

근접성을 비교할 대상을 하나의 Line으로 본다면 그림 4와 같이 표현된다. 같은 이동 객체에 대하여 바로 전에 보고된 데이터와 Rectangle(R)을 형성하고 같은 방식으로 형성된 다른 이동 객체의 Rectangle과 근접성을 계산하는 방법이다.

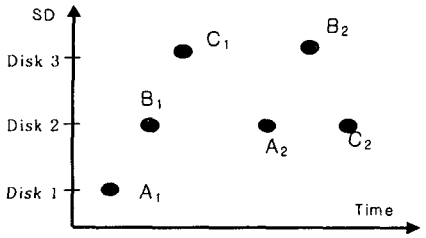

그림 3. Point로 SDT-Proximity 방법 적용

그림 4를 보면 C_2가 마지막에 보고된 SD이고 C_1이 같은 객체의 앞서 보고된 SD이다. C_2가 마지막에 보고되면 C_1과 Rectangle(R)을 형성하여 C_2의 형제노드들의 다른 Rectangle과 근접성을 계산하는 것이다. 이렇게 함으로써 이전 데이터에 대한 관련성이 높아져 더 정확한 디클러스터링을 할 수 있는 장점이 있다.

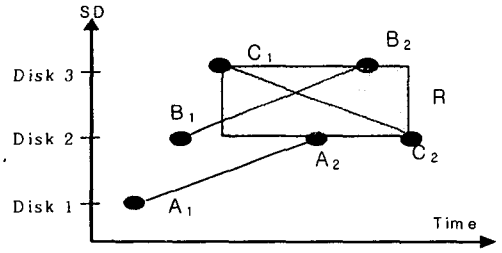

그림 4. Line Segment로 SDT-Proximity 방법 적용

위와 같은 두가지 접근 방법을 통하여 실제로 단말노드가 저장될 디스크를 결정할 수 있다. 여기서 결정되는 디스크는 실제 데이터가 저장되는 곳으로 이동체 데이터에 대하여 시공간 모두를 고려한 디클러스터링 결과가 된다.

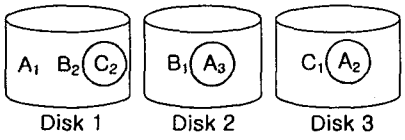

그림 5. SDT-Proximity 적용 후 결과

그림 4의 방법으로 SDT-Proximity를 적용한 결과는 그림 5에서 보여주고 있다. C_2는 디스크 1에 A_2는 디스크 3에 저장된다. 그러므로 2장에서와 같은 영역의 질의가 요구되면 해당 객체 A_2, C_2, A_3 모두 다른 디스크에 있으므로 한번의 디스크 I/O 시간으로 모든 데이터의 검색이 가능하다.

5. 성능 평가

실험은 윈도우 2000 OS와 256MB의 메인 메모리, CPU는 Pentium4 1.7Ghz, HDD 4개의 환경에서 실험 하였다. 데이터는 GSTD[5] 생성기를 통하여 10000개의 데이터를 생성하고 질의는 시공간 축에 5%의 영역질의에 대하여 100번의 질의로 성능 평가 하였다. 페이지 크기는 1024byte로 실험 하였다.

5.1 실험 결과

실험은 디스크의 개수를 2개에서 4개로 늘리면서 round-robin 방법과 공간영역만을 Proximity 하는 Spatial-Proximity 방법, 이 논문에서 제안한 SDT-Proximity 방법을 적용하였다.

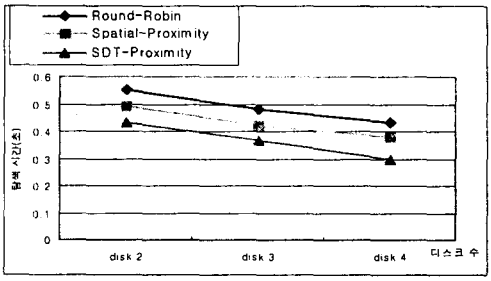

그림 6. 디스크 수에 따른 탐색 시간

그림 6에서 디스크의 수가 4개 일때를 보면 같은 영역 질의에 대하여 round-robin 방법보다 이 논문에서 제안한 SDT-Proximity 방법이 좀더 빠른 성능을 보여주고 있다. 이러한 결과는 SDT-Proximity 정책을 통한 데이터의 분산 방법이 영역 질의에 있어서 heuristic한 방법인 round-robin 보다 더 좋은 성능을 보여주고 있기 때문이다.

6. 결론 및 향후 연구

이 논문에서는 다중 디스크 기반의 서버환경에서 이동체가 자신의 위치를 주기적으로 서버에 보고할때 어떻게 디클러스터링 할 것인가에 대한 새로운 방법을 제시하였다.

3차원 이동체 데이터 중 시간을 제외한 공간 관련성만을 고려하여 근접성이 가장 작은 디스크를 SD로 설정한다. 여기서 SD와 시간 도메인을 다시 고려하여 SDT-Proximity 방법을 적용하고 실제 저장 되어야할 디스크를 결정하는 방법이다. 이렇게 함으로써 특정 영역의 질의시 빠른 응답 시간과 처리율을 향상 시킬 수 있다.

향후 연구로 더 정확한 디클러스터링 정책의 연구와 좀더 많은 데이터와 실험 요소로 다양한 환경에서 실험을 적용해야 할 것이다.

7. 참고 문헌

[1] Ibrahim Kamel , Christos Faloutsos, " Parallel R-Trees" , Proceedings of the 1992 ACM SIGMOD, pp. 195-204, 1992.

[2] Sanjiv Behl , Rakesh M. Verma," Efficient Declustering Techniques for Temporal Access Structures" , Proceedings of the 1991 SIGMOD Conference, pp. 436-445, 1991.

[3] Dieter Pfoser, Christian S. Jensen, Yannis Theodoridis, " Novel Approaches to the Indexing of Moving Object Trajectories" , In Proc, Of the 26th Int' l Conference on VLDB, pp. 395-406, 2000

[4] 홍은석,서영덕,홍봉희, " 이동체 데이터의 근접성을 이용한 시공간 디클러스터링 방법" , 한국정보과학회 2003 봄 학술발표논문집 제30권 1호 pp767

[5] Yannis Theodoridis, Jefferson R. O. Silva, Mario A. Nascimento, " On the Generation of Spatiotemporal Datasets" , In Proxeedings of the 6th Int'l Symposium on Large Spatial Databases 1999