이동체 관리를 위한 다중 처리 시스템의 설계

김진덕+ · 강구안+ · 육정수++ · 박연식++
+동의대학교 컴퓨터공학과, +경상대학교 정보통신공학과

A Design of Parallel Processing System for Management of Moving Objects

Jin-Deog Kim+ · Gu-An Kang+ · Jung-Soo Youk++, Yeoun-Sik Park++
+Dongeui University, ++Gyeongsang National University
E-mail : jdk@deu.ac.kr

요 약

자동차, 모바일 폰, PDA와 같은 이동 객체를 모바일 데이터베이스에 정확히 색인하기 위해서는 위치 정보를 계속적으로 변경해주어야 하며 이는 많은 시간이 소요된다. 기존의 공간 색인에 관한 연구는 효율적인 검색 방법을 제시하였지만 이동체 데이터베이스에서는 질의처리의 효율성보다 이동 객체의 위치정보를 빨리 획득하고, 최신 위치를 저장하는 것이 더 중요하다. 그러므로 가능한 한 정확한 현재 위치 정보를 제공해야 하는 이동체 데이터베이스를 위해서는 병렬 처리 시스템의 도입이 필요하다.

이 논문에서는 다중 처리기를 이용하여 모바일 객체를 공간 색인하는 시스템을 제안한다. 구체적으로 데이터베이스의 변경 연산을 최소화하기 위해 이동객체의 특성을 이용한 버켓 분할 기법을 제안한다. 그리고 처리기 간의 메시지 전송량을 줄이기 위한 데이터 획득 방법 및 버켓 경계 정보 전송 방법을 제안한다.

ABSTRACT

In order to index exactly moving objects(vehicle, mobile phone, PDA, etc.) in the mobile database, continuous updates of their locations are inevitable as well as time-consuming. The studies of pure spatial indices have focused on the efficient retrievals. However, the acquisition and management of the terminal location of moving objects are more important than the efficiency of the query processing in the moving object databases. Therefore, it will be need to adopt parallel processing system for the moving object databases which should maintain the object's current location as precise as possible.

This paper proposes a architecture of spatial indexing mobile objects using multiple processors. More precisely, we newly propose a method of splitting buckets using the properties of moving objects in order to minimize the number of database updates. We also propose a acquisition method for gathering the location information of moving objects and passing the information of the bucket extents in order to reduce the amount of passed messages between processors.

키워드
이동 객체, 다중 처리, 메시지 전송, 선택적 색인 변경

Ⅰ. 서 론

이동 객체는 대용량이면서 동시에 잦은 위치 정보 변경이 발생한다. 그렇지만 이동 객체의 위치 이동이 있을 때마다 공간 색인의 변경에 그대로 변경되어서는 안 되며, 아울러 가장 최근의 위치 정보를 보관해야 한다는 두 가지 조건을 모두 만족하여야 한다[1,8]. 이를 위해서는 위치 데이터의 갱신 연산을 효율적으로 수행하는 다차원 공간 색인 기술이 매우 중요하다.

그러나 지금까지의 공간 색인에 관한 연구는 주

로 정적인 객체에 대한 색인 기법을 제시하고 빠른 검색을 지원하기 위한 연구에 치중되어 왔다. 최근 이동 객체를 위한 공간 색인을 제시되고 있지만 다중 처리기를 이용하여 절대적인 처리 시간을 줄이고자 하는 연구는 거의 없다.

이 논문에서는 가능한 최근의 위치 정보를 제공하기 위해 다중 처리기를 이용하여 모바일 객체를 공간 색인하는 시스템을 제안한다. 구체적으로 이동 객체의 위치 이동에 따른 데이터베이스의 변경 연산을 최소화하기 위해 이동객체의 특성을 이용한 버켓 분할 기법을 제안한다. 이는 이동체의 움직임 특성을 파악하여 위치변경에도 불구하고 색인 재구성이 불필요한 경우에는 변경 연산을 수행하는 않는 방법을 택하는 것이다. 이를 위해 버켓 모양 변형과 같은 방법을 이용한다. 그리고 병렬 처리 환경에서 처리기 간의 메시지 전송량을 줄이기 위한 데이터 획득 방법을 제안하고 각각의 종 프로세서에 이동 객체를 부 그룹으로 할당하는 기준을 제시한다.

이 논문의 연구는 최근 활성화되고 있는 LBS에서 연산의 횟수를 줄여 빠르고도 가장 최근의 위치 정보를 획득하고 관리할 수 있는 기반 기술이 되며, 이를 바탕으로 과거의 궤적 및 과거 영역 질의보다는 현재의 위치를 중시하는 맞춤형 광고, PUSH&PULL형 LBS 및 텔레매틱스 응용에 사용될 수 있을 것으로 판단된다.

이 논문의 구성은 다음과 같다. 제 2장에서는 이동 객체의 공간 색인 및 데이터 획득에 관한 관련 연구에 대해 알아보고, 제 3장에서는 이동체의 관리를 위한 병렬 처리 시스템의 구조 및 이동체의 특성을 고려한 새로운 공간 색인 기법을 제안하며, 각 종 처리기에서의 데이터 획득 방법 및 버켓 관리 기법을 설명한다. 그리고 제 4장에서 결론을 맺는다.

II. 관련 연구

지금까지 이동 객체와 관련된 연구에는 효과적인 시공간 데이터의 저장, 데이터의 모델링 및 빠른 접근을 위한 시공간 색인 등에 대하여 집중적으로 연구되어져 왔으며, 위치 정보의 획득에 관한 연구는 상대적으로 부족하다. 앞으로는 대용량의 이동체의 위치정보를 획득할 때 최소의 시간과 통신 부하를 기록하는 방법론이 필요하다. 지금까지 주로 연구된 이동체를 위한 공간 색인에 관한 연구는 다음과 같다.

위치 정보를 수식으로 표현하는 방법[2], 이동 객체의 과거 이력 정보를 검색하기 위한 궤적으로 표현하는 방법[3,4,5], 미래의 위치를 표현하고 검색하기 위한 방법[6], 데이터베이스 변경횟수를 줄이기 위한 방법[7,9] 등으로 나누어진다.

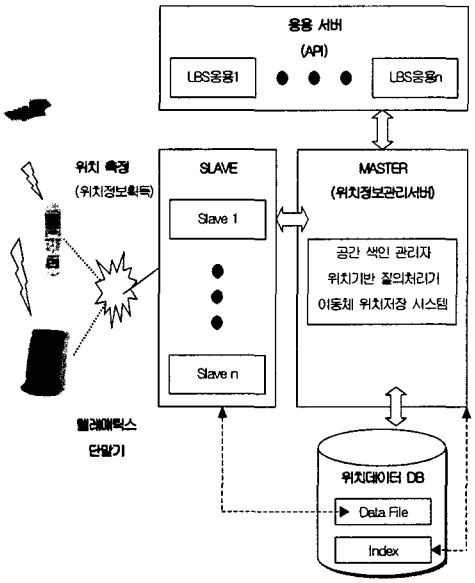
관련 연구 [6,10]은 이동체를 시간에 대한 선형 함수로 표현하는 색인구조인 TPR-Tree를 제시하였다. 관련연구 [7]은 R-tree에서의 빈번한 변경연산을 Bottom-up 방식으로 전환하였으며, 지역화된 연산 및 동시성을 제공한다.

관련연구 [9]는 이동객체의 데이터베이스 변경 횟수를 줄이기 위한 방안을 제시하였지만객체의 움직임을 단위 사각형으로 제안하여 보다 많은 비교 횟수가 요구되며, 다중 처리의 구체적인 방안이 제시되지 않았다.

III. 이동체의 관리를 위한 병렬 시스템

3.1 다중 처리기 기반 시스템 구조

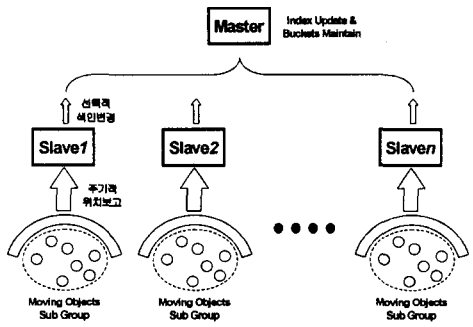
그림 1은 이 논문에서 제안하는 다중 처리기 기반 위치 정보 관리 시스템을 도식화한 것이다. 이동객체는 대규모이며, 위치 정보가 동적으로 변하며 그 변화폭 또한 매우 크기 때문에 본 논문에서는 다중 처리기를 도입하여 여러 개의 종(Slave) 프로세서가 각 이동 객체의 움직임을 주시하며 변화된 값을 관리한다.



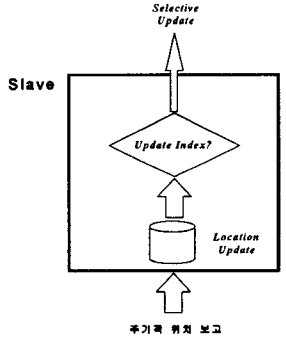
<그림 1> 다중 처리기 기반 위치 관리 시스템

그림 2는 위치 측정부를 보다 자세히 살펴본 것으로서 각 종 프로세서마다 관리하는 이동객체 그룹이 주기적인 위치 보고를 할 때마다 위치 데이터 DB의 데이터 파일 내에서 현재 위치를 기록하고 난 뒤, 그림 3과 같이 주 프로세서에서 일괄적으로 관리하는 공간 색인에 대한 변경 작업을 수행할지 여부를 결정하여 선택적인 색인 변경작업을 수행하도록 한다. 이러한 선택적인 색인 변경은 다수의 이동객체를 보다 빨리 색인화하는 중요한 기법이

다. 이 논문에서는 이동된 후의 객체 위치가 현재 버켓의 경계선을 벗어나지 않으면 공간 색인의 포인터 값을 변경시키지 않는다.



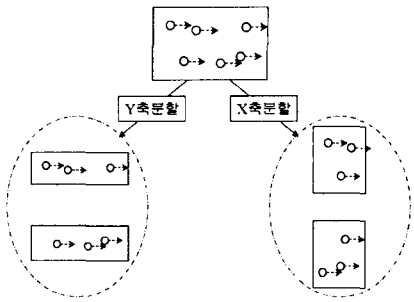
<그림 2> 종 프로세서의 위치 획득



<그림 3> 선택적인 색인 변경

3.2 이동체 특성을 이용한 버켓분할

이 논문에서 객체들을 두 개의 버켓으로 분산시킬 때 변경 연산 횟수를 최소화하기 위해 X축과 Y축을 선택할 수 있다. 기존의 색인에서는 분할된 버켓이 정사각형에 가깝도록 X축으로 분할하거나 X와 Y축을 번갈아 가며 분할한다. 예를 들어 그림 4에서 하나의 버켓을 X와 Y축을 기준으로 각각 분할할 수 있다. 그러나 이동체는 항상 움직인다는 특성을 감안하면 그림 4의 예에서는 Y축을 기준으로 분할하는 것이 변경연산을 최소화할 수 있다. 왜냐하면 버켓내의 객체들은 X축으로 따라 이동하기 때문에 Y축을 기준으로 분할하면, 이동체의 다음 위치 보고 시간에 해당 버켓 내에 존재할 확률이 높아 전술한 바와 같이 변경연산을 생략할 수 있기 때문이다.



<그림 4> 버켓 분할

이 논문에서는 분할하고자 하는 축을 선택할 때 우선 순위를 객체의 움직임 방향성, 인프라의 방향성, 움직임 통계정보 등을 차례대로 적용한다.

단, 분할될 두 버켓 중 한 버켓에 극단적으로 많은 버켓들이 배분될 경우 두 버켓에 골고루 배분될 수 있는 축으로 재설정한다. 이 논문에서는 한 버켓에 80%이상 치우치면 극단적 배분이라 가정한다. 분할한 후에도 객체의 수가 버켓의 용량을 초과하면 분할 알고리즘을 재귀적으로 호출한다.

3.3 프로세서간의 버켓 경계 정보 통신

대용량의 이동 객체에 대한 위치 획득은 가능한 위치 획득 횟수를 줄여서 통신 부하를 최소화하는 것이 매우 중요하다. 전술한 바와 같이 이 논문에서는 이동된 후의 객체 위치가 버켓의 경계를 벗어나지 않으면 현재의 위치를 데이터 파일에만 저장하고 공간 색인의 변경을 생략할 수 있는 방법이다. 그러므로 종 프로세서 내에서 공간 색인의 변경 여부를 결정하기 위해서는 해당 프로세서에서 버켓의 경계 정보를 보관하고 있어야 함을 의미한다.

그러나 이동체 데이터베이스의 특성상 잦은 위치 변경은 버켓 경계 정보 또한 이에 비례하여 빈번하게 변동된다. 그리고 대규모의 이동체이므로 전체 버켓의 경계 정보도 매우 크다. 그러므로 버켓 경계 정보가 변할 때마다 모든 버켓 정보를 각 종 프로세서에 전달하면 매우 많은 통신 부하를 보이게 된다.

따라서 이 논문에서는 통신의 부하를 최소화하기 위해 각 종 프로세서가 초기에 기존 인프라의 이동체 분포를 감안하여 버켓의 경계 정보를 배포한다. 그 뒤, 각각의 이동 객체의 움직임에 따라 버켓의 변경정보가 변경되었을 때 다음과 같은 규칙에 의해 버켓 정보가 이전된다.

우선, 버켓의 합병이 일어났을 경우에는 버켓 정보를 배포하지 않는다. 이는 각 종 프로세서에서는 버켓의 경계 정보가 부정확하여 선택적 색인 변경을 위한 비교 검색 횟수가 조금 증가할 뿐 전체적인 통신 비용을 줄일 수 있기 때문이다. 반면, 버켓 분할이 일어날 경우 즉시 각 프로세서에 전체 버켓 정보가 아닌 분할 정보만 전송하게 된다. 이 때 전송되는 데이터의 포맷은 다음과 같다.

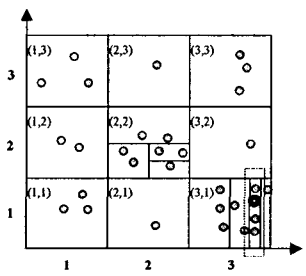
<버켓 주소, 분할축, 버켓위치정보>

이 때 분할축과 위치정보는 비트로 정보를 표현한다. 분할축이 X일 때 '0', Y일 때 '1'의 값을 갖는다. 그리고 버켓 위치 정보는 최근 분할된 버켓의 정확한 분할 위치를 언급한 것으로서 비트단위별로 각각 공간 색인의 버켓 위치를 나타내는 것으로 '0'은 왼쪽 또는 아래쪽을 의미하고, '1'은 오른쪽 또는 위쪽을 의미한다. 이 논문에서는 버켓 위치 정보를 위해 2바이트를 사용하며, 이는 동적 해쉉으로 인한 버켓 분할을 레벨 16까지 허용함을 의미한다.

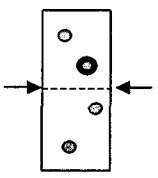
그림 5는 이동체를 위한 2차원 동적 해쉉 파일 구조로서 이와 같은 버켓 분할의 예를 보여 주고 있다. Nx와 Ny가 각각 3이며 버켓 용량은 3이다. 그림 5에서 (2,2), (3,1) 버켓은 이동객체가 밀집되어 있으므로 추가적인 분할이 발생하였음을 보여준다. 그런데 (2,2)와 (3,1)버켓의 분할은 제각기 다른 방식을 택하고 있다. (2,2) 버켓은 최초 Y축으로, 그 뒤에는 X축, Y축으로 연속적인 분할이 되었지만, (3,1) 버켓은 X축으로 세 번 연속 분할되었다. 그림 6은 이동체의 위치 정보 변경으로 버켓 분할이 발생한 경우를 나타낸 것이며, 이 때 버켓 분할로 주 프로세서가 각 종 프로세서에 해당 버켓이 분할 되었음을 알리는 메시지이다.

<9, bit(1), bit(110xxxx xxxxxxxx) >

위의 메시지에서 9는 버켓의 주소이며, 그 다음 '1'은 분할 축이 Y축을 나타내고, 그 다음 1은 현재 분할 기준에서 오른쪽 하위 노드로, 그 다음 1 또한 또다시 오른쪽 하위 노드로, 그 다음 0은 왼쪽 하위 노드로 내려가서 해당 버켓이 Y축으로 분할되었음을 의미한다. 그 다음의 비트값들은 더 이상 의미가 없다. 왜냐하면 현재 종 프로세서가 저장하고 있는 버켓의 경계 정보에 더 이상의 하위 노드가 없기 때문이다.



<그림 5> 버켓의 경계



<그림 6> 버켓의 분할

그리고 주기적으로 주 프로세서가 갖고 있는 버켓 경계 정보를 종 프로세서에 배포하는 데, 이 때 초기의 인프라를 근거로 한 버켓 정보와 새롭게 배포된 버켓 정보의 합집합을 구해 각 종 프로세서가 보관하게 된다.

버켓의 경계를 벗어났다고 판단되면 해당 종 프로세서는 주 프로세서에 다음과 같은 정보를 전달하여 공간 색인의 변경을 요청한다.

<Oid, x, y, data pointer>

3.4 종 프로세서의 이동 객체 할당

병렬 처리 환경에서 보다 좋은 성능을 보이기 위해서는 각 종 프로세서의 부하 편차가 적어야 한다. 이를 위해 이 논문에서는 종 프로세서에 할당할 이동객체의 서브 그룹을 형성할 때 다음과 같은 규칙을 따른다.

1) 주어진 영역 내의 이동체들을 유형별로 묶는다. 유형이란 주로 속도에 의해 구분된다. 즉, 사람, 자동차, 기차 등 비슷한 속도를 가지는 것은 유사한 그룹으로 형성된다.
2) 각 그룹내에서 지역별로 다시 소그룹을 형성한다.
3) 소그룹내에서는 라운드 로빈 방식으로 각 종 프로세서에 할당한다.

이렇게 함으로써 각 종 프로세서는 여러 지역의 이동체를 다양한 속도별로 관리하게 된다. 여러 지역의 이동체를 갖게 되면 데이터의 디클러스터링 효과에 의해 영역질의나 점 질의에 좋은 성능을 보인다. 또한, 관리하는 이동체의 속도가 다양하면 선택적 공간 색인의 변경 연산 횟수가 특정 프로세서에 편중 되지 않은 전체적인 부하 평준화를 이룰 수 있다.

IV. 결 론

이 논문에서는 대규모 이동 객체의 위치 정보를 효과적으로 관리하기 위해 다중 처리 시스템을 이용하는 방법을 기술하였다.

구체적으로 이동체의 위치 획득시 통신 비용을 최소화하기 위해 공간 색인의 버켓 경계 정보를 각 종 프로세서에 효과적으로 전달하기 위한 방법을 제시하였으며, 이동체의 현재 속도 및 방향이 계속된다는 특성과 이동체는 도로망 등의 인프라를 따라 움직인다는 특성을 반영하는 버켓 분할로 버켓 분할을 최소화하는 기법을 제시하였다.

또한 이를 바탕으로 각 종 프로세서에서 독자적으로 공간 색인의 선택적인 변경이 가능하도록 하여 주 프로세서의 부하를 분산시키는 방법 및 시스템 구조도를 제안하였다.

앞으로는 수많은 이동체가 짧은 주기마다 위치 정보를 보고하는 이동체 데이터베이스에서의 색인 속도 향상을 위해서 다중 프로세서 환경에서 각 이동체의 위치 정보를 시간과 방향 및 속도로 표현할 경우의 부하 평준화 방안과 각 이동체의 움직임 속도가 다양할 경우의 부하 평준화 방법에 대해 연구하고자 한다.

참고 문헌

[1] O. Wolfson, B. Xu, S.Chamberlain, and L. Jiang. □□Moving Objects Databases: Issues and Solutions,□□ Proc. of SSDBM Conf., pp 111-122, 1998

[2] G. Kollios, D. Gunopulos, and V. J. Tsotras. □□On Indexing Mobile Objects,□□ Proc. of PODS Conf. pp. 261-272, 1999

[3] D. Pfoser, C. Jensen, Y. Theodoridis, □□Novel Approaches to the Indexing of Moving Object Trajectories□□, Proc. of VLDB Conf., pp. 395-406, 2000.

[4] Y. Theodoridis, □□Spatio-Temporal Indexing for Large Multimedia Applications□□, Proc. of Multimedia Computing and Systems Conf, pp. 441-448, 1996

[5] M. Nascimento, J. Silva, Y. Theodoridis,□□Evaluation of Access Structures for Discretely Moving Points.□□Proc. of STDBM Conf., pp. 171-188, 1999

[6] S. Saltenis, C. Jensen, S. Leutenegger, M. Lopez,□□Indexing the Positions of Continuously Moving Objects.□□ , Proc. of ACM SIGMOD Conf., pp. 331 - 342, 2000.

[7] M.L Lee, W. Hsu, C.S. Jensen, B. Cui, K.L. Teo, "Supproting Frequent Updates in R-trees : A Bottom-Up Approach", Proc. of VLDB Conf, 2003

[8] O. Wolfson, L. Jiang, A. Sistla, S. Chamberlain, N. Rishe, M. Deng, "Databases for Tracking Mobile Units in Real Time", Proc. of ICDT Conf., pp. 169-186, 1999

[9] Z. Song, N. Roussopoulos, "Hashing Moving Objects", LNCS, pp. 161-172, 2001

[10] Y. Tao, D. Papadias, J. Sun, "The TPR* -tree: An Optimized Spatio-Temporal Access Method for Predictive Queries", Proc. of VLDB Conf. 2003