대용량 이동 객체의 위치 정보 인덱싱

박원순[0] 전세길 나연묵
단국대학교 컴퓨터공학과
{wonsooni[0], sjgeon}@dankook.ac.kr, ymnah@dku.edu

Positional Information Indexing on Large Volume of Moving Objects

Wonsoon Park,[0] Segil Jeon, Yunmook Nah
Dept. of Computer Engineering, Dankook University

요 약

대용량 이동 객체의 경우에 특정 객체의 이동경로를 추적유지 해야할 필요성이 있으며 효과적으로 검색하기 위한 인덱스가 필요하다. 본 논문에서는 이동 객체의 위치 정보를 위한 데이타 구조를 제시하고 저장된 데이타를 효과적으로 처리하기 위한 인덱스 구성방안을 제안한다. 인덱스 구조는 대량의 객체의 동시 이동으로 인한 갱신 오버헤드를 최소화할 수 있도록 한다. 이 시스템은 이동 통신 분야에서 고객 위치 정보를 필요로 하는 다양한 서비스와 항공기 운항 제어 등의 분야에서 활용될 수 있다.

1. 서 론

계속적으로 움직이는 대량의 객체 데이터에 대한 저장 및 검색의 필요성이 항공기 운항 제어, 지능적 교통제어, 이동 통신 시스템 등 많은 응용 분야에서 증가하고 있다.

최근의 이동 통신에서 고객 위치 정보 서비스의 경우에서 보면 계속적으로 변화하는 객체 데이타를 데이타베이스에 유지해야하는 경우가 있다. 예를 들어, 어느 음식점에서 특정 지역으로부터 반경 1km 내에 있는 고객들에게 할인 쿠폰을 핸드폰으로 발송하는 경우 단순히 고객의 현재 위치뿐만 아니라 시간에 따른 고객의 위치 변화에 대한 정보도 필요하게 된다.

시간(temporal)데이타 객체를 위한 인덱스 구조로는 TSB-트리 등이 있으며, 데이타 공간(spatial) 데이타 객체를 위한 인덱스 구조는 KD-트리, KDB-트리, R-트리, 격자화일 등이 제안되었다[5]. 기존의 공간 인덱스 구조의 문제점은 데이타 처리 방법에 있는 것이 아니라 데이타 객체의 이동경로를 보존 유지할 수 없다는 데에 문제가 있다. 기존의 공간 데이타 인덱스 처리 방법들은 데이타 객체들이 같은 경로 안에 속하는지 아닌지를 고려하지 않고 단순히 데이타를 선분(line segment)들의 집합으로 취급한다. 따라서 선분들은 근접성과 같은 공간적인 특성에 따라 그룹화 된다. 기존의 GIS에서의 데이타와 달리 이동하는 객체 데이타의 경로를 저장 및 검색하기 위해서는 시간과 공간의 개념을 모두 고려해야 한다.

시공간상의 이동 객체를 처리하기 위해 기존에 연구된 방법으로는 R-트리의 변형된 형태인 STR-트리(Spatio-Temporal R-트리)가 있으며, R-트리를 이용한 경우 데이타 객체의 이동이 대량인 경우에 많은 갱신 연산을 요구하므로 갱신 연산을 줄이기 위해 해시구조를 사용한 경우도 있다[1, 2].

기존의 시공간 데이타 인덱싱에서 STR-트리인 경우 대량의 데이타가 연속으로 입력되는 경우 많은 갱신 연산으로 성능 저하를 초래할 수 있으며, 해시구조를 사용하는 경우 빠른 검색을 위해 해시된 결과를 별도의 인덱스 구조로 유지 관리해야 할 필요성이 있다.

본 논문에서는 대용량 이동 객체 데이타의 이동경로 저장을 위한 데이타 구조를 제시한다. 또한, 이동 객체 데이타의 인덱싱을 위해 기존의 R-트리 기법에 메모리 캐쉬를 사용하여 갱신 연산을 사용하는 방안과 R-트리 대신 해시 구조를 사용하여 갱신 연산을 줄이고 해시된 결과를 격자화일로 유지하는 인덱스 방안을 제안한다.

제안된 인덱스 구조를 이용해 객체의 저장 및 검색하기 위해 인덱싱 시스템의 구조를 설계 및 사용자 인터페이스를 구현한다.

본 논문의 구성은 다음과 같다. 2장에서는 위치정보 저장을 위한 시스템 구조와 데이타 구조를 소개한다. 3장에서는 이동객체에 대한 인덱싱 기법을 제안하고, 4장에서는 이동객체의 처리를 위한 프로토타입 시스템을 기술한다. 5장에서는 결론을 맺고 향후 연구과제를 제시한다.

2. 이동 객체 데이터베이스 구조

2.1 시스템 구조

이동 객체의 저장 및 검색을 위한 시스템 구조는 그림 1과 같다.

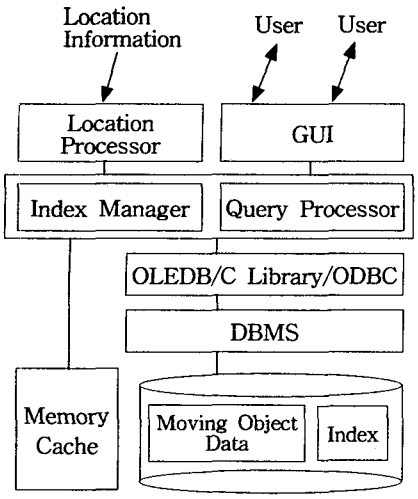

그림. 1 이동 객체 인덱싱 시스템 구조

그림 1에서 위치 처리기는 수집된 위치 정보를 디코딩하여 인덱스 관리자로 전달하는 역할을 한다. 인덱스 관리자는 객체 데이타의 현재 위치에 따라 메모리 캐시에 기록할지 디스크에 저장할지 여부를 결정하고 인덱스 구조를 생성, 갱신한다.

사용자는 GUI 형태의 폼을 이용해 질의 내용을 입력할 수 있으며, 입력된 질의는 질의 처리기를 통해 해석 처리되어 사용자에게 보여진다. 이동 객체 데이타 정보와 인덱스 정보는 데이타베이스내에 저장되며 상용 DBMS에 의해 관리된다.

2.2 데이타 테이블 구조

이동 객체 데이타 정보를 저장하는 데이타베이스 테이블은 현재시간에서의 위치와 시간별 이동객체의 히스토리 정보가 포함된다.

그림 2는 현재위치와 히스토리 정보간의 관계를 표현한 ER다이어그램이다.

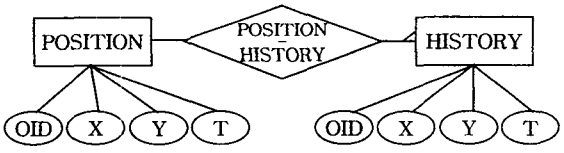

그림. 2 이동 객체 데이타 ER 다이아그램

그림 2에서 속성 OID는 객체의 고유번호 이고 X와 Y는 위치 좌표를 나타낸다. T는 마지막으로 갱신된 시간을 저장한다. 현재의 위치를 나타내는 POSITION은 위치정보에 대해 2차원 인덱스를 생성하고, HISTORY는 위치 좌표와 시간에 대해서 3차원 시공간 인덱스를 생성한다. 이 인덱스는 상용 데이타베이스의 인덱싱 서비스를 이용해 B-트리 형태로 생성된다.

3. 이동객체 인덱스 구조

3.1 순수 R-트리

순수 R-트리만을 이용해 인덱스를 구축하는 경우이며 별도의 추가 프로그램이 필요 없이 R-트리 인덱스 코드만으로 구현 가능하므로 간단히 구현할 수 있다. 시간에 따라 위치가 변하는 이동 객체 데이타가 대량인 경우에는 갱신연산이 많이 일어나 심각한 성능저하가 발생한다.

3.2 메모리 캐쉬 + R-트리

R-트리 만을 이용했을 경우에 발생하는 갱신 연산을 다소 줄이기 위해 특정 영역을 벗어난 경우에 대해서만 트리를 갱신하는 연산을 수행한다. 같은 MBR내의 이동시에는 메모리갱신을 다른 MBR로의 이동시에는 메모리와 디스크 모두에 정보를 갱신한다.

3.3 메모리 캐쉬 + 해시/격자화일

격자 화일은 정적 상황뿐만 아니라 레코드 삽입과 삭제가 빈번히 발생하는 동적 상황에서도 잘 적용되는 화일 구조이다.

시간에 따라 계속적으로 변화하는 객체 데이타의 경로 정보를 인덱싱 하기 위해서 우선 공간상의 위치 정보를 간략한 비트 스트링으로 표현한다. 이러한 좌표 변환의 결과로 만들어지는 비트 스트링은 버켓 번호를 의미하며 특정 해싱함수에 의해 수행된다. 한 객체의 이동이 같은 버켓에 저장된다는 것은 동일 지역 내에서의 이동을 의미한 것이다.

특정 객체의 동일 지역 내 이동으로는 디스크 갱신 연산이 일어나지 않으며 메모리 캐쉬에만 기록된다. 셀을 벗어난 이동에 따른 해당 객체의 변화 정보(현재 객체의 해당 버켓 번호 등)는 해시와 격자 파일을 결합한 인덱스 구조로 유지된다.

또한 과거의 위치정보도 히스토리 로그 형태로 디스크에 저장되어 유지된다.

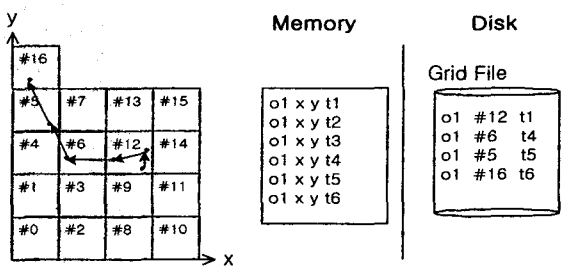

그림. 3 이동 객체 예

그림 3은 객체 o1이 시간 t1에서 t6사이에 움직인 경로와 이때 메모리와 디스크에 기록되는 내용이다. 메모리 상에는 객체의 모든 시간별 좌표가 배열 형태로 저장 되며, 디스크에는 셀 경계를 벗어난 이동이 있는 경우에만 해싱함수를 적용한 후 격자화일 형태로 저장된다.

4. 프로토타입 구현

4.1 구현환경

본 논문에서 제안한 시스템의 구현 환경으로 하드웨어는 Pentium III에 OS는 Windows 2000을 사용한다. 인덱스 구축은 C++로 구현하고 사용자 인터페이스는 Visual C++을 사용한다. DBMS는 MS SQL Server 2000을 이용한다.

4.2 폼 기반 질의 예

그림 4는 사용자가 포인트 질의, 범주 질의, k-최근접 질의를 수행하기 위해 값을 입력하는 폼이다. 이 그림에서는 현재 시간을 기준으로 좌표(5, 6)에서 반경 1Km이내에 존재하는 객체를 질의하고 있다.

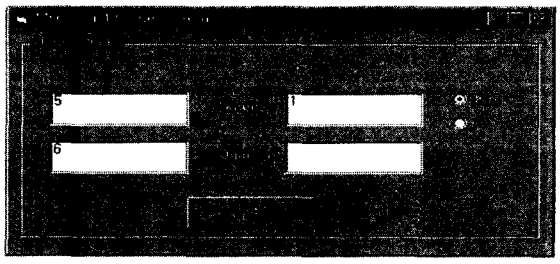

그림. 4 사용자 질의 입력 폼

그림 5는 그림 4에서 입력한 조건에 따른 결과를 보여주고 있다.

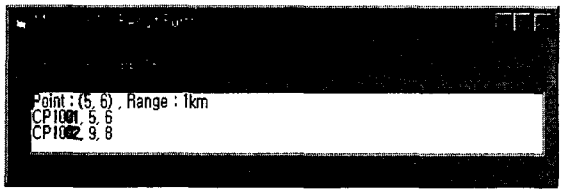

그림. 5 질의 결과 폼

5. 결론

본 논문에서는 계속적으로 변화하는 대용량 이동 객체의 저장 및 검색을 위한 인덱싱 방안을 제안하고 프로토타입 개발을 위한 시스템 구조를 제시하였다.

기존의 공간 데이타베이스나 시간 데이타베이스의 인덱싱에서는 정적인 위치정보나 시간과 연관된 단순 데이타만을 취급하였으므로 계속적으로 변화하는 특정 객체의 경로 정보를 보존할 수 없었다. 이러한 경우 경로 저장을 위한 데이타 구조와 인덱스에 시공간적인 개념이 도입이 필요하다.

본 논문에서는 이동 객체 데이타의 경로 보존을 위한 데이타 구조와 효과적인 검색을 위한 인덱스 방안을 제시하였다. 이동 객체 데이타의 인덱스 방법으로 R-트리에 메모리 캐쉬를 사용하여 갱신 연산을 줄이는 방안과 갱신 연산으로 성능 저하가 심한 R-트리 대신에 해시 구조와 데이타의 동적 변화에 적합한 격자 화일을 사용하는 방식을 제안하였다.

제안된 내용을 구현하기위해 상용 DBMS을 이용해 시스템을 설계하였고 사용자 인터페이스를 구축하였다.

향후 연구 과제로 본 논문에서 제안한 대용량 이동 객체에 대한 대안적 인덱스 방법에 대해서 세부 구축 후 각 방법간의 성능 비교 분석이 필요하다.

참고문헌

[1] Dieter Pfoser, Christian S. Jensen, Yannis Theodoridis, "Novel Approaches in Query Processing for Moving Objects," Proc. of the 26th Int'l Conference on Very Large Databases (VLDB'00), Cairo, Egypt, pp. 395-406, September 2000.

[2] Zhexuan Song 1 and Nick Roussopoulos, "Hashing Moving Objects," MDM 2001, LNCS 1987, pp. 161-172, 2001.

[3] Ouri Wolfson, Bo Xu, Sam Chamberlain, Liqin Jiang, "Moving Objects Databases: Issues and Solutions," Proc. of the 10th Int. Conf. on Scientific and Statistical Database Management (SSDBM98), Capri, Italy, July 1-3, 1998, pp. 111-122.

[4] George Kollios, Dimitrios Gunopulos, Vassilis J. Tsotras, "On Indexing Mobile Objects," Proc. of the 18th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 261-272, May 1999.

[5] Elisa Bertino, Beng Chin Ooi, Ron Sacks-Davis et al., *Indexing Techniques for Advanced Databases Systesms*, Kluwer Academic Publishers, 1997.