제19회 한국정보처리학회 춘계학술발표대회 논문집 제10권 제1호 (2003.5)

모바일 환경에서의 동일한 위치 정보 처리를 위한 이동 객체 색인 구조1)

손봉호*, 이응재*, 류근호*, 이재호**, 박진수***
*충북대학교 데이터베이스연구실
**한국전자통신연구원
***청주대학교 정보통신공학부
e-mail:{bhson, eungjae, khryu}@dblab.chungbuk.ac.kr

Moving Objects Indexing for Processing of Identical Location Information in Mobile Environments

Bong Ho Son*, Eung Jae Lee*, Keun Ho Ryu*, Jae Ho Lee**, Jin Soo Park***
*Database Laboratory, Chungbuk National University
**Electronics and Telecommunications Research Institute
***School of Information and Telecommunication, Chongju University

요 약

최근에는 GPS(Global Positioning System)와 무선 데이터 전송 능력이 있는 휴대용 전화기의 보급 및 이동/무선 컴퓨팅 기술의 발달로 인해 이를 이용한 응용 서비스에 대한 관심이 고조되고 있다. 특히, 이동 차량, PDA, 휴대용 전화기, 노트북 컴퓨터 등을 이용한 위치기반서비스(LBS: Location Based Services)가 무선 인터넷 시장의 중요한 이슈가 되고 있다. 위치 정보를 기반으로 다양한 위치기반서비스를 제공할 수 있는 플랫폼은 향후 무선 인터넷 응용의 핵심기술이므로, 모바일 환경에서는 이러한 이동 객체들의 위치 정보를 효과적으로 관리할 수 있어야 한다. 따라서 이 논문에서는 현재 모바일 환경에서 위치기반서비스 관리 및 제공과정에서 얻어지는 데이터 형태를 살펴보고, 기존의 이동 객체 색인들을 이용하여 실세계에서 얻어지는 데이터를 처리하였을 때의 문제점들을 검토한다. 그리고 이러한 문제를 해결할 수 있는 색인 방법으로 동일한 시공간 정보를 갖는 이동 객체들을 블록단위로 저장하는 색인 구조를 제시함으로써 전체 색인 크기를 줄여 중복검색을 해결하고 디스크 I/O수를 줄이도록 설계한다.

1. 서론

현재 모바일 환경의 위치기반서비스에서 모바일 사용자(즉, 이동 객체[1, 2, 3])에 대한 정보를 다루기 위한 시공간 데이터베이스의 색인 기법들은 기존 공간 색인 기법인 R-tree 계열 색인 구조를 객체의 공간 정보뿐만 아니라 시간의 흐름에 따라 변화하는 특성을 고려하여 변형 및 확장하였다. 현재 모바일 환경의 위치기반서비스의 구성은 그림 1과 같이 모바일 사용자의 위치를 획득하기 위한 무선위치측위기술[4], 획득된 정보를 가공 처리하여 관리하도록 하는 이동 객체 정보 관리 기술, 이동 객체 정보를 이용하여 사용자들에게 제공되는 서비스 기술로 이루어진다. 현재 위치기반서비스에서 가장 기본적이면서도 중요한 능력이 바로 모바일 사용자의 위치 정보 관리 기법이다. 또한, 모바일 사용자의 위치를 얻기 위한 위치측위기술들은 처리 방법상 동일한 MBR정보를 갖는 데이터들을 발생시킨다. 하지만 이러한 데이터들을 기존 시공간 색인 기법을 이용할 경우 이동 객체가 여러 리프 노드에 분산 저장되어 색인 크기가 커지게 된다는 문제가 발생된다. 또한 질의 시 노드의 중복 방문으로 인한 디스크 I/O수의 증가로 많은 비용이 든다는 문제점을 가지고 있다. 특히, 국내에서 사용되고 모바일 사용자의 위치를 획득하기 위하여 사용되고 있는 기술은 망 기반의 Cell/ Sector ID를 사용한 측위 방법을 사용하고 있기 때문에, 동일한 시공간 정보를 가지는 객체 정보를 발생시킨다.

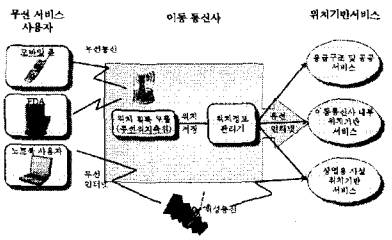

그림 1. 위치기반서비스의 구성

이 논문에서는 이러한 이동 객체의 과거 위치를 관리하기 위한 색인들의 문제점을 해결하기 위하여, 동일한 MBR을 갖는 이동 객체들을 블록 단위로 저장함으로써 전체 색인 크기를 줄여 디스크 I/O수를 감소시킴으로써, 시공간 질의 처리 속도를 향상시켜 효율적인 질의가 가능한 새로운 색인 기법을 제안하였다.

2. 무선위치측위기술

모바일 환경에서 모바일 기기를 사용하는 사용자에 대한 모든 서비스들은 모바일 사용자의 위치를 얼마나 정확하게 측정할 수 있는가가 핵심 요소가 되며, 이러한 모바일 사용자의 정확한 위치정보를 측정하기 위한 기술을 "Wireless Location Technology"[5, 6]라고 한다. 이러한 위치 정보를 제공하는 방법은 크게 두 가지로 나눌 수 있다. 먼저 기존 단말기의 하드웨어나 소프트웨어의 변경이 불필요한 네트워크 망 기반 위치 추적 기술로 Cell/Sector ID 방법, Time Difference of Arrival(TDOA) 방식, Angle of Arrival (AOA) 방법이 있다. 다음으로 기존 단말기에 GPS칩을 내장해야 하는 등 하드웨어와 소프트웨어적인 변경이 필요한 핸드셋 기반 위치 추적 기술로는 GPS를 이용하는 Time of Arrival (TOA) 방식이 있다.

이러한 실세계에서 사용되는 여러 가지 무선위치측위기술들은 모두 공통적으로 기지국 영역 내에서 이동하고 있는 사용자들의 위치정보는 모두 동일하다고 가정하여 위치를 파악하고, 위치가 파악된 이동 객체들을 이동 객체 데이터베이스에 저장하고 검색을 하게 된다.

3. 기존 이동 객체 색인 검토

기존의 이동 객체 정보를 관리하는 기법들은 시간을 하나의 공간 차원으로 간주하여 3차원 R-tree로 구성한 3DR-tree(Three Dimensional R-tree)[7], 모든 timestamp마다 현재의 상태에 대한 2차원 R-tree를 만들어 모든 이전 상태를 2차원 R-tree로 유지하는 구조를 가지고 있으며, 새롭게 생성되는 노드들의 수가 가능한 적게 유지되도록 하는 HR-tree(Historical R-tree)[8], HR-tree의 단점을 보완하여 성능을 개선시킨 HR+-tree(Efficient Historical R-tree)[9]등이 있다.

3DR-tree는 time-slice질의에 대한 성능은 크게 떨어지고, 위치나 모양 변경이 거의 없는 이동 객체 들은 3D R-tree에 커다란 빈 공간(dead space)을 제공한다. 그럼에도 불구하고 3D R-tree는 불필요한 정보가 없기 때문에 색인 크기가 가장 작고, time-interval 질의에 대해서는 좋은 성능을 보인다.

HR-tree는 많은 변화가 있는 객체들에서는 많은 저장 공간을 요구하기 때문에 비효율적이고 실제적인 응용에는 무리가 따른다. 이와 더불어 이 색인 기법은 time-slice 질의에는 성능이 뛰어나지만, time-interval 질의에는 매우 비효율적이고 색인의 크기가 3D R-tree보다 대략 3~4배 크다는 단점을 가지고 있다.

HR+-tree는 HR-tree에 비하여 작은 공간 요구하며, HR-tree의 time-slice 질의 효율성을 상속받았다. 또한 HR-tree에 비해 time-interval 질의에서 더 좋은 성능을 가지고 Buffer 크기가 커짐에 따라 검색 성능이 향상된다. 그러나, HR+-tree는 HR-tree와 마찬가지로 time-interval 질의 시에 색인 노드를 중복해서 방문한다는 단점을 가지고 있다.

하지만, 이러한 기존 색인 기법들은 실세계에서 빈번히 발생하는 동일한 시공간 데이터를 효율적으로 처리하지 못하는 문제를 발생한다.

4. 무선 위치 측위에서 기존 색인의 문제점

제 3장에서 살펴본 바와 같이, 이동하는 객체의 과거 위치 및 궤적 검색을 위하여 제안된 색인들은 질의의 종류에 따라 각각 장·단점을 가지고 있다. 더욱이 현재 모바일 환경에서 제공되는 시공간적으로 동일한 MBR을 가진 객체들의 정보는 처리하여야 할 객체들의 수가 페이지 크기보다 커진 경우 여러 노드에 나누어 저장하여야 한다. 그림 2는 동일한 시공간 정보를 갖는 이동 객체들을 기존의 R-tree기반 시공간 색인 기법에 적용하였을 경우, 동일한 시공간 정보가 여러 노드에 분산 저장되는 것을 나타낸 것이다.

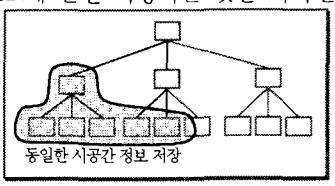

그림 2. 기존 색인에 동일한 MBR을 갖는 이동 객체의 저장

이러한 색인 구조는 같은 지역이나 같은 시간에서 사용자들의 과거 특정 시점 질의라든지, 과거 기간동안 움직임을 파악하기 위한 질의, 특정 지역과 시간에서의 이동 객체들에 대한 궤적 질의 시에 노드를 중복적으로 방문하여 색인 성능을 떨어뜨리게 된다. 또한, 동일한 시공간 정보를 갖는 이동 객체들을 삽입하여 색인을 구성하는 경우 저장되는 데이터의 양이 많아짐에 따라 색인의 크기가 커지게 되고, 객체 삽입, 삭제, 검색시의 접근하는 디스크 I/O수가 증가함으로써 색인 성능의 저하를 가져온다.

따라서 이 논문에서는 이러한 문제점들 가운데 색인 생성 시간 및 검색 시 디스크 I/O에 커다란 영향을 미치는 동일한 시공간 정보를 갖는 다수의 데이터 삽입 시 색인의 성능이 떨어지는 것을 해결하기 위해 동일한 MBR을 갖는 이동 객체들을 처리하기 위한 색인 구조를 제안한다.

5. 동일한 시공간 정보 처리를 위한 색인 구조

제안된 구조에서는 기존 색인에서 사용하는 페이지 단위 저장과 더불어 이동 객체 삽입 시 '입력 캐시'라는 기법을 사용하여 동일한 시공간 정보를 갖는 이동 객체들을 구분하고, 블록 단위로 저장하는 기법을 함께 사용한다. 또한, 제안된 구조의 적용 및 삽입, 삭제, 검색 등의 동작 방법을 설명하기 위하여 기존의 이동 객체 색인 중에서 3D R-tree에 이 논문에서 제안한 방법을 적용시켜 알고리즘을 확장한다.

5.1 기본 동작

이 논문에서 제안한 색인의 전체적인 구조는 동일한 시공간 정보를 처리하기 위하여, 기존의 이동 객체 색인에 동일한 MBR을 갖는 객체들을 저장하기위한 블록 데이터 관리 방법을 추가하였다. 그러나 제안된 색인은 동일한 시공간 정보를 갖는 이동 객체 정보들의 저장을 위하여 이동 객체 삽입 시 입력 캐시를 이용하여 동일한 MBR을 갖는 이동 객체들을 구분한다. 단일 시공간 정보를 갖는 이동 객체들은 페이지 파일에 저장되며, 동일한 시공간 정보를 갖는 이동 객체들은 블록 단위로 관리되는 블록 파일에 저장된다. 즉, 위치 측위 기술에 의해 동일한 시간 정보를 갖는 이

동 객체들, 동일한 공간정보를 갖는 객체들이 수집되고, 입력 캐시를 통해 삽입되는 이동 객체들이 이전에 삽입된 객체와 동일한 MBR을 갖는지 여부를 판별한다. 만일 동일한 MBR을 갖는다면, 블록 파일에 저장되고, 동일한 MBR을 갖지 않는다면, 페이지 파일에 저장된다. 또한, 동일한 시공간 정보를 갖는 이동 객체의 관리를 위하여 이동 객체가 저장된 블록 파일의 Block ID가 페이지 파일의 리프 노드에 저장된다. 페이지 파일의 리프 노드에는 해당 객체가 페이지 파일에 저장되어 있는지, 또는 블록 파일에 저장되어 있는지 여부를 판별할 수 있는 정보도 함께 포함하고 있다. 다음 그림은 제안한 색인의 노드 구조를 표현한 그림이다.

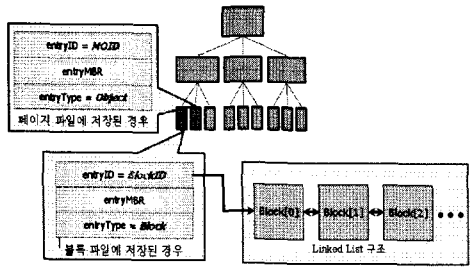

그림 3. 색인 구조

그림 3에서 보는 것과 같이, 이동 객체가 페이지 파일에 저장될 경우, entryID에는 "MOID"가 저장되고, entryType에는 "Object"라는 값이 할당되어 해당 객체가 페이지 파일에 저장되었음을 나타내었다. 또한, 이동 객체가 블록 파일에 저장되었을 경우에는, 제안된 색인의 페이지 파일의 리프 노드에서 entryID에 "BlockID"가 저장되고, entryType은 해당 객체가 블록 파일에 저장되었음을 나타내도록 하기 위해 "Block"이라는 값을 할당한다. 실제 이동 객체가 저장되는 블록 파일은 동일한 시공간 정보를 갖는 객체의 수가 블록 크기를 넘어서 여러 블록 파일에 분산되어 저장될 경우 각 블록 끼리 링크드 리스트 구조로 관리하여 처리한다.

5.2 삽입 알고리즘

현재 모바일 환경에서의 모든 데이터는 시간의 흐름에 따라 연속적으로 발생한다. 제안된 색인 구조에서는 시간적인 순서를 갖는 이러한 이동 객체 정보들의 삽입을 다음과 같이 처리한다. 먼저, 특정 시간에서의 동일한 시공간 정보를 갖는 이동 객체들이 삽입된다. 다음으로 특정 시간에서의 이동 객체 삽입이 이루어진다. 이동 객체 삽입 시 중간 노드의 사각형 영역이 최소로 증가하도록 하여 간접적으로 겹치는 영역을 줄이고 삽입할 리프 노드에 overflow가 발생하였을 때 리프 노드에서 분할(split)이 일어나는 3D R-tree의 삽입 알고리즘을 사용한다. 또한, 동일한 시공간 정보를 가지면서 블록 파일에 저장되는 이동 객체와 단일 시공간 정보를 가지고 페이지 파일에 저장되는 이동 객체를 구분하기 위하여 입력 캐시를 통해 삽입한다. 알고리즘 1은 제안한 색인의 페이지 파일에 새로운 색인 엔트리 E를 삽입하는 삽입 알고리즘을 표현한 것이다.

알고리즘 1. 새로운 이동 객체 삽입 알고리즘

```
INSERT(e : NewEntry)
begin
//동일한 MBR을 갖는 이동 객체인지 여부 판단
   if CheckInputCache(e) then
      return;
   end if
//루트 노드로부터 하위 노드까지 탐색 시작
   node = root
//리프 노드까지 path 결정
   while(node is not a leaf) do
         node = CHOOSESUBTREE(node, e)
   end while
//리프 노드에 새로운 객체 삽입
   INSERTINLEAF(node, e)
//만약 리프 노드가 overflow라면, 분할(Split)이 수행되고 tree 조정
//만약 리프 노드가 overflow가 아니면, path 조정
   if(node overflows) then
         SPLITANDADJUST(node)
   else
         ADJUSTPATH(node)
   end if
   // 입력 캐시 갱신
   UpdateInputCache();
end
```

또한 다음 알고리즘은 동일한 시공간 정보를 가진 이동 객체인지 여부를 판별하여 동일한 MBR을 갖고 있으면 블록 파일에 저장하고, 단일 MBR을 갖고 있는 이동 객체들은 페이지 파일에 저장할지 여부를 결정하는 InputCache 알고리즘을 나타낸 것이다.

알고리즘 2. InputCache 알고리즘

```
Bool CheckInputCache(e : NewEntry)
begin
//동일한 MBR을 갖는지 여부 판별
   if(e의 MBR == MBR) then
 //기존 객체가 페이지 파일에 저장되었나?
      if(entryType==Object) then
         Create Block
         Save ID into Block
         Save e.MOID into Block
         update pageFile
      end if
    //기존 객체가 블록 파일에 저장되었나?
      if(entryType == Block) then
         Save newMOID into Block
      end if
         return true;
//동일한 MBR을 갖고 있지 않다면
   else
         return false;
   end if
end
```

5.3 삭제 알고리즘

제안한 색인 방법의 삭제 알고리즘은 객체를 삭제한 후 그 노드의 엔트리 수가 최소 엔트리 수보다 적으면 그 노드를 삭제하고 그 노드의 엔트리들을 재 삽입(Re-insertion)하는 3D R-tree의 삭제 알고리즘을 사용한다. 제안한 색인 구조의 삭제 알고리즘과 기존 색인에서 사용하는 삭제 알고리즘과의 차이점은 삭제할 엔트리를 찾을 경우 entryType에 따라 페이지 파일에 저장된 이동 객체를 찾아야 하는지, 아니면, 블록 파일에 저장된 이동 객체를 찾아야 하는지 여부를 결정해 주어야 한다는 점이다. 알고리즘 3은 제안한 색인 방법에서 특정 엔트리 E를 삭제하는 삭제 알고리즘을 표현한 것이다.

알고리즘 3. 삭제 알고리즘

```
Delete (e : LeafEnty) Algorithm
begin
//엔트리 e를 포함하는 리프 노드 찾기
   L = FINDLEAF (e)
  if (entryType ==Object) then
//엔트리들 삭제하고 L과 e로부터 시작하는 트리로 재구성
//재구성 결과 노드들의 집합 Q 반환
   Q = REORGANIZE (L, e)
//Q의 노드들 안에 엔트리들을 재 삽입
    REINSERT (Q)
  else
//블록 파일에 존재하는 엔트리 삭제
   DeleteFromBlock(e)
  end if
end
```

5.4 이동 객체의 검색

이동 객체 데이터베이스에 저장된 이동 객체를 검색하기 위한 알고리즘은 루트부터 시작해서 트리의 아래 방향으로 검색하며 질의 영역과 겹치는 중간 노드의 사각형들에 대해 대응하는 자식 노드들을 루트로 하여 재귀적으로 검색하는 방법을 사용한다. 알고리즘 4는 제안된 색인에서 포인트 질의를 표현한 것이다.

알고리즘 4. 포인트 질의 알고리즘

```
PointQuery (P : Point) : set(oid) 알고리즘
begin
  Result = 0
// 단계 1 : 루트 노드로부터 트리를 운행하고, 그 다음 SL 계산
// 또한, P는 dr을 포함한 리프 노드들의 집합을 의미
   SL= TREETRAVERSAL (root, P)
// 단계 2 : 리프 노드를 탐색하고, P를 포함한 엔트리들을 유지
    for each L in SL do
// 리프 노드 L안에 엔트리 들을 탐색
      for each e in L do
       if (e.mbr contains P) then
         if (entryType==Object) then
           result += {e.oid}
         end if
         else
           result += {GetBlockIDs()}
        end if
       end if
      end for
   end for
  return result
end
```

5.5 제안된 색인의 효율성

기존 색인 방법에서 사용하는 페이지 파일 저장방식만을 이용해 이동 객체를 저장하면, 구조가 비교적 단순하고 구현이 용이한 특징을 가지고 있다. 하지만, 이러한 페이지 파일에서의 저장은 이동 객체가 데이터베이스에 삽입되는 순서를 가지고 페이지 파일에 저장된다. 모바일 환경의 위치기반서비스에서 현재 우리나라에서 사용하는 무선 측위 기법인 Cell/Sector ID 방식을 사용하거나, 또는, 기지국을 사용한 무선측위기법을 통해 모바일 기기를 사용하는 사용자의 위치를 데이터베이스에 저장할 경우 기존의 페이지 파일을 사용한 색인 방법을 사용하면 효율적인 검색을 할 수 없게 된다. 따라서 동일한 시공간 정보를 갖는 이동 객체들을 블록 단위로 저장하기 위한 구조를 페이지 파일 저장 구조와 독립적으로 두고 사용하면 동일한 MBR을 가지는 이동 객체들이 같은 블록 안에 저장되기 때문에, 페이지 파일에서의 검색과 같은 중복 방문이 없어지게 되며, 이로 인해 디스크 I/O수가 작아져 비용이 줄어들게 된다. 또한, 색인의 크기가 페이지 파일만을 사용한 저장 구조 보다 작게 되므로 질의 수행 시간이 더욱 빨라지게 된다는 장점이 있다.

6. 결 론

이 연구에서는 지금까지 위치기반서비스에서 개발되어야 할 대표적인 기반 기술 중에 하나로 모바일 사용자의 위치를 파악하기 위한 무선위치측위기술들에 대해 살펴보았다. 또한, 이들의 효율적인 검색을 위한 기존 이동 객체 색인방법들을 분석하였으며, 기존 색인에서의 문제점을 파악하고, 이 문제의 해결방법을 제시하였다. 즉, 기존 색인에서 사용하는 이동 객체의 페이지 파일 저장 구조와 더불어 동일한 공간 정보를 갖는 이동 객체들을 따로 독립적인 블록 파일에 저장하는 기법을 제시함으로써 중복검색 문제를 해결하였다. 이 제안된 방법은 동일한 MBR을 갖는 이동 객체들을 블록 단위로 저장함으로써 색인의 크기를 줄여 중복검색을 해결하였다. 아울러, 전체 색인 크기를 작게 하여 디스크 I/O수를 감소시킴으로써, 시공간 질의 처리 속도를 향상시켜 효율적인 질의가 가능하도록 하였다.

앞으로의 연구는 동일한 시공간정보를 갖는 데이터들에 대하여 기존 기법들과 제안된 기법과의 성능을 평가하고 분석하는 것이다. 또한, 이러한 이동 객체 저장 기법에 관한 성능 분석 연구와 더불어, 이동 객체의 시간적, 공간적 연관성을 고려하여 혼합질의에 우수한 성능을 보이는 버퍼관리방법에 대한 연구가 필요하다.

참고문헌

[1] Martin Erwig, Ralf H. Guting, Markus Schneider, Michalis Vazirgiannis, "Spatio- Temporal Data Types : An Approach to Modeling and Querying Moving Objects in Databases", GeoInformatica, Vol. 3, No. 3, 1999.

[2] Ralf Hartmut Guting, Michael H. Bohlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos, Markus Schneider, Michalis Vazirgiannis, "A foundation for representing and quering moving objects", ACM Transactions on Database Systems, Vol.25, No.1, 2000.

[3] 류근호, 안윤애, 이준욱, 이용준, "이동 객체 데이터베이스와 위치기반서비스의 적용", 데이터베이스연구회지, Vol.17, No.03, 2001.

[4] 양영규, "위치 기반 서비스(LBS: Location Based Service) 기술 현황 및 전망", 정보처리학회 Vol18, No6, 2001.

[5] 금동준, "Wireless Location Technology-Survey on technology and standard", IT Forum Korea 2002, 2002.

[6] 이상정, "위치기반서비스에서의 무선 측위 기술", 제 1회 국제 LBS 기술 워크샵, 2002.

[7] Vazirgiannis M., Theodoridis, Y., Sellis, T., "SpatioTemporal Composition and Indexing for Large Multimedia Applications", ACM Multimedia Systems, 1998.

[8] M.A. Nascimento, J.R.O. Silva, "Towards historical R-trees", In Proc. of the 1998 ACM Symposium on Applied Computing, 1998.

[9] Tao. Y., Papadias. D., "Efficient Historical R-trees", Proceedings of 13th IEEE Conference on Scientific and Statistical Database Management (SSDBM), 2001.