공간 네트워크 데이터베이스에서의 질의 처리 알고리즘의 설계

김용기° 장재우
전북대학교 컴퓨터공학과
{ykkim°, jwchang}@dblab.chonbuk.ac.kr

Design of Query Processing Algorithms in Spatial Network Databases

Yong-Ki Kim°, Jae-Woo Chang
Dept. of Computer Engineering, Chonbuk National University

요 약

최근 이동 객체를 위한 공간 데이터베이스에 관한 연구가 활발히 진행되어 왔다. 그러나, 주로 제한조건이 없는 이상적인 공간에서의 연구가 진행되어져 왔기 때문에, 도로나 철도와 같은 이미 정해진 공간 네트워크 상에 적용하는데는 문제점을 지니고 있다. 따라서, 본 논문에서는 기존 연구가 지니고 있는 문제점을 제시하고, 공간 네트워크 데이터베이스에 적합한 효율적인 질의 처리 알고리즘을 설계한다.

1. 서 론

일반적으로 공간 데이터베이스 (spatial databases) 분야는 지난20년 이상 활발한 연구가 진행되어 왔다 [1]. 한편 이동 객체를 위한 대부분의 기존 연구는, 제한조건이 없는 이상적인 공간을 가정하고 설계되었으나, 이는 실제 응용에 직접 적용하는데 문제를 지니고 있다. 예를 들면, Euclidean 공간에서 질의와 가장 가까운 주유소가 (gas station), 실제로는 직선 도로가 없어서 도로 네트워크 상에서는 다른 주유소보다 거리가 더 멀 수 있다. 따라서 LBS(location-based service) 의 효과적인 지원을 위해, 최근에는 이상적인 공간 대신, 실제 도로나 철도와 같은 공간 네트워크(network) 을 고려한 연구가 활발하게 수행 중에 있다 [2, 3, 4, 5, 6]. 이러한 공간네트워크 데이터베이스 연구는 일반 공간 데이터베이스의 연구와 마찬가지로 크게 3가지 주제로 요약된다. 즉, 공간 네트워크를 위한 데이터 모델, 질의 처리 알고리즘, 공간 네트워크 데이터베이스를 위한 저장 및 색인구조 등이다.

본 논문에서는 위의 3가지 주제 가운데, 공간네트워크 데이터베이스에서 효율적인 질의처리 알고리즘을 다룬다. 공간 네트워크는 도로나 철도와 같은 이미 정해진 공간 네트워크 상에서 객체의 이동이 이루어지기 때문에, 공간 네트워크 상에서의 질의처리 알고리즘은, 기존의 제한조건이 없는 이상적인 공간을 가정한 기존 질의처리 알고리즘과는 매우 다르다. 최근 공간 네트워크 상에서의 질의 처리 알고리즘의 연구로, 이상적인 공간을 가정한 기존 알고리즘을 확장하는 방법과 네트워크를 확장하는 방법 등이 연구되었으나 [6], 이 방법들은 응용 환경에 따라서 다수의 디스크 I/O 나 불필요한 연산 등이 존재한다. 따라서 본 연구에서는 앞서의 두 가지 방법을 효과적으로 결합하여, 응용 환경에 관계없이 보다 효율적으로 질의를 처리하는 알고리즘을 제안한다. 제안하는 알고리즘은 공간 네트워크 데이터베이스의 대표적인 질의인, k-최근접 질의, 범위 질의, e-distance join 질의를 효율적으로 처리해 준다.

논문의 구성은 다음과 같다. 2장에서는 관련 연구를 소개하고, 3장에서는 본 논문에서 설계한 질의처리 알고리즘을 제시한다. 4장에서는 결론 및 향후연구를 제시한다.

본 연구는 대학 IT연구센터 육성·지원사업의 연구결과로 수행되었음

2. 관련 연구

공간 네트워크 데이터베이스에서 질의처리 알고리즘에 관한 연구는 크게 3가지이다. 첫째, 덴마크의 Aalborg 대학에서는 도로 네트워크에서 이동객체의 최근접 질의처리를 위한 데이터 표현 기법을 제시하고, 이러한 데이터 표현 기법에 근거하여, client-server 구조상에서 k-최근접 질의 처리 알고리즘을 제시하였다 [4]. 즉, 서버에서 초기의 최근접 질의 처리 결과인 최근접 후보집합을 생성한다. 다음으로 클라이언트에서 질의와 후보집합의 점 사이의 거리를 재계산하여, 현재의 질의 결과를 유지한다. 이것은 사용자에게 최근의 상태를 반영한 정확한 질의 결과의 제공을 가능하게 해 준다. 둘째, Univ. of Southern California 에서는 도로 네트워크에서 k-최근접 질의를 위한 road network embedding (RNE)기법을 제안하였다 [5]. 이 기법은 도로 네트워크를 거리 함수를 용이하게 이용할 수 있는 고차원 공간으로 변환하고, Chessboard 거리에 기반하여 점들 간의 실제 거리의 근사치(approximation)를 계산한다. 그러나, RNE 기법의 한가지 단점은 네트워크 내의 모든 점들 간의 shorestest path를 미리 계산을 해야 한다는 것이다. 이를 극복하기 위해, 미리 계산된 소수의 점들 간의 shortest path 만을 가지고도, 어느 정도 높은 수준의 정확도로 k-NN 질의를 처리하는 truncated RNE 기법을 제안하였다. 마지막으로, HKUST 에서는 공간 네트워크와 Euclidean 공간을 합성한 구조를 제안하였다 [6]. 이 구조에 근거하여, 탐색 공간을 효율적으로 가지치기(pruning)를 하기 위하여 Euclidean restriction 및 network expansion 기법을 제안하였다. 아울러 이 기법들을 대표적인 공간 질의, 즉 최근접 질의, 범위 질의, closest pairs, e-distance join 질의에 적용하였다.

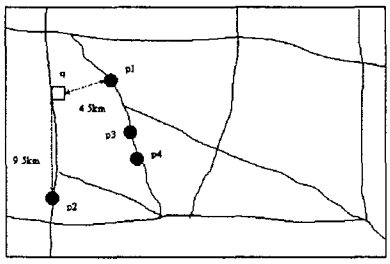
3. 공간 네트워크 질의처리 알고리즘의 설계

3.1 k-최근접 질의처리 알고리즘

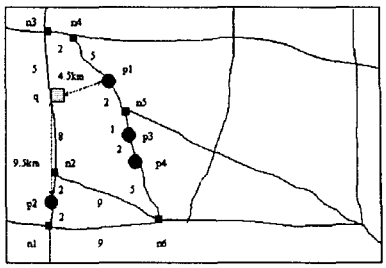
공간 네트워크 상에서의 최근접 질의처리 알고리즘은, 객체가 이미 정해진 공간 네트워크 상에서만 이동이 가능하므로, Euclidean 공간을 가정한 기존 최근접 질의처리 알고리즘과는 매우 다르다[7,8]. 예를 들면, 이상적인 공간인 Euclidean 공간에서 질의의 최근접 점이, 공간 네트워크상의 공간에서는 최근접 점이 되지 못하는 경우가 빈번히 발생한다. (그림 1) 는 이러한 예를 나타내고 있다. 즉, Euclidean 공간에서 질의 q의 최근접 점은

(예를 들면 gas station) p1으로써 둘 사이의 Euclidean 거리는 4.5Km이다. 그러나 실제 공간 네트워크상에서의 q의 최근접 점은 p2 이며, 둘 사이의 거리는 약 10Km 이다. 왜냐하면 q와 p1 사이에는 직선 도로가 존재하지 않음으로 해서, 실제의 도로상에서의 둘 사이의 거리는 10km 를 초과하기 때문이다.

이러한 공간 네트워크상에서의 k-최근접 질의처리 알고리즘을 설계하기 위하여, 가장 먼저 고려할 수 있는 방법은 기존 Euclidean 공간을 가정한 기존 k-최근접 알고리즘[8]을 확장하는 접근방법이다. 그러나 이러한 접근방법은 실제 네트워크 상에서의 최근접 점이 찾아지기 전에, 많은 수의 계산 및 디스크 I/O가 요구된다는 단점이 존재한다. 예를 들면, 기존 알고리즘을 확장한 접근방법을 사용하면, (그림 1) 에서 질의 q의 공간 네트워크 상에서의 최근접 점을 위해, 먼저 Euclidean 공간상에서의 최근접점 p1을 찾아 이것의 공간 네트워크상의 거리, 즉 $d_{SN}(q, p1)$를 구한다. 다음 단계에서는 $d_{SN}(q, p1)$으로 필터링이 이루어지며, 그러나 (그림 1)에서는 모든 점들이 이 범위 내에 있음으로 해서, 전혀 필터링이 이루어지지 않고, 따라서 검색 성능이 현저히 감소하는 현상이 발생한다. 이러한 문제점을 극복할 수 있는 방법으로 연구된 것이 네트워크 확장 방법이다 [6]. 이 방법은 질의가 속한 에지를 탐색하여, 이 에지를 시작으로 k-최근접 점을 찾을 때까지 계속하여 에지를 큐(queue)에 추가하여 네트워크를 확장해 가는 방법이다. 이 방법을 통해 얻어지는 k-최근접점 리스트의 거리는 공간 네트워크상의 거리를 의미하며, 새로이 추가된 에지를 통해 도달 가능한 노드의 거리가 k-최근접점 거리보다 커지면 알고리즘은 종료된다. 예를 들면, 네트워크 확장 알고리즘은 (그림 2) 에서 질의 q가 속 한 에지 e(n2,n3) 를 탐색한다. 이 에지의 탐색을 통해 $d_{SN}(q, n2) = 8$, $d_{SN}(q, n3) = 5$ 를 계산한다. 여기서 q에서 n3까지의 거리가 짧으므로, 에지 e(n3,n4)를 추가하는 네트워크 아직까지는 어떠한 POI가 찾아지지 않았으며, 이때의 큐의 상태는, 즉 에지 추가에 따른 도달 가능한 노드와 q에서 그 노드까지의 네트워크 상에서의 거리, Q=<(n4,7), (n2,8)> 이다. 따라서 다음단계의 에지 추가는 n4와 incident 한 에지 e(n4, n5) 가 된다. 이 확장을 통하여 하나의 최근접점 후보자인 p1을 탐색하고, $d_{SN}(q, p1) = 12$가 공간 탐색 bound 인 dmax가 된다. 이때의 queue 상태는 Q=<(n2,8), (n5,14)> 이다. 따라서 n2에 incident 한 에지들로 네트워크 확장이 이루어지고, 이때 p2가 탐색된다. $d_{SN}(q, p2) = 10$ 는 현재의 dmax보다 작기 때문에, p2가 새로운 최근접점 후보자가 되며 dmax를 10으로 변경한다. 이때의 queue 상태는 Q=<(n1,12), (n5,14), (n6,17)> 이다. 다음 단계에서 n1에 incident 한 에지들로 네트워크 확장을 수행하고자 하는 데, 이미 n1까지의 거리가 dmax를 초과하고 있으므로, 더 이상 짧은 거리의 최근접점을 찾는 것은 불가능하여 알고리즘은 종료된다. 따라서 현재의 최근접점 후보자인 p2가 최종의 최근접점이 되고, 거리는 10이 된다.



(그림 1) 공간 네트워크 상에서의 최근접 질의



(그림 2) 공간 네트워크 확장 최근접 질의처리

앞서 살펴본 Euclidean 공간 기반 알고리즘 확장방법과 네트워크 확장방법은 서로 다른 접근 방법의 알고리즘이다. 즉, 전자의 알고리즘이 global한 최근접점을 Euclidean 공간에서 찾아 그것의 네트워크 공간상의 거리를 계산하는 방법이라면, 후자의 알고리즘은 질의가 속한 local 한 지역으로부터 네트워크를 확장해 가는 방법이다. 따라서 뉴욕의 맨해턴과 같은 바둑판 형태의 도로 지역에서는 전자의 알고리즘이 우수하며, 산지가 많아 도로가 평지 중심으로 만들어진 지역에서는 후자의 알고리즘이 우수하다. 따라서 모든 경우에 효율적인 알고리즘을 설계하기 위하여, 위 두 접근방법을 효과적으로 결합하는 연구가 필요하다. 이를 위해 3가지의 고려사항이 필요하며, 이는 다음과 같다.

- 고려사항 1: 어떤 응용 특성을 지닌 공간 네트워크 간에 효율적으로 동작할 수 있는 최적에 가까운 초기 k-최근접점을 획득하여야 한다.
- 고려사항 2: 최적에 가까운 초기k-최근접점을 획득하였다면, 이에 근거하여 가능한 빨리 최종의 k-최근접 점에 도달할 수 있어야 한다.
- 고려사항 3: k 값에 의존하여, k 값이 적을수록 Euclidean 공간 기반 알고리즘 확장방법이 worst case 가 발생한 확률이 많으며, k 값이 클수록 네트워크 확장방법이 worst case 가 발생한 확률이 많다.

이러한 위의 3가지 고려사항을 만족하도록 설계된 k-최근접 질의처리 알고리즘은 (그림 3)과 같다.

```
Algorithm K-NN(q, k)  /* q is the quey point */
1. depending on k, determine k', the number of initial
   candidates for k-NN,
and c', the number of edge connection from q for acquiring
   initial candidates
2. {p1,...,pk'} = Euclidean-NN(q,k)  /* k' < k */
3. for each POI pi
        dN(q,pi) = compute_networkDistance(q, pi)
4. sort {p1,...,pk'} in ascending orderof dN(q,pi)
5. e(ni,nj) = find_edge(q)
6. Q' = <(ni, dN(q,ni)), (nj, dN(q,nj))> /* sorted on their
   network distance */
7. (En, Q) = expand_network(e(ni,nj), c, Q')  /* En is the set
   of edges in a network which is expanded by edge e(ni,nj)
   within c' connections from the edge. Q is updated
   through network expansion from Q' */
8. Se = find_POI(En) /* Se is the set of POIs covered by the
   edge set En */
9. {p1,...,pk} = the k (network) nearest POIs by merging
   {p1,...,pk'}and Se sorted in ascending order of their
   network distance (pm,...,pk may be ...if the merged result
   contains just m-1 POIs with m ≤ k )
10. dmax = dN(q,pk) /* if pk = ..., dmax = ∞ */
11. update Q by insering into the queue, pj in the merged
```

```
    result which is originated from {p1,...,pk'} acquired by
    Euclidean-NN(q,k) */
12. delele from Q the node n with the smallest dN(q,n)
13. while(dN(q,n) < dmax) {
14.         for each non-visited adjacent node nj of n {
15.                 Sp = find_POI(e(nj,n))
16.                 update {p1,...,pk} from {p1,...,pk} U Sp
17.                 dmax = dN(q,pk)
18.                 insert (nj, dN(q,pj)) into Q  }  /* end of for
    each */
19.         delele from Q the next node n with the smallest
    dN(q,n)
20. } /* end of while
End K-NN
```

(그림 3) 제안하는 최근접 질의 처리 알고리즘

3.2 범위 질의처리 알고리즘

공간 네트워크 상에서의 범위 질의처리 알고리즘은, 앞서와 마찬가지로 Euclidean 공간을 가정한 기존 범위 질의처리 알고리즘과는 매우 다르다. 예를 들면, (그림 3) 에서 모든 점이 주유소라고 가정하고, "질의 q의 10Km 내에 존재하는 주유소를 찾아라." 라는 질의가 있다고 가정하자. 이때 Euclidean 공간상에서 찾는다면, p1, p2, p3, p4 가 검색될 것이지만, 실제 공간 네트워크 상에서는 p2만이 이 질의를 만족한다.

한편 Euclidean 공간을 가정한 범위 질의처리 알고리즘을 확장한 접근방법은 질의를 만족하는 범위내의 점들을 검색하기 전에, 다수의 디스크 I/O 가 요구된다는 단점이 존재한다. 한편 질의에서 r이내의 거리에 있는 영역을 탐색하여 범위내의 점들을 찾는 네트워크 확장 방법은 영역 내의 거리 밖의 점들까지 불필요한 검색과 연산을 수행한다는 단점이 존재한다.

따라서 이러한 단점들을 보완하여 보다 효율적인 알고리즘을 설계하기 위하여, 두 알고리즘을 효과적으로 결합하는 연구가 필요하며, 이에 따라 제안된 범위 질의 처리 알고리즘은 (그림 4)와 같다

```
Algorithm Range(q, r, node_id, QS)
/* q is the quey point and r is the network distance */
1. result = ϕ
2. S = Euclidean-range(q,r)
3. if node_id is an intermediate node
4.      compute QSi for each entry Ei in node_id //join
5.          Ei = Ei - (Ei - S)      // remove needless POIs
6.      for each entry E in node_id
7.          if(QSi ≠ ϕ )
8.              Range(q,r,Ei.node_id, QSi)
9. else         // node is a leaf node
10.     for each entry E in node_id
11.         if (match(node_id.entry, QSi))
12.          dN(q,node_id. entry) = compute_NetworkDistance(q,
                                    node_id. entry)
13.         insert from result the entry E
End Range
```

(그림 4) 제안하는 범위 질의처리 알고리즘

3.3 e-distance join 질의처리 알고리즘

e-distance join 질의처리 알고리즘은 범위 질의처리 알고리즘의 확장으로 볼 수 있다. 즉 범위 질의처리 알고리즘이 주어진 질의 q에서 범위 e 내에 존재하는 점들을 찾는 질의라면, e-distance join 질의처리 알고리즘은 범위 e 내에 존재하는 점들의 쌍을 찾는 질의이다. 예를 들면, e-distance join 질의는 "10Km 내에 존재하는 호텔 및 레스토랑의 쌍을 찾아라." 라는 형태이다. 만약 (그림 2)에서 p1과 p2 가 호텔에 속하는 집합이며, p3과 p4가 레스토랑에 속하는 집합이라고 가정하면, 이 질의를 만족하는 쌍은 <p1,p> <p1,p4> 이다. 이러한 공간 네트워크 상에서의 e-distance join 질의처리 알고리즘을 위하여, 앞서와 마찬가지로 2개의 알고리즘을 고려해 볼 수 있다. 즉, 기존 Euclidean 공간을 가정한 e-distance join 질의처리 알고리즘을 확장하는 접근방법과, 에지를 추가하여 네트워크를 확장하는 접근방법이다[6]. 그러나 두개의 알고리즘은 응용 환경에 따라 서로간의 장, 단점을 지니고 있기 때문에, 보다 효율적인 알고리즘을 설계하기 위하여, 두 알고리즘을 효과적으로 결합하는 연구가 필요하다. 이에 따라 제안된 e-distance join 질의처리 알고리즘은 (그림 5)와 같다.

```
Algorithm e-Distance(S, T, r) /* S and T are two POI data sets
      and r is the network distance */
1. result = ϕ
2. while S has not been exhausted
3.     get next s1,...,sn points
4.     for each point si
5.         QSsi = expand_point(si,r)
6.         let QS = the Union of QSsi (for 1≤i≤n)
7.         Range(q, r, rootRtree, QS)
8. end while
End e-Distance
```

(그림 5) 제안하는 e-Distance join 질의처리 알고리즘

4. 결론 및 향후 연구

본 논문에서는 공간 네트워크 데이터베이스에서 기존에 연구되었던 Euclidean 공간 기반 알고리즘 확장방법과 네트워크 확장방법을 효과적으로 결합하여, 보다 효율적인 질의 처리가 가능한 알고리즘을 제안하였다. 향후 연구로는 본 논문에서 제안한 알고리즘을 앞서의 두 가지 확장방법의 알고리즘과 성능을 비교하는 연구이다.

참고문헌

[1] S. Shekhar et al., "Spatial Databases Accomplishments and Research Needs," IEEE Tran. on Knowledge and Data Engineering, Vol. 11, No. 1, pp 45-55, 1999.

[2] T. Brinkhoff, "A Framework for Generating Network-Based Moving Objects," GeoInformatica, Vol. 6, No. 2, pp 153-180, 2002.

[3] L. Speicys, C.S. Jensen, and A. Kligys, "Computational Data Modeling for Network-Constrained Moving Objects," Proc. of ACM GIS, pp 118-125, 2003.

[4] C.S. Jensen, J. Kolar, T.B. Pedersen, and I. Timko, "Nearest Neighbor Queries in Road Networks," Proc. of ACM GIS, pp 1-8, 2003.

[5] C. Shahabi, M.R. Kolahdouzan, M. Sharifzadeh, "A Road Network Embedding Technique for K-Nearest Neighbor Search in Moving Object Databases," GeoInformatica, Vol. 7, No. 3,, pp 255-273, 2003.

[6] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query Processing in Spatial Network Databases" Proc. of VLDB, pp, 802-813, 2003.

[7] N. Roussopoulos, S. Kelly, and F. Vincent, "Nearest Neighbor Queries," Proc. of ACM SIGMOD, pp 71-79, 1995.

[8] T. Seidl and H. Kriegel, "Optimal Multi-step k-Nearest Neighbor Search, Proc. of ACM SIGMOD, 1998.