무순위 연속 k 최근접 객체 탐색을 위한 효율적인 분할점 추출기법

김 진 덕*

*동의대학교

A Efficient Method of Extracting Split Points for Continuous k Nearest Neighbor Search Without Order

Jindeog Kim

*Dongeui University

E-mail : jdk@deu.ac.kr

요 약

최근 이동 중인 경로 상에 존재하는 모든 지점에 대해 k개의 최근접 객체를 탐색하는 연속 k 최근접 객체 탐색 질의가 위치기반 서비스와 지능형 교통 시스템의 응용 분야에 폭넓게 사용되고 있다. 이러한 질의는 위와 같은 응용에 빠른 응답을 요구하고, 공간 네트워크 데이터베이스에 적용가능해야 한다.

이 논문에서는 공간네트워크 상에서 움직이는 질의 객체를 위한 최근접 객체를 효율적으로 탐색하는 새로운 기법을 제안하고자 한다. 제안하는 기법은 다수의 분할점과 그에 상응하는 k개의 최근접 객체 집합들을 결과로 추출하며, POI들 간에는 순서가 없다. 분석을 통해 제안한 기법에 기존 기법에 비해 우수함을 보인다.

ABSTRACT

Recently, continuous k-nearest neighbor query(CkNN) which is defined as a query to find the nearest points of interest to all the points on a given path is widely used in the LBS(Location Based Service) and ITS(Intelligent Transportation System) applications. It is necessary to acquire results quickly in the above applications and be applicable to spatial network databases.

This paper proposes a new method to search nearest POIs(Point Of Interest) for moving query objects on the spatial networks. The method produces a set of split points and their corresponding k-POIs as results. There is no order between the POIs. The analysis show that the proposed method outperforms the existing methods.

키워드

CkNN, 공간 네트워크, 분할점, LBS

Ⅰ. 서 론

최근 위치기반서비스(Location Based Service)와 ITS(Intelligent Transportation System)등의 응용 서비스가 활성화되면서 최근접 객체(Nearest Neighbor) 검색 등에 대한 관심이 고조되고 있다. 지금까지 다양한 유형의 최근접 객체 검색에 대한 연구가 진행되었지만, 공간 네트워크 경로를 진행 중인 질의점과 고정객체에 대한 연속 kNN 질의는 많은 응용분야에서 사용되고 있으며, LBS와 ITS의 주요 연산이다. 이러한 연산들은 많은 수의 질의점과 POI 객체가 관련되어 있으며, 빠른 질의처리를 요구한다. 그러나 지금까지의 연구는 연속kNN 질의 처리를 위해 각 단계마다 거리 및 방향 계산을 요구하는 기법으로 실시간 질의 응답 시간을 제공하기 어렵다.

따라서 이 논문에서는 무순위 연속 k 최근접 객체 검색 기법을 제안하고자 한다. 제안하는 기법은 공간 네트워크의 각 노드별로 최근접 객체들을 저장하며, 이동 경로에 대해 kNN이 변경되는 지점인 분할점을 간단한 계산에 의해 추출한다. 또한 무순위 객체를 검출하므로 보다 적은 수의 분할점을 생성하며, 거리와 방향계산을 하지 않아 수행 시간을 단축할 수 있다.

이 논문의 구성은 다음과 같다. 제 2 장에서는 정규최근접 객체 검색, 연속 최근접 객체 검색, 이동체 연속 최근접 객체검색 기법에 대한 관련 연구를 살펴보고 제 3장에서는 이 논문에서 제안하는 무순위 연속 k 최근접 객체 기법에 대해 자세히 설명하고, 기존 연구와 비교 분석 결과도 제시하며, 제 4장에 결론을 맺는다.

II. 관련연구

과거에는 하나의 질의점과 고정 객체 사이의 전통적인 kNN에 대한 연구가 활발히 진행되었지만, 최근에는 질의점이 이동 중인 연속 kNN에 대한 연구 및 이동 중인 POI 객체에 대한 최근접 객체의 모니터링에 관한 연구[4]도 있으며, 유클리디언(Euclidean) 거리보다 공간 네트워크 기반 거리[3]를 활용한 최근접 객체에 대한 연구가 진행 중이다.

일반적인 k 최근접 객체 검색 기법은 인덱스[5,7]를 사용하는 기법이 많이 연구되어 왔다. 관련 연구 [5]에서는 R-tree를 사용하여 최근접 객체를 검색하는 기법을 제안하였다. 그러나 이 연구는 깊이 우선 탐색에 의해 불필요하게 많은 디스크 탐색이 요구된다. 그리고 유클리디언 거리를 기초로 제안된 기법으로서 공간 네트워크에 적용하기 어렵다.

그리고 계산 시간의 단축을 위해 사전에 미리 계산된 정보를 활용하는 연구[1,2]가 진행되었어며, 질의마다 보로노이(Voronoi) 다이어그램을 이용하여 1개의 최근접 객체와 인접 보로노이 다각형 정보를 저장하고, 최근접 객체 탐색 시 영역을 확장하는 기법을 관련연구 [2]에서 제시하였다.

최근 이동 객체의 연구와 더불어 연속 k 최근접 객체 검색에 관한 연구도 활발하다. 관련연구 [8]에서는 연속 k 최근접 객체 검색 모델을 처음으로 제안하였으며, 관련연구 [6]에서는 시간기반 질의 개념을 도입하였다. 이 기법 또한 점진적인 결과 집합을 추출한다. 그러나 이들 CkNN기법 또한 유클리디언 거리를 기반으로 수행된다.

관련 연구 [9]는 도로 네트워크 기반 cNN 기법을 제시하였으며, 관련 연구 [3]은 도로 네트워크 기반 CkNN을 제시하였다. 이 연구에서는 질의 경로 상에서 kNN객체가 변하는 분할점을 추출하는 기법을 제시하였다. 그러나 거리 및 방향을 조사하는 횟수가 과도하게 많으며, 각 단계마다 증가노드와 감소노드를 재계산해야 하는 단점이 있다.

본 논문에서는 도로 네트워크 기반으로 각 k개 객체간의 무순위 CkNN 탐색 기법을 제시하고자 하며, 증가 및 감소 노드 분할법을 통해 분할점을 효과적으로 추출하는 기법을 제안하고자 한다.

III. 무순위 연속 kNN 탐색 기법

무순위 연속 k 최근접 객체 탐색 질의는 사용자가 주어진 경로로 이동 중일 때 연속적인 k개의 최근접 객체를 탐색하는 것으로 정의한다. 그림 1은 이 질의의 예를 든 것이다. 이동 객체(예, 자동차)가 경로 A, B, C, D를 주행 중이며, 이동 중인 각 시점마다 해당 3개의 최근거리 식당(r_1, ……. r_8)을 무순위로 검색하는 질의이다. 이 질의의 결과는 구간과 해당 kNN의 집합으로 구성된다. 이 때 kNN이 변하는 분할점을 추출함으로써 구간을 설정할 수 있다.

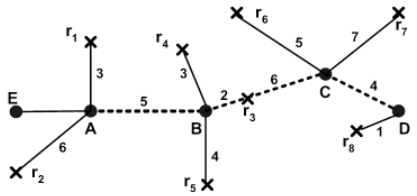

그림 1. 도로 네트워크와 POI 객체

그러므로 무순위 연속 kNN 탐색 질의는 주행 중인 경로 상에 분할점을 효율적으로 탐색하는 문제로 귀결된다. 또한 현재까지의 대부분의 연구가 유클리디언 거리를 기반으로 kNN을 검색하는 반면, 본 논문에서는 공간네트워크상의 kNN 검색 기법을 제안하고자 한다. 제안하는 기법은 사전에 검색된 각 노드의 최근접 객체 정보만을 이용한다. 또한, 기존 연구[3]의 점진적인 구간 추출 작업에 비해 제안한 기법은 일괄작업으로 진행된다. 그리고 일괄작업시 Dijkstra 알고리즘과 같은 최단 경로 검색 과정이 배제되는 효율적인 분할점 추출 기법이다.

3.1절에서는 무순위 연속 kNN 탐색 질의를 위한 도로 네트워크의 자료 구조에 대해 설명하며, 3,2절에서는 이 논문에서 제안한 무순위 연속 kNN 탐색 기법에 대해 자세히 기술한다. 향후, 이 논문에서는 무순위 연속 kNN 탐색 기법을 CkNN_NO(Continuous k Nearest Neighbor_No Order)로 명명한다.

3.1 CkNN_NO를 위한 자료 구조

제안하는 기법은 공간 네트워크를 기반으로 주어진 질의 경로에 대해 최근접 POI 객체를 탐색한다. 공간 네트워크는 노드테이블, 링크 테이블, POI 테이블로 구성된다. 노드 테이블은 ID, 노드에 연결된 링크 ID, 노드에 인접해 있는 n개의 POI ID 및 노드에서의 거리 정보를 가지고 있다.

링크 테이블은 ID, 시작 노드와 끝노드 및 링크의 길이정보를 포함하고 있다. POI 테이블은 ID, POI가 놓여있는 링크 ID 및 시작 노드에서의 위치 정보를 저장한다.

3.2 CkNN_NO 처리 단계

이 기법은 3단계로 구성된다. 1단계는 각 노드에 인접해 있는 n개의 POI 객체를 구하는 것으로서 사전계산에 의해 테이블에 저장된다. 이 때 Dijkstra 알고리즘에 의해 노드와 POI간의 거리를 구하여 가장 가까운 n개의 POI를 선택한다. 2단계는 두개의 노드에 연결된 2k개의 POI에 대해 증가객체와 감소객체를 구별한다. 3단계는 분할점을 추출한다.

(1) 증가객체와 감소객체의 구분

두개의 노드사이에 존재하는 분할점을 구하기 위해 두 노드에 속해있는 최근접 POI객체 O를 증가객체와 감소객체로 구분한다. S와 E노드는 각각 출발점과 종료점이고 링크의 길이를 L이라 할 때 다음의 다섯 가지 사항으로 구분한다.

① S에 있고 E에는 없는 POI 객체는 항상 증가객체로 분류한다.
② S에 없고 E에만 존재하는 POI 객체는 항상 감소객체로 분류한다.
③ 두 노드에 모두 있고 Dist(S,O) = L + Dist(E,O)이면 항상 감소객체로 분류한다.
④ 두 노드에 모두 있고 Dist(S,O) + L = Dist(E,O)이면 항상 증가객체로 분류한다.
⑤ 그 이외의 경우에는 [Dist(E,O) + L Dist(S,O)] / 2 가 증가객체에서 감소객체로의 전환점이 된다.

구해진 증가 및 감소 리스트를 오름차순으로 정렬한다.

(2) 분할점이 존재하는 범위

분할점이 존재하는 최소점과 최대점은 아래와 같이 정의할 수 있다.

- 최소점A = [min(dec list) max(inc list)] / 2
- 최대점B = [max(dec list) min(inc list)] / 2

이때 A값이 L보다 크면 분할점 없이 kNN은 E의 kNN이며 경로상의 다음 링크로 전진할 수 있다. 그리고 B값이 음수이면 분할점 없이 kNN은 S의 kNN이며 다음 링크로 전진할 수 있다. 따라서 제안한 UBA기법[3]과 같은 효과가 있으면서도 간단한 수식으로 계산된다.

(3) 분할점 및 kNN의 추출

분할점의 추출은 아래와 같은 6단계를 거친다.

① 초기 증가 리스트와 감소 리스트를 kNN 후보집합으로 정의하고 2원 합병 과정을 통해 거리의 오름차순으로 k개의 최근접 객체리스트 kNN를 구한다. 이 때 같은 거리값을 갖는 증가 객체와 감소객체가 있을 경우 감소 객체를 kNN에 포함시키는 것으로 한다. 이는 이동객체의 방향성을 고려한 것이다.
② kNN에 포함되는 감소 리스트의 객체는 그 리스트에서 제외한다(*규칙1*).
③ kNN에서 빠진 증가 리스트의 객체는 kNN 후보집합과 그 리스트에서 제외한다(*규칙2*).
④ 분할점{SPi = [min(dec list) max(inc list)] / 2}를 구한다. 이 때 분할점 계산에 참여한 증가객체는 kNN 후보집합에서 제외되고, 감소객체는 kNN 후보집합에 포함시킨다.
⑤ kNN 후보집합으로부터 kNN을 구한다.
⑥ 2, 3, 4, 5의 과정을 더 이상의 감소 리스트가 존재하거나 없거나 분할점이 최대점 B보다 클 때까지 반복한다.

분할점 및 5NN 추출 과정을 예를 들어 설명하고자 한다. 그림 2는 사용자의 경로중 하나의 링크를 나타낸 것이다.

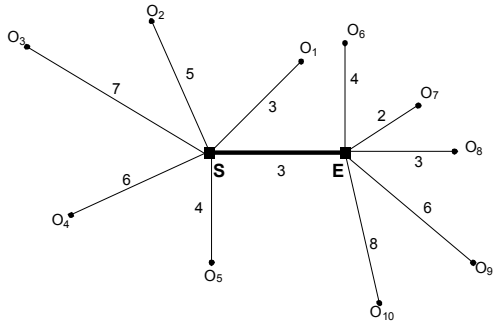

그림 2. 5NN위한 네트워크와 POI

단계1의 결과로서 S노드의 5NN{(O_1,3), (O_5,4), (O_2,5), (O_7,5), (O_4,6)}과 E노드의 5NN{(O_2,2), (O_8,3), (O_6,4), (O_9,6), (O_1,6)}이 사전에 저장된다. 이 때 각 쌍은 POI ID와 거리이다.

그리고 전술한 기법에 의한 증가리스트와 감소리스트를 각각 {(O_1,3), (O_5,4), (O_2,5), (O_4,6)}과 {(O_7,5), (O_8,6), (O_6,7), (O_9,9)}이다.

처리단계에 의해 분할점 및 5NN 추출과정에서 초기 5NN은 {(O_1,3), (O_5,4), (O_2,5), (O_7,5), (O_8,6)}이다.

초기 5NN이 정해지면 (규칙1)에 의해 O_7, O_8이 새로운 kNN 후보집합에 추가되며, 감소 리스트에서 제외된다. 그리고 (규칙2)에 의해 증가 리스트에 있던 O_4는 리스트에서 제거되며, kNN 후보 집합에서도 제거된다. 그리고 kNN 후보집합으로부터 5개의 kNN을 구한다. 표 1은 새로운 증감리스트를 나타낸 것이다.

표 1. 2단계 증가리스트와 감소리스트 상태

Inc List		Dec List	
POI ID	Dist	POI ID	Dist
1	3	6	7
5	4	9	9
2	5		

다음 과정은 {SP_i = [min(dec list) max(inc list)] / 2}에 의해 SP_1이 1(= [7-5]/2)이 되며, SP_2가 2.5(= [9-4]/2)가 된다.

3.3 성능 비교 분석

여기서는 기존의 CkNN의 연구인 IE&UBA 방식[3]과 CkNN_NO를 비교 분석하고자 한다. 그림 1의 도로 네트워크를 기준으로 IE&UBA 방식은 표 2와 같이 6개의 분할점이 발생하며, 각 분할점마다 분할점으로부터 각 객체간의 거리 및 방향을 계산해야 한다. 또한 후보 집합에 대한 새로운 정렬이 필요하다. 그러나 CkNN_NO 기법은 무순위 최근접 객체를 구하는 것으로서 2개의 분할점(1, 3)이 생성되며, 분할점으로부터 각 객체간의 거리 및 방향 계산이 불필요한 장점이 있다. 또한 2원 합병기법으로 한번 정렬된 후보 집합에 대한 부분정렬만이 필요하다.

표 2. IE 방식에 의한 각 단계별 분할점 생성

Split Point	Distance to A	Candidate Set
p_1	0.5	$\uparrow (r_1, 3.5), \downarrow (r_3, 6.5), \uparrow (r_2, 6.5), \downarrow (r_4, 7.5), \downarrow (r_5, 8.5)$
p_2	1.0	$\uparrow (r_1, 4), \downarrow (r_3, 6), \downarrow (r_4, 7), \uparrow (r_2, 7), \downarrow (r_5, 8)$
p_3	1.5	$\uparrow (r_1, 4.5), \downarrow (r_3, 5.5), \downarrow (r_4, 6.5), \downarrow (r_5, 7.5), \uparrow (r_2, 7.5)$
p_4	2.0	$\downarrow (r_3, 5), \uparrow (r_1, 5), \downarrow (r_4, 6), \downarrow (r_5, 7), \uparrow (r_2, 8)$
p_5	2.5	$\downarrow (r_3, 4.5), \downarrow (r_4, 5.5), \uparrow (r_1, 5.5), \downarrow (r_5, 6.5), \uparrow (r_2, 8.5)$
p_6	3	$\downarrow (r_3, 4), \downarrow (r_4, 5), \downarrow (r_5, 6), \uparrow (r_1, 6), \uparrow (r_2, 9)$

그리고 전술한 UBA 기법은 첫 분할점 이후에 분할점 계산 생략 조건이 발생해도 IE 계산법에 의해 알고리즘이 진행되지만, 본 논문에서 제안한 기법은 분할점이 링크의 길이보다 크면 그 링크의 이후 경로에 대해 분할점 계산이 생략되므로 UBA가 검지하지 못하는 것도 처리할 수 있는 장점이 있다.

Ⅴ. 결 론

이 논문에서는 도로 네트워크 데이터베이스에서 이동 중인 객체의 경로와 정적 객체들 간의 네트워크 거리를 기반으로 무순위 k 최근접 객체를 효율적으로 검출하기 위한 기법인 CkNN_NO를 제시하였다. 제안한 기법은 도로네트워크의 각 노드에서 k개의 최근접 객체를 사전에 계산하여 저장하며, 주어진 경로에 대해 kNN이 변경되는 지점인 분할점을 간단한 계산에 의해 추출하는 효율적인 방법이다.

이 기법은 무순위 객체를 검출하므로 보다 적은 분할점을 생성하며, 알고리즘수행 도중에는 거리와 방향계산을 하지 않아 수행 시간을 단축할 수 있는 장점이 있으며, 하나의 링크에 대해 분할점이 링크의 길이보다 크면, 언제든지 추가 분할점 계산이 생략될 수 있어 보다 효율적인 기법이다.

참고문헌

[1] X. Huang, C.S. Jensen, S. Saltenis, “The Islands Approach to Nearest Neighbor Querying in Spatial Networks”, Prof. of Int’l Symp. On Spatial and Temporal Databases, SSTD, pp. 73-90, 2005

[2] M. Kolahdouzan, C. Shahabi, “Voronoi-Based K-Nearest Neighbor Search for Spatial Network Databases”, Proc. Of Int’l Conf. on Very Large Databases, VLDB, pp. 840-851, 2004

[3] M. Kolahdouzan, C. Shahabi, “Continuous K Nearest Neighbor Queries in Spatial Network Databases”, Prof. of STDBM, 2004

[4] K Mouratidis, M.L. Yiu, D. Papadias, N. Mamoulis, “Continuous Nearest Neighbor Monitoring in Road Networks”, VLDB, pp. 43-54, 2006

[5] N. Roussopoulos, S. Kelley, F. Vincent, “Nearest Neighbor Queries”, SIGMOD, 1995

[6] Y. Tao, D. Papadias, “Time Parameterized Queries in Spatio-Temporal Databases”, SIGMOD, 2002

[7] 이상철, 김상욱, “도로 네트워크 데이터베이스에서 근사 색인을 이용한 k-최근접 질의 처리”, 한국정보과학회 논문지 : 데이터베이스, 제 35권, 제 5호, pp. 447-458, 2008

[8] P. Sistla, O. Wolfson, S. Chamberlain, D. S, “Modeling and Querying Moving Objects”, IEEE ICDE 1997

[9] Y. Tao, D. Papadias, Q. Shen, “Continuous Nearest Neighbor Search”, VLDB, 2002