도로 네트워크에서 k-최근접 이웃 검색을 위한 최단 경로 탐색

신성현, 이상철, 김상욱
한양대학교 전자컴퓨터통신공학과
e-mail: wook@hanyang.ac.kr

Shortest Path Finding for k-Nearest Neighbor Searching in Road Network Databases

Sung-Hyun Shin, Sang-Chul Lee, Sang-Wook Kim
Dept of Electronics and Computer Engineering, Hanyang University

요 약

본 논문에서는 최단 경로 탐색 및 거리 계산의 필요성을 가지고 근사 인덱싱 방법의 후처리 부분을 제안한다. 근사 인덱싱 방법이란 오프라인에서 네트워크 공간상의 객체들을 유클리드 공간 상의 절대 좌표로 사상하여 인덱싱한 후, k-최근접 이웃 질의를 처리하는 방법이다. 그러나 기존 연구는 질의 점으로부터 각 정적 객체까지의 경로를 탐색해주지 않을 뿐만 아니라 착오 기각이 발생한다. 따라서 본 논문에서는 질의 점으로부터 k개의 정적 객체까지의 경로를 효과적으로 탐색할 수 있는 방법을 제안한다. 또한, 이 방법을 통하여 착오 기각 역시 완화시킬 수 있는 방법을 제안한다. 실험을 통하여 제안하는 방법이 기존 경로 탐색 기법들에 비해 노드 탐색 횟수 및 실행 성능이 크게 향상시킨 것으로 나타났다.

1. 서론

이동 통신망이나 GPS 등과 같은 위치 탐색 및 서비스 기술이 발전함에 따라, 최근에는 이동 객체의 위치 기반 서비스를 지원하는 응용 분야에 대한 관심이 높아졌다[1, 2]. 이동 객체의 위치 기반 서비스는 이동 객체와 주유소 등과 같은 정적 객체의 위치 정보를 효율적으로 검색하기 위한 k-최근접 이웃 질의를 요구한다[3, 4, 5].

도로 네트워크에서 k-최근접 질의 이웃 질의를 처리하는 방법들이 연구되었다. 우선, Papadias 등[6]은 유클리드 거리가 네트워크 거리보다 작거나 같은 성질을 이용한 IER 방법과 모든 도로 세그먼트를 점진적으로 확장하면서 정적 객체 존재 유무를 검색하는 INE 방법을 제안하였다. 그러나 IER 방법은 여러 번의 시행착오가 필요하며, INE 방법은 다수의 디스크 접근이 필요한 문제점이 있다.

Papadias 등이 제안한 IER 방법과 INE 방법의 성능 저하를 개선하기 위하여 Kolahdouzan 등[7]이 VN^3 방법을 제안하였으며, Lee 등[8]이 정적 객체들간의 네트워크 거리에 기반하여 근사 인덱싱하는 방법을 제안하였다. 그러나 VN^3 방법은 저장 오버헤드가 심하며[9], 근사 인덱싱 방법은 VN^3 방법에 비해 적은 저장 오버헤드를 가지지만 착오 기각이 발생한다. 그리고 두 가지 방법 모두 질의 점과 가까운 k개의 정적 객체만 찾아줄 뿐 경로를 탐색해 주지 않는 문제를 가지고 있다.

따라서 본 논문에서는 k-최근접 이웃 질의 결과에 대하여 최단 경로를 효과적으로 탐색하는 방법을 제안한다. 우선, 근사 인덱싱 방법을 이용하여 k개의 최근접 정적 객체들의 위치들을 검색한다. 그 다음, 불필요한 경로 탐색을 최소화 하면서, 탐색하면서 발생한 정보를 유지하여 다수의 정적 객체에 대해 효과적인 경로 탐색을 한다. 실험 결과 기존 경로 탐색 기법들에 비해 제안하는 방법의 CPU 수행 시간 및 디스크 접근 시간이 감소하였다.

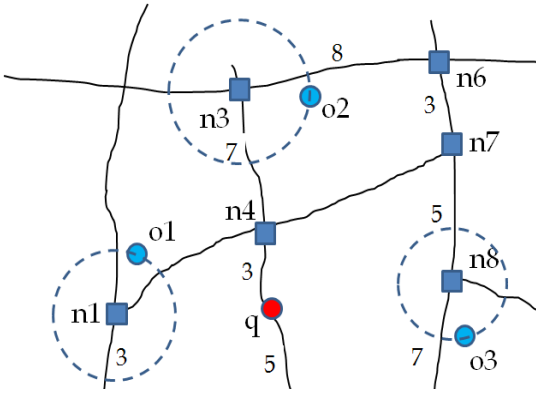
2. k-최근접 이웃 질의

2.1 기본 전략

본 논문에서 제안하는 k-최근접 객체들의 최단 경로 탐색 방법은 크게 다음과 같이 세 단계의 작업으로 구성된다. 1) 근사 인덱싱 방법을 통하여 질의 점에서 네트워크 상으로 가까운 k개의 정적 객체를 검색한다. 그 다음, 2) 질의 점부터 근사 인덱싱 방법으로 가장 가깝다고 판정된 정적 객체까지의 경로를 A* 알고리즘[10, 11]을 이용하여 탐색한다. 탐색하면서 방문한 각 노드까지의 거리 및 경로 정보를 저장하였다가 다음 객체까지의 경로를 탐색하는데 이용한다. 마지막으로 3) 정적 객체에서 가까운 노드로부터 정적 객체까지 거리를 계산하며, 이를 통하여 질의 점으로부터 정적 객체까지의 실제 거리를 산출한다.

그림 1은 질의 점으로부터 k-최근접 객체들을 탐색하는 예를 나타낸다. *o1, o2, o3*는 근사 인덱싱으로 검색된 정적 객체를 나타내며, 각 정적 객체로부터 가장 가까운 노드는 *n1, n3, n8*이다. 질의 점 *q*가 주어졌을 때 *o1, o2, o3*

의 위치 정보와 해당 객체와 가장 가까운 노드 정보가 주어진다. 그 다음, 각 객체까지의 경로를 탐색하는데 가장 가까운 정적 객체 *o1*부터 탐색한다. 이때, 정적 객체 o1은 네트워크 위에 존재하지 않을 수 있음으로 정적 객체 o1과 가장 가까운 노드인 *n1*까지의 경로를 계산한다. 또한, 질의 점 *q*로부터 도로 세그먼트를 확장하여 *n1*에 도달할 때까지 *n3, n7*을 탐색하게 되는데 질의 점 *q*로부터 *n1*까지의 경로 및 거리 정보뿐만 아니라 *n3, n7*까지의 경로 및 거리 정보도 힙에 저장한다. 그리고 두 번째로 가까운 정적 객체인 *o2*까지의 경로를 탐색한다. 질의 점으로부터 *n3*까지 탐색한 정보가 힙에 있기 때문에 해당 정보를 로드(load)하여 경로 및 거리를 반환한다. 세 번째로 가까운 정적 객체인 *o3*까지 경로를 탐색하는데 있어서 *n7*까지의 경로가 이미 탐색되어 힙에 저장되어 있기 때문에 *n7*에서 *n8*까지 도로 세그먼트를 확장하여 경로를 탐색한다. 그리고 질의 점부터 *n8*까지의 경로 및 거리정보를 힙에 저장하는데, 그 이유는 다음 정적 객체까지의 경로 탐색에 해당 정보를 이용하기 위해서이다. 근사 인덱싱에서 검색한 모든 정적 객체와 가장 가까운 노드까지의 경로를 탐색한 후, 해당 노드들로부터 실제 각 정적 객체까지의 거리를 계산한다. 이를 통하여 질의 점 *q*로부터 각 객체까지의 실제 거리를 산출하게 되며, 이 거리를 기준으로 각 객체를 재 정렬한다.



(그림 1) 도로 네트워크의 예.

그림 2는 제안하는 방법을 이용하여 기존 방법에서 발생한 착오 기각을 완화 할 수 있는 방법을 나타낸다. 우선, *k*보다 큰 수로 *k*-최근접 이웃 질의를 처리한다. 기존 근사 인덱싱 방법은 FastMap[12]으로 매핑하는 과정에서 정적 객체간 거리의 오차가 발생한다. 그러나 그 오차의 정도가 작기 때문에 *k*보다 조금 큰 수로 검색할 경우 정확하게 검색할 수 있다[8]. 그 다음, 검색된 정적 객체에 대해 경로를 탐색하고 실제 거리를 산출하여 *k*개의 정적 객체와 경로를 반환한다. 이는 *k*개의 정적 객체에 대한 경로를 찾는 비용을 조금 희생하여 *k+a*개의 정적 객체의 경로를 탐색함으로 적은 비용으로 착오기각을 완화시킬 수 있다.

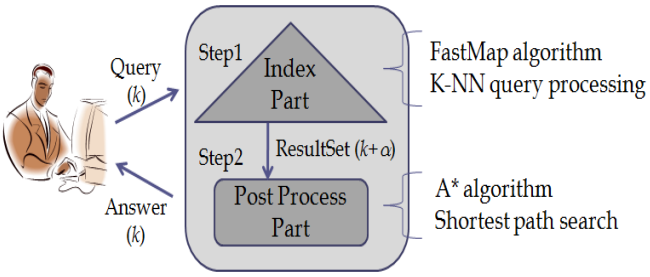


그림 2. 제안하는 방법을 이용하여 *k*-최근접 이웃 질의 처리 과정.

2.2 최단 경로 탐색

A* 알고리즘 질의 점으로부터 임의의 정적 객체까지의 최단 경로를 탐색하기 위해 A* 알고리즘을 이용할 수 있다. 그러나 질의 점과 정적 객체는 노드에 위치하지 않을 가능성이 크다. 따라서 질의 점에서 인접한 노드 u에서 정적 객체에서 인접한 노드 v까지 경로를 탐색한다. 이 두 정점 u와 v 간의 네트워크 거리 및 유클리드 거리를 표현하면 다음과 같다. 노드 u는 (u,v)의 시작 노드이고, v는 (u,v)의 끝 정점이라고 할 때, 네트워크 거리는 $dist_N(u,v)$라 표현하며, 유클리드 거리는 $dist_E(u,v)$로 표현한다. 도로 네트워크상에서 A* 알고리즘을 이용한 최단 경로 계산식을 다음 공식과 같다.

$$dist(u,v) = dist_N(u,w) + dist_E(w,v) \quad (1)$$

A* 알고리즘은 최단 경로를 효과적으로 탐색하기 위하여 *분기 한정(brand-and-bound) 방법*을 이용한다. 분기 한정 방법은 주어진 노드와 연결된 여러 노드들 사이에 경로를 공식 (1)에 의한 최단 경로 계산식을 이용하여 임의의 정적 객체까지 거리보다 큰 경우를 계산하여 불필요한 경로 탐색하지 않는 방법이다. 이러한 분기 한정 방법은 도로 네트워크에서 최적의 인접한 노드를 경유하여 최단 경로를 찾는 것은 주어진 두 노드 u와 v를 연결하는 경로들 $w_i(1 \le i \le n)$ 중에 $dist(u,w_i)$의 비용이 가장 작은 경로를 탐색하는 것을 의미하며, 주어진 노드와 연결된 여러 노드들 중에 최적 노드를 탐색하기 위해 공식 (2)를 제시한다.

$$dist^*(u,v) = \min_{1 \le i \le n}(dist_N(u,w_i) + dist_E(w_i,v)) \quad (2)$$

제안하는 방법은 목표 노드에 연결된 인접 노드들의 거리를 계산하여 최적 노드를 선별함으로써, 노드 탐색에 사용되는 메모리 사용량을 감소시킬 수 있다.

k개의 노드까지의 효과적인 경로 탐색 *k*-최근접 이웃 질의의 결과는 순서가 있는 *k*개의 정적 객체의 위치와 인접한 노드 정보이다. 이 노드 정보를 이용하여 위에서 설명한 A* 알고리즘을 적용하여 질의 점으로부터 각 정적

객체까지의 경로를 탐색한다. 우선, 질의 점에서 가장 가까운 정적 객체까지의 경로를 A* 알고리즘을 이용하여 탐색한다. 이미 탐색한 경로를 다시 탐색하지 않기 위해 탐색하면서 발생한 중간 정보인 $dist_N(u,w_i)$와 u에서부터 w_i까지의 경로를 힙에 저장한다. 그 다음, 질의 점에서 두 번째로 가까운 정적 객체까지의 경로를 A* 알고리즘을 이용하여 탐색한다. 이때, 기존에 힙에 저장된 정보를 사용하여 탐색하며, 필요할 경우 새로운 경로를 탐색하고 중간 정보 $dist_N(u,w_i)$를 힙에 저장한다. 위와 같은 방법으로 모든 정적 객체까지의 경로를 탐색한다. 본 논문에서 제안하는 한 노드에서 다수의 노드까지 효과적으로 탐색하는데 적합하며, 이는 실험을 통하여 A* 알고리즘을 독립적으로 k번 수행했을 때보다 우수한 성능을 검증하였다.

실제 정적 객체까지의 경로 탐색 실제 도로 세그먼트 상의 정적 객체들은 도로 세그먼트 상에 존재하는 것이 아닌, 노드에 인접하게 존재한다. 그러므로 도로 세그먼트와 정적 객체 간의 최소 거리 탐색이 필요하다. 만일 두 노드 u와 v 사이에 정적 객체 o_i가 주어지면, 두 노드와 정적 객체 사이의 최단 거리 함수는 다음 공식 (3)으로 정의된다.

$$mindist(u|v,o_i) = \min(dist_E(u,o_i), dist_E(v,o_i)) \quad (3)$$

공식 (1~3)에 의해, 제시한 방법은 앞서 제시한 최단 경로 탐색으로 선정된 특정 노드와 사전에 예측한 정적 객체 사이에 최소 거리를 계산함으로써, 정적 객체와의 최단 경로 탐색과 정적 객체의 근접한 거리를 계산할 수 있다.

3. 성능 분석 및 평가

3.1 실험 데이터 및 실험 환경

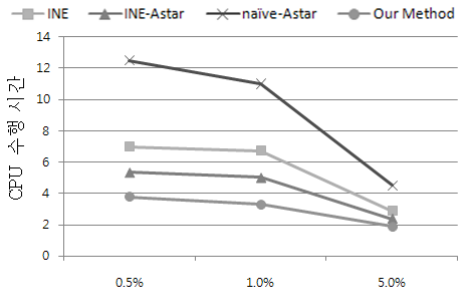
본 실험을 위한 실제 도로 네트워크 데이터로 ORN(Oldenburg Road Network)와 CRN(California Road Network)를 사용한다[13]. 정적 객체의 수는 ORN와 CRN에서 도로 세그먼트 수의 0.5%, 1%, 5%로 변화시켜 생성하였으며, 이들의 위치는 임의로 분산시켜 생성하였다.

성능 평가를 한 비교 대상은 다음과 같다. 실험 대상 중 INE 방법은 k개의 정적 객체를 찾는 방법 및 최단 경로를 탐색하는 방법을 제공하므로 비교 대상으로 추가하여 실험을 수행하였다.

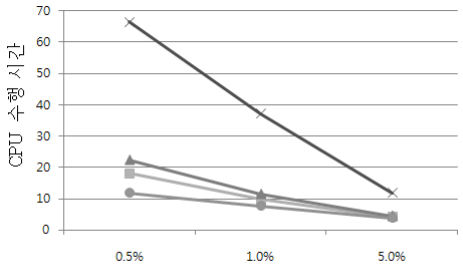
- INE: 기존 INE 방법으로 경로를 탐색하며 k개의 정적 객체를 찾는 방법
- INE-Astar: 미리 주어진 k개의 정적 객체에 대해 INE방법으로 경로를 탐색하는 방법
- Naive-Astar: 기존 A* 알고리즘만을 이용한 k개의 정적 객체에 대한 경로를 탐색하는 방법
- Our Method: 본 논문에서 제안하는 방법을 이용하여 k개의 정적 객체에 대한 경로를 탐색하는 방법

3.2 성능 실험 결과

그림 3는 k-최근접 이웃 질의에 대한 k는 10개로 고정하고, 도로 네트워크 내의 정적 객체의 수에 대하여 ORN과 CRN에서 각각 도로 세그먼트 수에 따른 정적 객체의 수를 0.5%, 1%, 5%로 변경하면서 성능을 측정한 결과이다. 그림을 보면, 모든 경우에 있어서 제안한 방법이 다른 방법에 비해 성능이 우수하다. 그 이유는 Naive-Astar의 경우 k의 수 만큼 정적 객체들의 거리를 반복적으로 계산하여 처리하기 때문이다. 반면에, 제안한 방법과 INE-Astar는 효과적으로 다수의 정적 객체까지의 경로를 탐색하기 때문에 정적 객체의 수가 증가하거나 감소하더라도 일정한 성능을 보였다. 또한, 제안한 방법이 INE-Astar 보다 성능이 우수한 이유는 INE-Astar는 인접한 모든 도로 세그먼트들을 확장하며 각 정적 객체의 경로를 탐색하는 반면에, 제안한 방법은 불필요한 도로 세그먼트의 확장을 줄였기 때문이다.



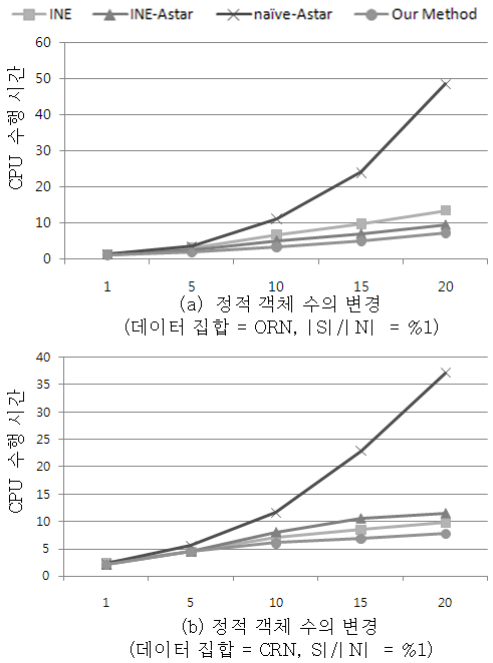
(a) 전체 정적 객체의 비율 변경
(데이터 집합 = ORN, k = 10)



(b) 전체정적 객체의 비율 변경
(데이터 집합 = CRN, k = 20)

(그림 3) 전체 정적 객체의 비율 변경에 대한 CPU 수행시간 비교 실험 결과.

그림 4는 도로 세그먼트에 따른 전체 정적 객체의 비율을 1%로 고정하고, k의 수를 증가시키면서 CPU 수행 시간을 측정한 실험 결과이다. 그림을 보면, 제안한 방법이 탐색해야 될 k의 수가 증가할수록 기존의 다른 방법에 비해 경로 탐색 비용을 크게 감소시켰다. 그 이유는 모든 경로 탐색을 위하여 불필요한 노드 방문 수를 줄이고, 하나의 정적 객체까지 경로 탐색하는 과정에서 발생하는 중간 결과들을 계속 유지 하여 다음 정적 객체까지 경로 탐색하는데 이용하기 때문이다. 따라서 제안하는 방법은 k의 수가 증가할수록 기존 다른 방법들에 비해 성능의 차는 더 커질 것으로 예측할 수 있다.



(그림 4) 정적 객체의 수 변경에 대한 CPU 수행 시간 비교 실험 결과.

4. 결론

본 논문에서는 k-최근접 이웃 질의에 대해 최단 경로 탐색 및 거리를 측정하는 방법을 제안하였다. 우선 기존 근사 인덱싱 방법을 통하여 k개의 정적 객체를 찾고 각 정적 객체까지의 경로를 탐색한다. 탐색하는 과정에서 분기 한정 방법을 사용한 A* 알고리즘을 적용하기 때문에 효과적으로 질의 점으로부터 다수의 정적 객체까지의 경로를 탐색할 수 있다. 실험을 통하여 본 논문에서 제안하는 방법이 다른 방법에 비해 우수함을 보였다. 기존 근사 인덱싱 방법에서 k보다 큰 수의 정적 객체들을 검색하여, 본 논문에서 제안하는 기법을 적용하여 질의 점으로부터 가장 가까운 k개의 정적 객체와 각 객체의 경로를 탐색한다면 착오 기각 문제가 크게 완화될 것이라 기대한다.

감사의 글

이 논문은 2007년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원(KRF-2007-314-D00221)과 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원(IITA-2009-C1090-0902-0040)으로 수행되었습니다.

참고 문헌

[1] Wu, S. and Wu, K., “Effective Location-Based Services with Dynamic Data Management in Mobile Environments,” Wireless Networks, Vol. 12, No. 3, pp. 369-381, 2006.

[2] Wolfson, O. et al., "Moving Object Databases: Issues and Solutions," In *Proc. Int'l. Conf. on Scientific and Statistical Database Management*, SSDBM, pp. 111-112, 1998.

[3] Jensen, C. S., Friis-Christensen, A., Pedersen, T. B., Pfoser, D., Saltenis, S., and Tryfona, N., “Location-based services: A database perspective,” In *Proc. of the 8th Scandinavian Research Conference on Geographical Information Science,* ScanGIS, pp. 59-68, 2001.

[4] Hjaltason, G. and Samet, H., “Distance Browsing in Spatial Databases,” ACM Trans. on Database Systems, TODS, Vol. 24, No. 2, pp. 265-318, 1999.

[5] Faloutsos, C., Ranganathan, M., Manolopoulos, Y., " Fast Subsequence Matching in Time-Series Databases," In *Proc. ACM Int'l Conf. on Management of Data,* ACM SIGMOD, 1994.

[6] Papadias, D., Zhang, J., Mamoulis, N., and Tao, Y., “Query Processing in Spatial Network Databases,” In *Proc. Int'l. Conf. on Very Large Data Bases*, VLDB, pp. 802-813, 2003.

[7] Kolahdouzan, M. and Shahabi, C., “Voronoi-Based K-Nearest Neighbor Search for Spatial Network Databases,” In *Proc. Int'l. Conf. on Very Large Data Bases*, VLDB, pp.840-851, 2004.

[8] Lee, S.-C., Kim, S.-W., Lee, J., and Yoo, J. S., “Approximate Indexing in Road Network Databases,” In *Proc. ACM Int'l Symp. on Applied Computing,* ACM SAC, pp.1568-1572, Hawaii, USA, Mar. 2009.

[9] Huang, X., Jensen, C. S., and Saltenis, S., "The Islands Approach to Nearest Neighbor Querying in Spatial Networks," In *Proc. Int'l. Symp. on Spatial and Temporal Databases*, SSTD, pp. 73-90, 2005.

[10] Hart, P. E., Nilsson, N. J., and Raphael, B., "A Formal Basis for the Heuristic Determination of Minimum Cost Paths in Graphs," IEEE Trans. on Systems Science and Cybernetics, Vol. SSC-4, No. 2, pp. 100-107, July 1968.

[11] Kung, R., Hanson, E., Ioannidis, Y., Sellis, T., Shapiro, L., and Stonebraker, M., “Heuristic search in data base systmes,” Expert Database Systems, 1986.

[12] Faloutsos, C. and Lin, K., “FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets,” In *Proc. ACM Int'l Conf. on Management of Data*, ACM SIGMOD, pp.163-174, 1995

[13] The R-tree Portal, http://www.rtreeportal.org/