Materialization 기법을 이용한 공간 네트워크 DB에서의 효율적인 범위 질의 처리 알고리즘의 설계 및 성능 평가

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

Design and Evaluation of Efficient Query Processing Algorithm using Materialization Technique for Spatial Network Database

Yongki Kim° Yongguk Kim, Jaewoo Chang
Dept. of Computer engineering, ChonBuk National University

요 약

지난 20년 동안 공간 데이터베이스에서 유클리디언(Euclidean) 공간 기반의 연구가 활발히 진행되어 왔으며, 최근에는 실제 응용에 적용하기 위해 도로 네트워크 기반의 질의 처리 알고리즘의 연구가 활발히 수행중이다. 본 논문에서는 도로 네트워크에서 제시된 기존 범위 질의처리 알고리즘의 성능을 향상시키기 위해, Materialization 기법을 이용한 효율적인 범위 질의 처리 알고리즘을 제안한다. 아울러 성능 평가를 통하여 Materialization 기법을 이용한 제안하는 알고리즘이 기존 알고리즘보다 검색 성능이 우수함을 보인다.

1. 서 론

지난 20년 동안 공간 데이터베이스(Spatial Databases) 분야에서 활발한 연구가 진행되어 왔다[1]. 그러나, 대부분의 연구는 유클리디언(Euclidean) 공간을 가정하고 연구가 수행되었기 때문에, 유클리디언 공간에서 질의와 가장 가까운 POI(Point Of Interest)가 실제 도로 네트워크 상에서는 다른 POI보다 거리가 더 멀 수 있다. 따라서 LBS(Location Based Service) 및 텔레매틱스(Telematics) 응용을 효과적으로 지원하기 위해, 최근에는 유클리디언 공간 대신 실제 도로나 철도와 같은 공간 네트워크(Spatial Network) 상에서 질의처리 알고리즘에 대한 연구가 활발히 수행중이다[2,3,4].

본 논문에서는 도로 네트워크에서 제시된 기존 범위 질의처리 알고리즘의 성능을 향상시키기 위해, 3개의 새로운 범위 질의처리 알고리즘을 제안한다. 제안하는 3개의 질의처리 알고리즘은 기존에 제시된 알고리즘을 Materialization 기법을 이용하여 확장하는 방법들이다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 소개하고, 3장에서는 본 논문에서 제안하는 범위 질의처리 알고리즘을 제시하고, 4장에서는 기존 연구된 알고리즘과 제안하는 알고리즘의 성능평가를 나타낸다. 마지막으로 5장에서는 결론 및 향후연구를 제시한다.

2. 관 련 연 구

공간 네트워크 데이터베이스에서 범위 질의처리 알고리즘을 설계하기 위하여, 가장 먼저 고려할 수 있는 방법은, 기존 유클리디언 공간을 가정하고 만들어진 범위 질의 처리 알고리즘을 직접적으로 확장하는 접근방법이다[4]. 이 접근방법 (RER : Range Euclidean Restriction)은 유클리디언 거리로써 R-tree를 탐색하여 범위 질의를 수행한 후, 범위 내의 에지들을 탐색하는 방법이다. 만약 탐색된 에지상에 결과 POI들이 존재하면 결과 집합에 포함하고, 그렇지 않다면 결과 집합에서 제외시킨다. 둘째, 공간 네트워크 데이터베이스에서 범위 질의처리 알고리즘으로 네트워크의 특성을 이용한 네트워크 확장 방법이 존재한다[4]. 이 접근방법(RNE : Range Network Expansion)은 질의가 속한 에지의 MBR(Minimum Bounding Rectangle)을 탐색하여, 그 MBR을 통하여 각각의 자식 엔티티로 내려가면서 범위 내의 모든 POI 를 찾을 때까지 계속하여 탐색하는 방법이다.

3. 공간 네트워크 범위 질의처리 알고리즘

공간 네트워크 데이터베이스에서 기존 범위 질의처리 알고리즘은 (RER,RNE) 다음과 같은 단점들이 존재한다. 먼저 RER 방법은 유클리디언 공간에서 범위 질의를 수행하여 범위 내의 POI를 탐색하고, 아울러 결과 POI들의 실제 네트워크 거리를 계산하여 범위 내에 존재하는 지 확인해야 하는 단점이 존재한다. 둘째, RNE 방법은 범위 내의 모든 MBR을 탐색해야 하기 때문에 전체의 트리를 탐색해야 한다는 단점이 존재한다. 이를 극복하기 위해 네트워크 내의 모든 노드들 간의 거리를 미리 계산하여, 질의 처리 시 사용하면 우수한 검색 성능을 획득할 수 있다. 이것이 가능한 이유는 공간 네트워크상의 모든 노드는 고정되어 변하지 않기 때문이다. 한편 이를 위해서는 모든 노드간의 거리를 보조기억장치에 저장하는 것이 필요하며, 약 20만개의 노드를 기준으로 하였을 때, 160GB (=200K*200K*4바이트)가 요구된다. 현재 Maxter, Western Digital, Segate 사에서는 150-300GB

본 연구는 대학 IT 연구센터 육성지원사업의 연구결과로 수행되었음

용량의 하드 디스크를 제공하고 있으며, LaCie 사에서는 최근 1.6TB 용량의 외부 하드 디스크를 출시하였다[5]. 따라서 모든 노드간의 거리를 미리 계산하여 저장하는, 즉 materialization 접근 방법을 사용하는 것이 가능하다 [6].

3.1 제안하는 알고리즘 1

Materialization 기법을 이용한 가장 간단한 범위 질의 처리 알고리즘은, R-Tree를 탐색하여 결과 POI를 구하고, 이들의 실제 거리 계산을 위해 Materialization 기법을 사용하는 방법이다. 이 방법은 실제 거리를 계산하는데 있어, 질의점을 포함하고 있는 에지와 POI를 포함하고 있는 에지를 바로 알 수 있기 때문에, 네트워크 거리 계산 알고리즘에 근거하여 실제거리를 쉽게 계산할 수 있는 장점이 있다. 제안하는 알고리즘 1은 (그림 1)과 같으며, 알고리즘 1이 이용하는 네트워크 거리 계산 알고리즘은 (그림 2)와 같다.

```
materialization 기반 범위 질의 처리 알고리즘 (q,r)
/* q 는 질의 점, r 은 질의 범위 */
1. Euclidean 범위 질의 결과셋을 얻는다.
2. q가 속해있는 에지를 구성하고 있는 노드를 노드 셋에 저장하고, 그 노드로부터 시작하는 materialization 파일을 가져온다.
3. 각각의 범위 질의 결과셋에 대하여 실제 거리를 측정한다.
4. 단계3에서 계산한 POI의 실제 측정한 거리가 범위 r보다 작다면 결과 셋에 넣는다.
```

그림 1. 제안하는 범위 질의처리 알고리즘 1

```
거리 계산 알고리즘 (q,p)
/* q 는 질의 점, p 는 POI */
1. q를 포함하고 있는 에지 Q(N1,N2)를 구한다.
2. p를 포함하고 있는 에지 P(N3,N4)를 구한다.
3. MINIMUM(Distance (N1,N3), Distance (N1,N4), Distance (N2,N3), Distance (N2,N4))를 결과값으로 반환한다.
```

그림 2. 거리 계산 알고리즘

3.2 제안하는 알고리즘 2

제안하는 알고리즘 1은 R-Tree 전체를 탐색해야 하는 단점이 존재하므로, 트리를 검색하지 않고 네트워크를 확장하면서 범위 내의 POI를 검색하는 좀 더 개선된 알고리즘 2를 제안한다. 제안하는 알고리즘 2는 R-Tree를 탐색하지 않고 질의 점으로부터 범위(r)까지 에지를 확장하는 네트워크 확장 방법이다. 이 방법은 Materialization 기법을 이용하여 질의점에서 가장 가까운 노드(노드와 연결되어 있는 에지)로부터 점차적으로 확장해나가는 방법이다. 만약 확장해야 할 노드가 범위(r)보다 크다면 알고리즘을 종료한다. 제안하는 알고리즘 2는 (그림 3)과 같다.

```
materialization 기반 범위 질의 처리 알고리즘 (q,r)
/* q 는 질의 점, r 은 질의 범위 */
1. q가 속해있는 에지에 걸려있는 POI를 결과 셋에 포함한다.
2. q가 속해있는 에지를 구성하고 있는 노드를 노드 셋에 저장하고, 그 노드로부터 시작하는 materialization 파일을 가져온다.
3. materialization 파일로부터 가장 가까운 노드를 검색한다. 만약, 가장 가까운 노드의 거리가 범위 r보다 크다면 알고리즘을 끝낸다.
4. 가장 가까운 노드와의 연결 에지를 확장한다.
5. 단계 4에서 확장하는 에지에 걸려있는 POI들의 실제 거리를 계산한 후, 범위 r보다 작은 POI들을 결과 셋에 포함한다.
6. 단계 3에서 5까지의 단계를 반복적으로 수행한다.
```

그림 3. 제안하는 범위 질의처리 알고리즘 2

그러나, 이 알고리즘은 노드를 확장함에 있어 모든 에지들을 모두 확장하기 때문에 기존에 확장했는지 체크해야 하는 번거로움이 있다. 또한, 체크를 하지 않는다면 중복하여 에지를 확장해야하는 단점이 존재한다.

3.3 제안하는 알고리즘 3

제안하는 알고리즘 2의 단점을 극복하기 위해, 노드를 스택에 저장하고 최소한의 에지 탐색만을 수행하도록 개선한 알고리즘 3을 제안한다. 제안하는 알고리즘 3은 Materialization 기법을 이용하여 질의점에서 가장 가까운 노드들을 스택에 저장하고, 확장해야 할 노드와 스택에 저장된 노드와의 연결 에지만을 먼저 확장하고, 나머지 연결 에지는 확장 에지 집합에 저장하여 나중에 확장하는 방법이다. 따라서 확장하고자 하는 노드와 연결되어 있는 에지를 무조건 확장하는 제안하는 알고리즘 2의 단점을 보안하여, 최소한의 에지 확장을 수행하는 장점을 지닌다. 제안하는 알고리즘 3은 (그림 4)와 같다.

```
materialization 기반 범위 질의 처리 알고리즘 (q,r)
/* q 는 질의 점, r 은 질의 범위 */
1. q가 속해있는 에지에 걸려있는 POI를 결과 셋에 포함한다.
2. q가 속해있는 에지를 구성하고 있는 노드를 노드 셋에 저장하고, 그 노드로부터 시작하는 materialization 파일을 가져온다.
3. materialization 파일로부터 가장 가까운 노드를 검색한다. 만약, 가장 가까운 노드의 거리가 범위 r보다 크다면 단계 7번을 수행한다.
4. 가장 가까운 노드와 이미 노드 셋에 저장된 노드와의 연결 에지는 확장하고, 노드 셋에 저장되어 있지 않은 노드와의 연결 에지는 확장 에지 셋에 저장한다.
5. 단계 4에서 확장하는 에지에 걸려있는 POI들의 실제 거리를 계산한 후, 범위 r보다 작은 POI들을 결과 셋에 포함한다.
```

6. 단계 3에서 5까지의 단계를 반복적으로 수행한다.
7. 확장 에지 셋에 저장된 에지들에 걸려있는 POI들의 실제 거리를 계산한 후, 범위 r보다 작은 POI들을 결과 셋에 포함한다.

그림 4. 제안하는 범위 질의처리 알고리즘 3

4. 성능 평가

성능평가를 위해 본 논문에서 제안하는 3개의 알고리즘을 Memory 2GB, CPU 2.4GHz를 사용하는 Windows Server 2003 Enterprise에서 구현하였다. 지도 데이터는 실제 샌프란시스코만 지도를 사용하였고 [7], 전체 노드 수는 175,343개, 전체 에지수는 223,199개로 구성되어 있다. 아울러, POI 개수는 RunTime21[8] 알고리즘을 사용하여 임의로 생성한 10846개를 사용하였고, 질의점은 임의로 1000개를 주어서 사용하였다. 마지막으로 Materialization 파일은 229GB가 사용되었으며, 알고리즘의 구현을 위해, 이미 개발된 공간 네트워크를 위한 저장/색인 구조를 사용하였다[9]. 제안하는 3개의 알고리즘 및 기존 알고리즘 RER, RNE 의 성능평가 결과는 (표 1)에 나타나며, 제안하는 알고리즘은 각각 Our RA(Range Query Processing Algorithm) 1,2,3 로 표기한다. 여기서 DISK는 DISK I/O 개수를 나타내며, CPU는 CPU Time(sec)을 나타내며, WALL은 전체 시간(sec)을 나타낸다. 범위 r의 단위는 1.5m이다.

표 1. 기존 알고리즘과의 성능평가

		RER	RNE	Our RA1	Our RA2	Our RA3
10	DISK	23	22	13	22.9	10.8
	CPU	0.037	0.078	0.052	0.030	0.016
	WALL	0.091	0.209	0.209	0.061	0.061
50	DISK	942	1773	64	172.6	100.2
	CPU	0.023	0.331	0.269	0.019	0.022
	WALL	0.095	0.934	0.898	0.066	0.064
100	DISK	1554	10773	208	662.3	383.3
	CPU	0.06872	1.097	0.862	0.025	0.025
	WALL	0.20464	2.975	2.849	0.066	0.064
300	DISK	957971	322720	1178	5756.8	3341
	CPU	4.859	14.853	4.277	0.055	0.034
	WALL	9.319	26.831	14.498	0.119	0.097
500	DISK	5744360	1534149	2491	16244	9444
	CPU	58.760	97.345	8.999	0.099	0.056
	WALL	84.899	134.230	31.123	0.267	0.166

(그림 6)은 5가지 알고리즘의 전체 질의 처리 시간을 나타낸다. R-Tree 전체를 검색하는 기존 RNE 기법은 성능이 매우 좋지 않음을 알 수 있다. 또한, 제안하는 알고리즘 1은 RER 기법보다 범위가 4000이상일 때 성능이 좋아짐을 알 수 있다. 제안하는 알고리즘 2와 3은 Materialization 기법을 이용하는 한편 효율적인 네트워크 확장 방법을 사용함에 따라 기존 RER 및 RNE 방법에 비해 매우 효율적임을 알 수 있다. 한편, 제안하는 알고리즘 3은 최소한의 네트워크 확장 방법을 채택함에 따라, 제안하는 알고리즘 2보다 우수함을 알 수 있다.

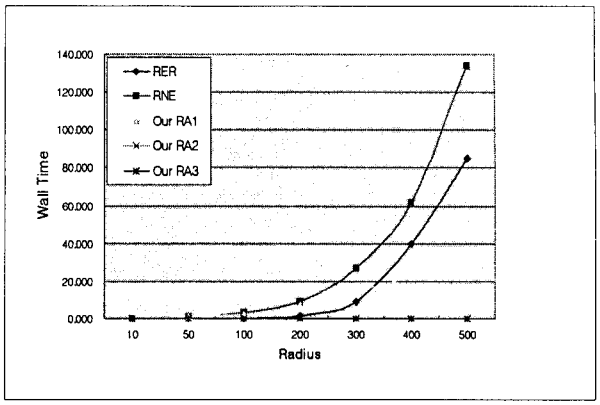

그림 6. 전체 시간 성능 평가

5. 결론 및 향후 연구

본 논문에서는 공간 네트워크 데이터베이스에서 Materialization 기법을 이용하여 보다 효율적인 범위질의 처리 알고리즘을 제안하였다. 성능평가를 통해, 제안하는 알고리즘이 기존 알고리즘 (RER, RNE)보다 성능이 매우 우수함을 보였다. 향후 연구로는 본 논문에서 제안하는 알고리즘을 텔레매틱스(Telematics)나 LBS의 실제응용에 적용하여 성능의 우수성을 검증하는 것이다.

참고문헌

[1] S.Shekhar et al., "Spatial Databases Accomplishments and Research Need", IEEE Tran. on Knowledge and Data Engineering, Vol.11, No. 1, pp45-55, 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] D. Papadias, J. Zhang, N. Mamoulis, and Y. Tao, "Query Processing in Spatial Network Databases" Proc. of VLDB, pp, 802-813, 2003.
[5] http://www.pcworld.com.
[6] R.Agrawal, S,Dar, and H.V.jagadish, "Direct Transitive Closure Algorithms : Design and Performance Evaluation", ACM Tran. on Database Systems, Vol. 15, No. 3, pp427-458, 1990.
[7] http://www.fw-oow.de./institute/japg.
[8] T.Brinkhoff, "A Framework for Generation Network-Based Moving Objects", GeoInformat.
[9] 강홍민, 장재우, "공간 네트워크 데이터베이스를 위한 저장 및 색인 구조의 설계", 한국정보과학회 가을 학술발표논문집, 제 31권 제2호, pp133-136, 2004.