평행한 도로들을 포함하는 L^1 평면상에서의 최단경로 탐색

김재훈 · 김수환

부산외국어대학교

Finding Shortest Paths in L^1 Plane with Parallel Roads

Jae-Hoon Kim · Soo-Hwan Kim

Pusan University of Foreign Studies

E-mail : jhoon@pufs.ac.kr · shkim@pufs.ac.kr

요 약

본 논문에서 우리는 교통 네트워크가 포함된 L_1 평면에서 최단거리 경로를 구하는 알고리즘을 제안한다. 교통 네트워크는 더 빠르게 움직일 수 있는 도로를 나타내는 평행한 선분들로 구성된다. 시작점 s가 주어 질 때, 알고리즘은 최단경로지도(SPM)를 만드는데, 이 지도를 통해서 s 로부터 임의의 평면상의 점 t까지의 최단거리를 O(log n) 시간에 찾을 수 있다. 우리는 이 SPM을 O(nlog n) 시간에 구하는 평면 스윕(plane sweep) 형태의 알고리즘을 설계할 것이다.

ABSTRACT

We present an algorithm for finding shortest paths in the L_1 plane with a transportation network. The transportation network consists of parallel line segments, called highways, through which a movement gets faster. Given a source point s, our algorithm constructs a Shortest Path Map(SPM) such that for any query point t, we can find the length of a shortest path form s to t in O(log n) time. We design a plane sweep-like algorithm computing SPM in O(nlog n) time.

키워드

L^1 평면, 최단경로지도, 교통 네트워크, 평면 스윕

1. 서 론

거리공간상의 임의의 두 점 사이에 최단거리 경로를 찾는 문제는 계산기하학 분야의 중요한 문제들 중의 하나이다. 최단거리 경로는 거리공간에 따라서 다른 모양을 가지고, 본 논문에서는 도로선분들을 포함하는 평면의 거리공간을 다룬다. 여기서 도로선분을 지나는 부분의 경로의 길이는 평면상에서의 길이보다 작다. 다시 말해, 도로선분을 따라서 움직이면 빠른 속도로 움직일 수 있다. 이런 도로선분들은 도시의 지하철 또는 버스 노선 또는 국가의 고속도로들을 모델링한다.

우리는 이러한 새로운 거리공간상에서 고정된 점 s로부터 평면상의 임의의 점사이의 최단거리 경로들을 찾고 싶다. n개의 도로선분이 주어질 때, 이 문제는 s에서의 최단경로지도(SPM)를 만들면 평면상의 임의의 점까지의 최단거리 경로를 O(log n)에 구할 수 있다. 여기서 SPM은 평면 세분(planar subdivision)으로 면(face)상의 모든 점들이 동일한 모양의 최단거리 경로를 가진다. 본 논문에서는 L^1 평면상의 고정된 점이 주어질 때, SPM을 구하는 알고리즘을 연구한다. SPM은 O(n)의 크기를 가지고 있다는 것이 증명되었다[1].

전통적으로 최단거리 경로 찾기 문제는 다각형 모양의 방해물들이 있는 평면상에서 많이 연구되었다[2, 3]. 본 논문에서 다루는 거리공간의 일반화된 모습으로 가중치 값을 가진 임의의 모양의 구역들로 나뉜 평면을 생각할 수 있다. 이러한 평

면에서 각 구역을 지나는 경로 부분의 길이는 가중치와의 곱으로 주어지고 이 길이들의 합이 경로의 길이가 된다. 이 가중치 구역들을 가진 평면에서 최단거리 경로를 구하는 문제가 [4]에서 연구되었다. 저자들은 $O(n^8L)$ 시간에 SPM을 구하는 알고리즘을 제안하였다. 여기서 L은 문제의 입력에 의해 결정되는 상수이다. 또한 L^1 평면의 직선(rectilinear) 구역에 대해서 SPM을 구하는 $O(n(\log n)^{3/2})$ 시간 알고리즘이 존재한다[5]. [6]에서는 기하 최단거리 경로문제의 많은 결과들을 소개하고 있다.

도로선분들을 가진 평면을 다루는 문제는 최근에 Voronoi diagram을 구하는 문제에서 시작되었다[1]. L^1 평면상에 n개의 도로선분과 m개의 점들이 존재할 때, Voronoi diagram은 $O(n\log n + m\log m)$ 시간에 구할 수 있다.

본 논문에서는 L^1 평면상에 n개의 평행한 도로선분이 존재할 때, SPM을 구하는 $O(n\log n)$ 시간 알고리즘을 제안한다.

II. 최단경로 알고리즘

L^1 평면상에 고정된 시작점 s와 x축에 평행한 n개의 도로선분들이 존재한다. 편의상 s=(0, 0)이라고 하고 모든 도로선분은 평면의 일사분면에 존재한다고 가정한다. 또한 각각의 도로선분들은 일정한 속도 v를 가진다. 어떠한 도로선분상의 두 점 x, y사이의 거리는 $d_1(x, y)/v$로 주어진다. 여기서 d_1는 L^1 거리를 나타낸다. 우리는 이러한 새로운 거리공간상에서 SPM을 구하는 알고리즘 A를 제안할 것이다.

우선 고정된 시작점 s에 대해서 하나의 도로선분 h가 주어진 경우를 생각해 보자. 도로선분 h의 왼쪽 끝점을 p라고 하면, 평면상의 점들은 s로부터의 최단거리 경로가 h를 지나는 점과 그렇지 않은 점으로 나뉜다. 그리고 s로부터의 최단거리 경로가 h를 지나는 점들의 집합은 평면상의 연결된 구역을 이루고 이 구역은 그림 1.에서처럼 p에서 위쪽으로 h와 수직을 이루는 선분과 h와 각 θ를 이루는 선분이 경계를 이룬다. 여기서 θ는 경계선위의 점에 대해 s로부터의 거리와 h를 지나는 경로의 거리가 일치하는 관계로부터 계산될 수 있다. θ는 $\theta = \tan\frac{1}{2}(1-\frac{1}{v})$ 와 같이 주어진다.

알고리즘 A는 y축에 평행한 직선 $\mathcal{L}$을 왼쪽에서 오른쪽으로 움직이면서 직선이 지나온 부분에는 정확한 SPM을 구하는 평면 스윕 알고리즘이다. 알고리즘 A는 직선 $\mathcal{L}$을 움직이면서 이벤트 점들을 x값에 의해 정렬해서 E에 저장하고 현재 직선 $\mathcal{L}$이 지난 도로선분을 y값에 의해 정렬해서 H에 저장한다. 그리고 현재 직선 $\mathcal{L}$이 지날 때, 만나는 SPM의 경계선분들을 y축에 대해서 정렬해서 R에 저장한다. 스윕 직선 $\mathcal{L}$이 움직이면서 이벤트 점 p를 만나면 p는 E로부터 주어지고 R로부터 p를 포함하는 SPM의 구역 r을 $O(\log n)$ 시간에 구할 수 있다. 구역 r을 결정하는 도로선분이 스윕 직선 $\mathcal{L}$을 지나면 이 구역을 A-type이라고 하고 지나지 않으면 B-type이라고 한다.

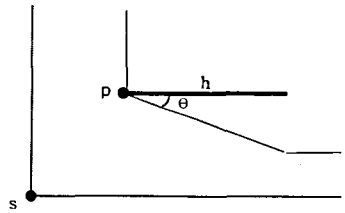

그림 1. h가 결정하는 구역

이제부터 SPM을 구하는 알고리즘 A를 설명한다. 초기에 도로선분의 양 끝점들을 모두 E에 넣는다. 그리고 E에 저장된 점이 없을 때까지 다음의 과정을 반복한다.

E에서 한 점을 출력하고 이를 p로 나타낸다. 우선 R에서 p를 포함하는 구역을 찾고 이를 r로 나타낸다. p는 어떠한 도로선분 h의 시작점 또는 끝점이거나 교차점일 수 있다. 따라서 이 세 경우에 p가 만드는 SPM의 부분을 구분할 수 있다.

Case 1. p가 h의 시작점

p를 포함하는 구역 r의 종류에 따라 두 가지 경우로 나눈다.

먼저 r이 A-type 구역이면 p가 정의하는 어떠한 구역도 만들어지지 않으며 p가 속한 도로선분 h만 비활성 도로로서 H에 넣는다.

r이 B-type 구역이면 p가 속한 도로선분 h가 SPM의 한 구역을 결정한다. r를 결정하는 도로선분을 $\bar{h}$이라고 하고, $y(\bar{h})$와 y(h)를 각각 $\bar{h}$와 h의 y좌표라고 하자. $y(\bar{h}) < y(h)$이면 앞에서 설명한 도로선분이 하나만 있을 때처럼 p에서 아래를 향하고 h와 θ의 각을 이루는 선분 l_1과 위로 향하고 h와 직각을 이루는 선분 l_2, 그리고 이 선분이 r의 경계를 이루는 선분과 만날 때, 경계 선분이 기울기를 가지면, 평행한 선분 l_3, 경계 선분이 평행하면, 기울기 θ를 가지는 선분 l_3가 새롭게 만들어진다. 그리고 이 선분들 l_1, l_2, l_3가 SPM의 새로운 구역의 경계선을 이루고, l_1과 l_3가 구역들의 리스트 R에 새롭게 추가된다. $y(\bar{h}) > y(h)$ 이면 위와 비슷하게 h가 만드는 구역을 생성할 수 있다. 또한 우리는 교차점의 이벤트 점을 가지게 된다. l_1은 $\bar{h}$의 연장선을 반드시 만나게 되고 이 교차점을 이벤트 큐 E에 추가한다. 마지막으로 도로선분 h를 H에 활성 도로로서 추가한다. (그림 2.)

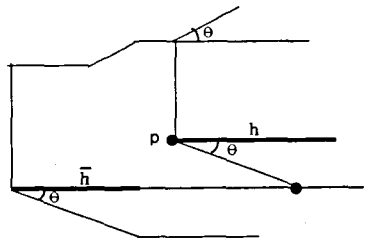

그림 2. p가 h의 시작점

Case 2. p가 h의 끝점

h가 비활성 도로이면 h를 H로부터 제거한다. 결과적으로 h는 SPM의 어떠한 구역도 만들지 못한다.

h가 활성 도로이면 H에 저장된 도로선분 중에서 h에 인접한 두 선분 h_1, h_2를 찾는다. 다시 말해, $y(h_1) < y(h) < y(h_2)$를 만족하고 h에 가장 가까운 y좌표값을 가지는 두 선분 h_1, h_2를 찾는다. h_1이 비활성 도로이면 h_1이 정의하는 새로운 SPM의 구역이 만들어진다. x(p)가 p의 x좌표라고 할 때, p'이 직선 x=x(p)와 도로선분 h_1과의 교점이라고 하면 p'에서 시작하여 h_1과 θ의 각을 이루고 위쪽을 향하는 선분 l_1과 h_1과 직각으로 아래쪽을 향하는 선분 l_2, 이 선분은 h가 정의하고 있는 구역 r의 경계와 만나서 역시 θ의 각을 이루고 아래로 향하는 선분 l_3를 만든다. 이때 l_1, l_2, l_3는 h_1이 정의하는 구역의 경계를 이루고 l_1, l_3는 R에 추가된다. 이때, l_1과 h의 연장선과의 교점을 교차점 이벤트로서 E에 추가한다. 그리고 h_1을 활성도로로 만든다. (그림 3.) h_1이 활성 도로이면 h_1은 h가 정의한 구역 r과 인접한 구역 r'을 정의하고 있다. 따라서 이 경우에 r과 r'과의 경계선에 변화가 생긴다. 현재 r'이 A-type이면 경계선은 수평선분에서 위쪽으로 θ 기울어진 선분으로 바뀌고 r'이 B-type이면 경계선은 기울어진 선분에서 수평선분으로 바뀐다. (그림 4.) h_2에 대해서는 위의 경우와 대칭적으로 비슷하게 구역을 정의한다.

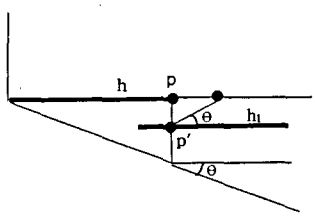

그림 3. p가 h의 끝점

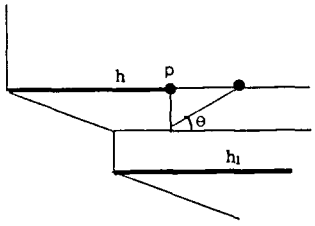

그림 4. p가 h의 끝점

Case 3. p가 교차점

p가 r의 경계를 지난다고 하면, 이 경계를 공유하는 구역 r'이 존재한다. 일반성의 손실 없이 r'의 기울어진 경계선과 r을 결정하는 도로선분의 연장선의 교점이 p라고 하자. 이 때, 구역 r을 R로부터 제거(또한 구역 r을 정의한 도로선분을 H로부터 제거)하고 r'의 경계선을 수정한다. p를 결정한 r'의 기울어진 경계선이 아래(위)를 향하고 있었으면 수직으로 아래(위)로 향하는 선분 l_1과 이 선분이 r의 경계선과 만나면 아래(위)로 θ 만큼 기울어진 선분 l_2를 추가해서 r'의 경계를 수정한다. (그림 5.)

p가 r의 경계를 지나지 않으면 p는 무시한다.

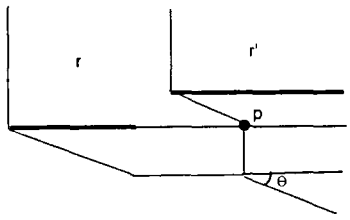

그림 5. p가 교차점

위에서 설명한 알고리즘의 시간 복잡도는 자료구조 E, R, H의 구현에 의해 결정된다. E는 우선순위 큐로 구현하면 E에서 이벤트 점들을 첨가하고 제거하는데 O(log n) 시간이 걸린다. R과 H는 탐색트리로 구현하는데, 특히 R은 SPM의 각 구역의 경계선 선분만을 저장한다. R과 H에서 원하는 구역과 도로선분을 찾는데 O(log n)이면 충분하다. 따라서 위 알고리즘의 시간복잡도는 O(nlog n)이다.

정리 1. n개의 평행한 도로선분을 포함하는 L^1 평면에서 SPM을 O(nlog n) 시간에 구할 수 있다.

참고문헌

[1] O. Aichholzer, F. Aurenhammer, and B. Palop. Quickest paths, straight skeletons, and the city voronoi diagram. In Proc. 18th Annu. ACM Sympo. Comput. Geom., pages 151-159, 2002.

[2] J. S. B. Mitchell. L_1 shortest paths among polygonal obstacles in the plane. Algorithmica, 8:55-88, 1992.

[3] J. S. B. Mitchell. Shortest paths among obstacles in the plane. Internat. J. Comput. Geom. Appl., 6(3):309-331, 1996.

[4] J. S. B. Mitchell and C. H. Papadimitriou. The weighted region problem: Finding shortest paths through a weighted planar subdivision. Journal of the ACM, 38(1):18-73, 1991.

[5] D. Z. Chen, K. Klenk, and H-Y. Tu. Shortest path queries among weighted obstacles in the rectilinear plane. SIAM J. Comput., 29:1223-1246, 2000.

[6] J. S. B. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, editors, Handbook of Computational Geometry, pages 633-701. Elsevier, 2000.