디루니 삼각분할의 병렬처리 알고리즘

위영철
아주대학교
ycwee@madang.ajou.ac.kr

황시영
현대정보기술
shwang@shinbiro.com

A Parallel Algorithm for Constructing the Delaunay Triangulation in the $L_\infty(L_1)$ Metric

Youngcheul Wee
Ajou University

Seeyoung Hwang
Hyundai Information Technology

요 약

본 논문은 영역별 근접 그래프 (geographic nearest neighbor graph)와 레인지 트리 (range tree)를 이용하여 평면 위의 n 개의 점에 대한 $L_\infty(L_1)$ 거리 (metric) 상의 디루니 삼각분할 (Delaunay triangulation)을 구축하는 방법을 소개한다. 이 방법은 $L_\infty(L_1)$ 거리 상에서 디루니 삼각분할에 있는 각 삼각형의 최소한 한 선분이 영역별 근접 그래프에 포함됨을 이용하여 레인지 트리 방법으로 디루니 삼각분할을 구축한다. 본 방법은 $O(n\log n)$의 순차계산 시간에 $L_\infty(L_1)$ 디루니 삼각분할을 구축하며, CREW-PRAM (Concurrent Read Exclusive Write Programmable Random Access Machine)에서 $O(n)$의 프로세서로 $O(\log n)$의 병렬처리 시간에 $L_\infty(L_1)$ 디루니 삼각분할을 구축한다. 또한, 이 방법은 직선간의 교차점 계산 대신 거리비교를 하기 때문에 수치오차가 적고 구현이 용이하다.

1. 서론

$L_\infty(L_1)$ 거리 (metric) 상의 보로노이 다이아그램 (Voronoi diagram: VD)은 로보틱스 (robotics), VLSI 설계, scheduling 문제 [5,6] 등 많은 분야에 활용되고 있다. 디루니 삼각분할 (Delaunay triangulation: DT)은 VD의 듀얼(dual) 그래프이므로 DT와 VD 구축은 같은 문제로 취급된다. 평면 위의 n 개의 점에 대한 DT를 $O(n\log n)$의 최적시간에 구축하는 순차적 알고리즘들은 [8] 알려져 있으나, 병렬처리 알고리즘들은 CREW-PRAM (Concurrent Read Exclusive Write Programmable Random Access Machine)에서 $O(n)$ 프로세서로 $O(\log^2 n)$ 시간 [1,2] 또는 $O(n^3)$ 프로세서로 $O(\log n)$ 시간이 소요된다 [7]. 본 논문에서는 영역별 근접 그래프 (geographic nearest neighbor graph: GNNG)에 레인지 트리 (range tree) 방법을 도입하여 $L_\infty(L_1)$ 거리 상에서 평면 위의 n 개의 점에 대한 DT를 구축하는 방법을 소개한다. 이 방법은 $O(n\log n)$의 최적 순차계산 시간에 DT를 구축하며, CREW-PRAM 상에서 $O(n)$의 프로세서로 $O(\log n)$의 최적 병렬처리 시간에 DT를 구축한다.

평면 위의 n 개의 점 S에 대한 점 a의 한 사분면 근접 점은 점 a의 한 사분면에 있는 S의 점들 중에서 a에 가장 가까운 점이 된다. S에 대한 사분면 근접 그래프 $QNG(S)$는 S의 각 점에 대한 네 방향 사분면 근접 점들을 에지로 연결 한 그래프이다. L_∞ 거리의 단위거리형 (한 점에서 단위거리에 있는 점들의 집합)은 밑변이 x 축에 평행한 정사각형이 된다. L_1 거리의 단위거리형은 L_∞ 단위거리형을 45° 회전 한 모양이 된다. 앞으로 별도의 언급이 없으면 거리는 L_∞로 간주하며 논의한 결과는 L_1 거리 상에서도 바로 적용된다.

본 논문에서는, $DT(S)$에 있는 각 삼각형의 최소한 한 에지가 사분면 근접 그래프 $QNG(S)$

에 포함됨을 밝히고, $QNG(S)$의 각 에지에 대응되는 $DT(S)$의 각 삼각형을 레인지 트리 탐색으로 찾아내는 방법을 논의한다. 이 방법은 $DT(S)$의 각 삼각형을 $QNG(S)$의 각 에지로부터 개별적으로 구축하기 때문에 병렬처리가 용이하게 된다. 또한, 이 과정에서 직선간의 교차점 계산 대신 거리비교를 하기 때문에 수치오차가 적고 안정적이다.

2. QNG와 DT의 기하학적 특성

세 점 a, b, c가 변에 위치하는 정사각형을 $\square(a, b, c)$로 나타내기로 하자. $\square(a, b, c)$가 S의 점을 내부에 포함하지 않으면 비어있다고 하고 $\square_E(a, b, c, S)$로 나타내기로 한다. 임의의 세 점 $a, b, c \in S$에 대하여 한 삼각형 (a, b, c)가 $DT(S)$에 포함되기 위한 필요충분 조건은 $\square_E(a, b, c, S)$가 존재하는 것이다 [4]. 여기서, 두 점 이상이 정사각형의 한 변에 있으면 $\square_E(a, b, c, S)$가 유일하지 않을 수 있으며, $DT(S)$도 유일하지 않을 수 있다. 따라서, 다른 논문에서와 같이 논의를 간결하게 하기 위하여 두 개 이상의 점이 정사각형의 한 변 있을 수 없는 것으로 가정한다. 본 방법을 일부 변형하면 이 조건을 만족하지 않는 경우도 처리된다.

정리 1: $DT(S)$에 속한 한 삼각형 (a, b, c)의 두 점 a, b가 $\square_E(a, b, c, S)$의 서로 인접한 변에 놓여있으면 $(a, b), (b, a)$ 중의 적어도 하나는 $QNG(S)$에 속하게 된다.

증명: 생략. ■

세 점이 한 정사각형의 각각 다른 변에 놓여 있으면 적어도 한 쌍의 두 점은 정사각형의 인접한 변에 위치하게 된다. 따라서, $DT(S)$의 각 삼각형 (a, b, c)의 적어도 한 에지가 빈 정사각형 $\square_E(a, b, c, S)$에서 인접하게 되며, 정리 1에 따라 $QNG(S)$에 속하게된다. 그러므로, $QNG(S)$의 각 에지 (a, b)에 대응되는 $\square(a, b, _, S)$를 찾으면 바로 $DT(S)$를 구축하게 된다. 임의의 두 점 $a, b \in S$에 대하여, 두 점 a, b가 인접하여 위치한 $\square_E(a, b, _, S)$를 찾는 방법은 다음과 같다.

두 점 a, b가 $\square(a, b, _)$의 인접한 두 변에 있으면 $\square(a, b, _)$의 한 꼭지점 r은 a, b를 통과하는 수직선과 수평선의 교점 (a_x, b_y) 또는 수평선과 수직선의 교점 (b_x, a_y)이 된다. 두 점 a, b에 의하여 결정되는 $\square(a, b, _)$의 꼭지점을 a, b의 수직교점이라 하자.

정리 2: 두 점 $a, b \in S$의 수직교점 r의 사분면 $\llcorner r, a, b$에 대한 근접 점이 $c \in S$ 일 때

$$d(r, c) \geq \max(d(r, a), d(r, b)) \Leftrightarrow (a, b, c) \in DT(S).$$

증명: 생략. ■

정리 2에 따라, 수직교점 r의 사분면 $\llcorner r, a, b$에 대한 근접 점을 알면 바로 $(a, b, c) \in DT(S)$를 판별 할 수 있다.

3. 알고리즘 및 분석

2절의 논의 결과에 따라, $DT(S)$는 구축하는 알고리즘은 다음과 같이 구성된다. $\llcorner r, a, b$

Delaunay(S)

1) 사분면 그래프 $QNG(S)$를 구축한다.

2) $DT = \{\ \}$

$QNG(S)$에 속한 각 에지 (a, b)의 두 수직 교차점 r에 대하여

- 사분면 $\llcorner r, a, b$의 근접 점 c가 있고 $d(r, c) \geq \max(d(r, a), d(r, b))$ 이면
  $DT = DT + (a, b, c)$
- 사분면 $\llcorner r, a, b$의 근접 점이 없으면
  $DT = DT + (a, b)$

위의 알고리즘 Delaunay에서 1)의 사분면 근접 그래프 구축은 2)의 사분면 근접 점 질의에 필요한 레인지 트리를 구축함과 동시에 구축된다. 또한, 한 사분면에 대한 근접 점 처리 방법은 좌표축의 회전에 의하여 다른 사분면 근접 점에도 적용된다. 다음은 남서 사분면 근접 점에 대한 레인지 트리를 구축하는 방법과 근접 점 질의를 처리하는 방법을 살펴본다. L_∞ 거리의 단위거리형은 정사각형이므로 한 사분면 영역 내의 단위거

리형의 경계 면은 하나의 수직선과 하나의 수평선으로 구성된다. 영역내의 단위거리형의 경계 면이 하나의 직선이 되면 일반적인 레인지 트리 방법을 근접 점 질의에 적용 할 수 있다. 남서 사분면 근접 점은 남서서 팔분면 근접 점 또는 남남서 팔분면 근접 점이 되므로 두 개의 팔분면 근접 점에 대한 레인지 트리로 남서 사분면 근접 점 질의를 처리 할 수 있다. 레인지 트리는 $O(n)$ 프로세서로 $O(\log n)$ 시간에 구축 할 수 있으며 이 트리 상에서 각 질의는 $O(1)$ 프로세서로 $O(\log n)$ 시간에 처리 할 수 있다 [9,10]. 단계 1)의 사분면 그래프 $QNG(S)$는 S에 있는 n개의 각 점에 대하여 사분면 근접 점 질의를 함으로써 구축 될 수 있으며 $O(n)$ 프로세서로 $O(\log n)$ 시간에 처리된다. $QNG(S)$의 각 에지 당 2 개의 수직교차점이 있으며 $QNG(S)$의 에지 수는 $O(n)$ 이기 때문에 단계 2)에서 전체 근접 점 질의는 $O(n)$ 프로세서로 $O(\log n)$ 시간에 처리된다. 따라서 위의 알고리즘은 $O(n)$ 프로세서로 $O(\log n)$ 시간에 처리된다.

4. 결론

본 논문에서는 영역별 근접 그래프와 레인지 트리를 이용하여 평면 위의 n 개의 점에 대한 DT (따라서, VD)를 $O(n\log n)$의 순차계산 시간과 CREW-PRAM 상에서 $O(n)$의 프로세서로 $O(\log n)$의 병렬처리 시간에 처리하는 방법을 고안하였다. 디루니 삼각분할은 최소 $\Omega(n\log n)$의 계산시간을 요구하기 때문에 이 방법의 처리 시간은 최적이 된다. 또한, 이 방법은 직선간의 교차점 계산 대신 거리비교를 하기 때문에 수치 오차가 적고 안정적이며 구현이 용이하여 실용적이다.

본 논문과 관련된 미해결 문제들을 살펴보면, L_2 거리 상의 최적 병렬처리 디루니 삼각분할 방법은 아직도 중요한 미해결 문제로 남아 있으며, L_2 거리 상의 영역별 근접 그래프 문제는 병렬처리 뿐만 아니라 순차계산도 최적시간 알고리즘이 알려져 있지 않다 [2,3].

5. 참고문헌

[1] A. Aggarwal, B. Chazelle, L. Guibas, C. Ó'Dúnlaing and C. Yap, Parallel Comutational Geometry, *Algorithmica*, 293-327, 1988.

[2] R. Cole, M.T. Goodrich, Optimal Parallel Algorithms for Polygon and Point-Set Problems, *Proc. of the Fourth Ann. Symp. on Computational Geometry*, 201-210, 1988.

[3] L.J. Guibas and J. Stolfi, On Computing all North-east Nearest Neighbors in the L_1 Metric, *Info. Process. Lett.* **17**, 219-223, 1983.

[4] D.T. Lee, Two-Dimensional Voronoi diagrams in L_P-Metric, *J.ACM* **27**(4), 604-618, 1980.

[5] D.T. Lee and C.K. Wong, Voronoi Diagrams in $L_1(L_\infty)$ Metrics with 2-Dimensional Storage Applications, *SIAM J. Comput.* 9(1), 200-211, 1980.

[6] E. Papadopoulou and D.T. Lee, Critical Area Computation via Voronoi Diagrams, *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 18(4), 463-474, 1999.

[7] W. Preilowski and W. Mumbeck, A Time-Optimal parallel Algorithm for the Computing of Voronoi diagrams, Lecture Notes *in Computer Science, Springer Verlag*, 424-433, 1988.

[8] P. Su, R.L. Drysdale, A Comparison of Sequential Delaunay Triangulation Algorithms, *Computational Geometry* 7, 361-385, 1997.

[9] Y.C. Wee and S. Chaiken and S. S. Ravi, Rectilinear Steiner Tree Heuristics and Minimum Spanning Tree Algorithms Using Geographic Nearest Neighbors, *Algorithmica* 12, 421-435, 1994.

[10] D.E. Willard and Y. C. Wee, Quasi-valid Range Querying and its Implications for Nearest Neighbor Problems, *Proceedings of the Fourth Annual Symp. on Computational Geometry*, 34-43, 1988.