

Algorithmics

Sebastian Iwanowski
FH Wedel

6. Fundamentals of Algorithmic Geometry

6.1 Basic Problems and the Use of Voronoi Diagrams for Solving them

Algorithmics 6

6.1 Basic Problems

Given n reference points in \mathbb{R}^k

1) Detection of the closest reference point for a new point $x \in \mathbb{R}^k$

Brute force algorithm: $O(n)$

- i) Compute the distance between x and each reference point.
- ii) Successively update the reference point with minimum distance to x .

2) Detection of the closest pair among the reference points

Brute force algorithm: $O(n^2)$

- i) Compute the distance between each pair of reference points.
- ii) Successively update the closest pair.

References:

deBerg et al., ch. 1, Levitin, ch. 3.3

Algorithmics 6

6.1 Basic Problems

Given n reference points in \mathbb{R}^k

3) Minimum spanning tree between all reference points

Brute force algorithm: $O(n^2 \log n)$

- i) Compute the distance for each pair of reference points and create an edge weighted with that distance.
- ii) In the resulting graph, compute the minimum spanning tree with the algorithm of Kruskal.

4) Detection of a set of reference points comprising the convex hull

Brute force algorithm: $O(n^{k+1})$

- i) For each subset of k points, compute the supporting hyperplane determined by them.
- ii) For each hyperplane, check if the other points are located on the same side:
The supporting hyperplane is part of the convex hull
 \Leftrightarrow All points are located on the same side.
- iii) Collect the points belonging to the supporting lines of the convex hull.

References:

deBerg et al., ch. 1, Levitin, ch. 3.3

Algorithmics 6

6.1 Voronoi Diagrams in the plane \mathbb{R}^2

Def.: Given a set S of n reference points in the plane.
The Voronoi diagram is a data structure belonging to the equivalence classes of the closest reference points *if that point is unique*:
The diagram partitions the plane into disjoint regions, each having exactly one reference point as the closest. The boundaries of these regions consist of the points whose closest reference point is not unique.

Structure: The Voronoi diagram $V(S)$ consists of the following type of objects:

- nodes: points having at least three reference points as the closest
- edges: points having exactly two reference points as the closest
- regions: points having exactly one reference point as the closest

One region is uniquely associated with one reference point.

The incident edges and nodes are attributes of a region.

The attributes of an edge are the incident regions and nodes.

The attributes of a node are the incident regions and edges.

Theorem: There are only $O(n)$ Voronoi objects.

References:

Klein, Kap. 5.1, 5.2 (in German), de Berg et al., ch. 7.1

Algorithmics 6

6.1 Voronoi Diagrams in the plane \mathbb{R}^2

Applications

1) Detection of the closest reference point for a new point $x \in \mathbb{R}^2$

Algorithm: $O(n^2 \log n)$ preprocessing + $O(\log n)$ runtime

$O(n)$ preprocessing time
is achievable by a more
sophisticated method

Preprocessing: $O(n^2 \log n)$

- i) Sort the nodes of $V(S)$ by their y-coordinate and divide the plane into according horizontal strips.
- ii) Intersect all edges of $V(S)$ with the horizontal strips and sort them by x-coordinate within a particular strip. **Note:** This may yield $O(n^2)$ segments

Runtime: $O(\log n)$

- i) Determine the correct horizontal strip for x by binary search towards the y-coordinate.
- ii) Determine the two edges closest to x by binary search towards the x-coordinate.
- iii) The region belonging to both of the edges is the correct one.

References:

Klein, Kap. 5.3.1 (in German)

Algorithmics 6

6.1 Voronoi Diagrams in the plane \mathbb{R}^2

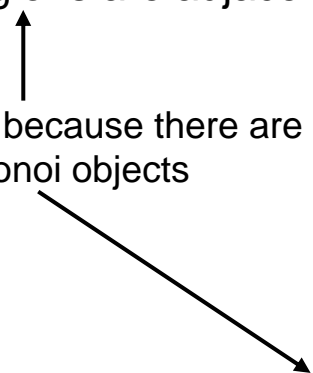
Applications

2) Detection of the closest pair among the reference points

Algorithm: $O(n)$

- i) Compute the distance only between points whose regions are adjacent.
- ii) Successively update the closest pair.

of order $O(n)$, because there are only $O(n)$ Voronoi objects



3) Minimum spanning tree in the plane

Algorithm: $O(n \log n)$

- i) Compute the distance from each reference point only to those reference points belonging to adjacent regions and create an edge weighted with that distance.
- ii) In the resulting graph, compute the minimum spanning tree with the algorithm of Kruskal.

References:

Klein, Kap. 5.3.2, 5.3.3 (in German)