CS 6050/7050 Computational Geometry

Spring Semester, 2018

Assignment 3: Divide-and-Conquer, the Closest-Pair Problem, DCEL, and Point Set Triangulation

Due Day: 8:00 p.m. Wednesday, March 14, 2017 (at the beginning of the class)

1. Let P be a set of n points in the plane. We say that point (x, y) *dominates* point (x', y') if $x \geq x'$ and $y \geq y'$. A point in P that is dominated by no other points of P is said to be *maximal*. Note that P may contain multiple maximal points. See Fig. 1 for an example. The problem is to compute all all maximal points of P. For simplicity of discussion, we make a general position assumption that no two points of P have the same x-coordinate or y-coordinate.

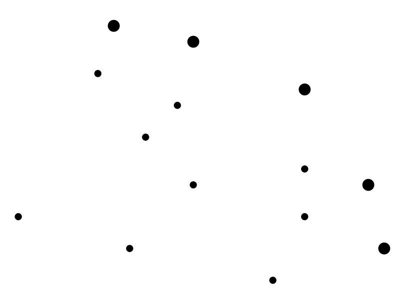

Figure 1: The five larger points are maximal points.

   (a) Design an $O(n \log n)$ time **divide-and-conquer** algorithm for the problem. **(15 points)**

   (b) Suppose all points of P have already been sorted from left to right by their x-coordinates. Design an $O(n)$ time algorithm for the problem. **(15 points)**

2. **(20 points)** We have studied the *Euclidean closest pair* problem in class, where the distance of two points in the plane are measured by their Euclidean distance. In this exercise, we consider the *rectilinear closest pair* problem. Let P be a set of n points in the plane. For any two points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$, their *rectilinear distance* is defined to be $|x_1 - x_2| + |y_1 - y_2|$. The rectilinear closest pair problem is to find a closest pair of points of P with respect to the rectilinear distance.

   Modify the $O(n \log n)$-time algorithm for the Euclidean closest pair problem we covered in class to solve the rectilinear closest pair problem in $O(n \log n)$ time.

   Remark: Euclidean distance is also called L_2-*distance*, and rectilinear distance is also called L_1-*distance* or *Manhattan distance*.

3. **(20 points)** In this exercise, we consider the following problem we left in class.

Suppose we have a planar subdivision P represented by the doubly-connected edge list (DCEL). Let v be a vertex of P. Let m be the number of edges of P that are incident to v. By using the DCEL data structure, give an $O(m)$-time algorithm to report all edges of P that are incident to v.

4. **(20 points)** Let P be a set of n points in the plane. We make a general position assumption that no three points of P lie on the same line. Let T be a triangulation of P. Let n_t be the number of triangles of T. Let n_e be the number of edges of T. Let h be the number of edges of the convex hull of P. See Fig. 2 for an example.

   Prove that $n_e = 3n - h - 3$ and $n_t = 2n - h - 2$.

   Hint: If we consider T as a planar subdivision, then there are n vertices, n_e edges, and $n_t + 1$ faces (each triangle is a face and the outer space is also a face). According to Euler's formula, we have $n + (n_t + 1) = n_e + 2$.

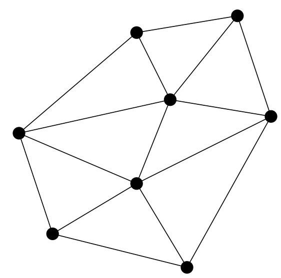

Figure 2: Illustrating a triangulation of $n = 8$ points. We have $h = 6$, $n_e = 15$, and $n_t = 8$.

Total Points: 90