COMP163 Homework Assignment 2: Due Tuesday, October 2, 2018

Reading: Please read Chapters 2-3 in the Text and also begin reading the Planar Point Location section of "Computational Geometry: A User's Guide." Also review the "Marriage Before Conquest" Convex Hull Algorithm in the lecture notes as well as the Convex Hull section of "Computational Geometry: A User's Guide." Begin reading Chapter 6 in the text on Point Location.

General Information: When describing an algorithm, do not forget to analyse its running time and explain why the algorithm is correct. Although you may discuss these problems in the preliminary stages with others, work submitted should be done individually. If you have any discussions with others (students, friends, TAs, faculty, ...) relative to a homework problem or if you gain information from a written source other than your own notes from lecture, you are expected to identify your collaborator/source.

Problems

1. Maxima Finding: A two-dimensional point $p = (p.x, p.y)$ is said to *dominate* another two-dimensional point $q = (q.x, q.y)$ if and only if $p.x \geq q.x$ AND $p.y \geq q.y$. A point $p \in S = \{p_1, ..., p_n\}$ is *maximal* in S if there is no point $q \in S$ such that q dominates p. The *maxima* of S form the set $M = \{p \in S : p$ is maximal in $S\}$.

   a. Draw a set S of 10 points in the first quadrant of the cartesian plane. For each of these points, draw the largest rectangle within the first quadrant that is dominated by that point. Outline the boundary of the union of these rectangles. Color in those points that represent the maxima of S.

   b. Describe and analyse an incremental algorithm for computing the maxima of a set S that contains n points. Assume that all points in S are distinct.

   c. Describe and analyse a divide-and-conquer algorithm for computing the maxima of a set S that contains n points. Assume that all points in S are distinct.

   d. Describe and analyse a dynamic algorithm for computing the maxima of a set S.

   e. Describe and analyse a marriage-before-conquest algorithm for computing the maxima of a set S.

2. Line Sweep and Augmented Data Structures

   Let S be a set of n disjoint triangles in the plane. We want to find a set of $n - 1$ segments with the following propoerties:

   - Each segment connects a vertex of one triangle to a vertex on the boundary of another triangle.
   - The interiors of the segments are pairwise disjoint and they are disjoint from the triangles.
   - Together they connect all triangles to each other, that is by walking along the segments and the triangle boundaries it must be possible to walk from any triangle to any other triangle.

   Develop a plane sweep algorithm for this problem that runs in $O(n \log n)$ time. State the events and the data structures that you use explicitly, and describe the cases that arise and the actions required for each of them. Also state the sweep invariant.

   [N.B. Consider adding two edges into the mix: $(x_{min} - 1, y_{max} + 1), (x_{max} + 1, y_{max} + 1)$ and $(x_{min} - 1, y_{min} - 1), (x_{max} + 1, y_{min} - 1)$.]

3. Convex Polygons and Monotone Polygons.

   Given a polygon P made of up vertices $v_i = (x_i, y_i)$ in counterclockwise order, we want to determine whether it is (a) convex or (b) monotone.

   a. Convex Polygon? For each edge of the polygon P extending from v_i to v_{i+1}, direct the edge from v_i to v_{i+1} and translate it to the origin. What needs to be true about these vectors emanating from the origin for P to be a convex polygon? How long does it take to check it?

   b. Monotone Polygon? Again, translate each directed edge of polygon P to the origin. What needs to be true about these vectors for P to be monotone in *some* direction? Describe the most efficient algorithm that you can to check whether P is monotone.

4. DCEL data structure. The *pockets* of a simple polygon are the areas outside the polygon but inside its convex hull. Let P_1 be a simple polygon with m vertices and assume that a triangulation of P_1 as well as its pockets is given. Let P_2 be a convex polygon with n vertices. Show that the intersection of P_1 with P_2 can be computed in $O(m + n)$ time.