Homework Exam 1 2018

Deadline: 21 December 2018, 11:00

This homework exam has 6 questions for a total of 100 points. You can earn an additional 10 points by a careful preparation of your hand-in: using a good layout, good spelling, good figures, no sloppy notation, no statements like "The algorithm runs in $n \log n$." (forgetting the $O(..)$ and forgetting to say that it concerns time), etc. Use lemmas, theorems, and figures where appropriate. Your final grade will be the number of points divided by 10 (with a maximum of a 10). Note that solving only a subset of the problems is sufficient to get a 10.

Question 1

Let $\mathcal{S}$ be a planar subdivision with n vertices, and let e be a half-edge of $\mathcal{S}$ that is incident to the outer face.

(a) *(10 points)*
Give pseudo-code for an algorithm that, given a pointer to e, computes all vertices of $\mathcal{S}$ that have "hop-distance" at most one to the outer face. That is, vertices that incident to the outer face, or are adjacent to a vertex on the outer face. Your algorithm should use the 'Twin', 'NextEdge', 'PrevEdge', etc. fields to navigate (i.e. you cannot assume you can directly access a list of vertices). Describe in a few sentences the main idea of your algorithm and give its running time.

Hint: You can use the data fields of vertices, half-edges, and faces to store marks/flags.

(b) *(10 points)*
Suppose that you do not get a pointer to e, but instead get a pointer to some arbitrary half-edge f of $\mathcal{S}$. Describe an approach to find a half-edge incident to the outer face, and give its running time. (You are not required to give the pseudo-code for this approach.)

Question 2 *(10 points)*

Let P be a polygon with n vertices and h holes. Prove that any triangulation of P consists $n + 2h - 2$ triangles.

Question 3

Let S be a set of n disjoint line segments in the plane, and let $p \in \mathbb{R}^2$ be a point that does not lie on any of the line segments. You may assume that the segments in S are open, and that the set that contains p and all endpoints from S contains exactly $2n - 1$ points and that no three such points are colinear.

(a) *(10 points)*
Develop an $O(n \log n)$ time algorithm to compute the length of a longest (open) segment s that contains p but does not intersect any segment in S. If segment s does not exist your algorithm should return ∞. Prove that your algorithm is correct and achieves the stated running time.

(b) *(5 points)*
Is your algorithm still correct if the segments in S may intersect? If so, argue why, if not, give an example why not, and describe how to fix it. You do *not* have to argue about the running time of your algorithm in this scenario.

Question 4 *(20 points)*

Given a set R of n "red" points and a set B of n "blue" points in $\mathbb{R}^2$. Develop an algorithm that can test if there exists a line ℓ that separates R from B, that is, such that all points in R lie right of ℓ and all points in B lie left of ℓ. Prove that your algorithm is correct and analyze its running time. You may assume that any line contains at most two points of $R \cup B$ (i.e. there are no three colinear points).

Note: the number of points rewarded for this question will depend on the running time of your algorithm.

Question 5 *(10 points)*

Let P be a set of n points in $\mathbb{R}^2$, and let R be the shortest (in terms of Euclidean length) closed curve such that all points of P lie inside (or on the boundary of) the area enclosed by R. Prove that R is the convex hull $CH(P)$ of P.

Question 6

Let P be a set of n points in $\mathbb{R}^2$, let $D(c)$ be a unit disk, that is, a disk of radius one and center c, and let $P_c = P \cap D(c)$ be the subset of P that lies in a unit disk centered at c.

(a) *(10 points)*
Prove that there are at most $O(n^2)$ different sets P_c over all points $c \in \mathbb{R}^2$.

(b) *(10 points)*
Give a construction that shows that the above bound is tight in the worst case. In other words, show that there can sometimes be $\Omega(n^2)$ different sets P_c.

(c) *(5 points)*
Let k be a natural number. Sketch an $O(n^2 \log n)$ time algorithm that can compute the number of subsets of P of size k that can be covered exactly (i.e. the disk contains no additional points) by a unit disk. One or two paragraphs of description is sufficient; you do not have to prove correctness or give the full analysis.