Final Exam 2019-2020

29 January 2019, 17:00-20:00

This exam has 7 questions for a total of 90 points. You can earn an additional 10 points if you write readable, unambiguous, and technically correct. No statements like "The algorithm runs in $n \log n$." (forgetting the $O(..)$ and forgetting to say that it concerns time), etc. Your final grade will be the number of points divided by 10.

Read every question carefully (!), make sure you understand it, and be sure to answer the question. Answer questions in sufficient but not too much detail. You may *not* use the textbook, or any other notes during the exam. Be sure to put your name on every piece of paper you hand in. Good Luck!

Question 1 *(10 points)*

For each of the following tasks, state the running time for the best possible algorithm to perform the task. If the algorithm is deterministic, give the worst case running time. If the algorithm is randomized, indicate this and give the expected running time. Use k to denote the output size if applicable.

Stating only the running time is sufficient, no need to explain your answers in detail.

(a) Given a set of n line segments in the plane, testing if they are all pairwise disjoint.
(b) Let S be a set of n horizontal line segments in $\mathbb{R}^2$, and let $\mathcal{T}$ be an interval tree on S in which at every node stores its segments in a range tree with fractional cascading. Querying $\mathcal{T}$ to report all segments in S intersected by a vertical (query) line segment.
(c) Triangulating an x-monotone polygon with n vertices.
(d) Computing the Delaunay triangulation of n points in $\mathbb{R}^2$
(e) Given the Voronoi diagram of a set P of n points in $\mathbb{R}^2$, computing the convex hull of P.

Question 2

Let $\mathcal{A}$ be the arrangement of a set $\mathcal{L}$ of n lines in $\mathbb{R}^2$.

(a) *(5 points)*
Using O-notation, give a tight worst case upper bound on the total number of vertices of all unbounded faces of $\mathcal{A}$. Briefly explain your answer.

(b) *(5 points)*
Show that the bound you have given in the previous question is indeed tight.

Question 3 *(15 points)*

Let $\overline{qr}$ be a line segment with endpoint q left of endpoint r. Let ℓ be a line through q and r with strictly positive slope. The x-coordinate of p lies in between those of q and r, and above ℓ. Point p is separated from r by a horizontal slab S.

Formulate the above paragraph in its dual form using the usual point-line duality. It does not have to be a literal translation, but it should capture all geometric information from the above paragraph. Draw the construction in the dual plane, and clearly label the following objects in your drawing: $\overline{qr}^*, p^*, q^*, r^*, \ell^*, S^*$.

Question 4 *(15 points)*

Let P be a set of n points in $\mathbb{R}^2$ out of which k points appear on the convex hull of P. Prove that the Delaunay triangulation of P has exactly $2n - 2 - k$ triangles.

Question 5

To solve the "casting a 3D-polyhedron" problem we can either (i) explicitly construct the intersection of the feasible halfplanes, or (ii) use 2D-linear programming to find a point in the feasible region.

(a) *(4 points)*
Briefly describe/explain the algorithm of method (ii).

(b) *(3 points)*
Give one reason why you may want to prefer to use method (ii) over method (i) to solve the casting problem.

(c) *(3 points)*
Give a *different* reason (i.e. a reason involving a different aspect of the algorithm) why you may want to use method (i) rather than method (ii) to solve the casting problem (or maybe a variant of the casting problem).

Question 6 *(15 points)*

Let $\mathcal{S}$ be a planar subdivision, and let $x_1, .., x_n$ be the x-coordinates of the n vertices of $\mathcal{S}$, in increasing order (you can assume all x-coordinates are unique).

Recall the following naive data structure $\mathcal{D}$ to answer point location queries on $\mathcal{S}$:

Data structure $\mathcal{D}$ stores the sorted x-coordinates $x_1, , .x_n$, and an array of binary search trees. In particular, entry $\mathcal{D}[i]$, for $i \in [1, n)$, of the array stores a binary search tree of all edges of $\mathcal{S}$ intersected by a vertical line ℓ at x-coordinate $(x_i + x_{i+1})/2$, ordered from bottom to top.

This data structure $\mathcal{D}$ uses $O(n^2)$ space, and can answer point location queries in $O(\log n)$ time.

Briefly (at most a few paragraphs) describe an algorithm to construct $\mathcal{D}$ in $O(n^2)$ time. Argue that your algorithm achieves the desired running time.

Question 7

Let S be a set of n line segments in $\mathbb{R}^2$, and let R be a (query) rectangle. A segment $s \in S$, is a *valid* segment (with respect to R) if it has an endpoint inside R.

Hint: Read the entire question first!

(a) *(5 points)*
Briefly argue that using a range tree (without fractional cascading) we can *report* all k valid segments in $O(\log^2 n + k)$ time.

(b) *(5 points)*
Consider augmenting the range tree so that for every node v (in the primary tree and the secondary trees) we store the number of points in the associated set P_v of node v.

Argue why this is *not* sufficient to *count* the number of valid segments of some query rectangle R in $O(\log^2 n)$ time.

(c) *(5 points)*
Sketch how we can adapt the above approach to support counting the number of valid segments in $O(\log^c n)$ time, for some constant c.