Final Exam 2020-2021

3 February 2020, 15:15-18:15

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 P of n points in $\mathbb{R}^2$, for each point $p \in P$ computing the point $NN(p)$ in P closest to p.
(b) Given two simple polygons with n and m vertices, respectively, computing their intersection.
(c) Given the Voronoi diagram of a set P of n points in $\mathbb{R}^2$, computing the convex hull of P.
(d) Given a set of n halfplanes in $\mathbb{R}^2$, testing if there is a point q contained in all halfplanes.
(e) Given a planar subdivision $\mathcal{S}$ (as a DCEL) with n vertices, and a pointer to a face F in $\mathcal{S}$, reporting the faces in $\mathcal{S}$ adjacent to F.

Question 2 *(8 points)*

Give two reasons why it may be useful to prove that some given simple polygon is convex.

Question 3 *(15 points)*

Let $\overline{pq}$ be a line segment with positive slope, and endpoint p left of endpoint q. Let m be a line that intersects $\overline{pq}$ in point s, and such that p lies below m. The line ℓ be a line parallel to $\overline{pq}$ that intersects m in point r, with $r_y > q_y$.

(a)
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.

(b)
Draw the above construction in the dual plane, and clearly label the following objects in your drawing: $\overline{pq}^*, p^*, q^*, r^*, \ell^*, m^*$.

Question 4 *(7 points)*

Consider the Art Gallery Problem: given a simple polygon P with n vertices, compute a minimum size set of point guards G that together guard P (i.e. so that any point in P is seen from at least one point in G).

We saw an algorithm to compute a set of $\lfloor n/3 \rfloor$ vertices that together guard P. Is this a c-approximation, for some constant c, for the Art Gallery problem as stated above? Argue why or why not.

Question 5 *(15 points)*

Let P be a set of n points in $\mathbb{R}^2$, let p be a point in P, and let $q \in P$ be the nearest neighbor of p. Prove that $\overline{pq}$ is an edge in the Euclidean Minimum Spanning tree of P.

Question 6

Let $\mathcal{A}$ be an arrangement of n lines in $\mathbb{R}^2$.

(a) *(10 points)*
Briefly describe how we can use a trapezoidal decomposition to store $\mathcal{A}$ so that for **any** query point $q \in \mathbb{R}^2$ we can report all k edges of the face of $\mathcal{A}$ containing q in $O(\log n + k)$ expected time. State how much space your data structure uses, and briefly argue why it attains these space and query bounds.

There is no need to describe the trapezoidal decomposition or the query algorithm on a trapezoidal decomposition itself. Focus on what is needed in addition (in terms of what your data structure stores and how to answer queries).

(b) *(5 points)*
Briefly describe how we can store $\mathcal{A}$ to obtain $O(\log^2 n + k)$ query time in the worst case, using $O(n^2 \log n)$ space.

Question 7

Let P be a set of n points in $\mathbb{R}^2$, and let $\mathcal{T}(P)$ be a kd-tree storing P.

(a) *(5 points)*
Briefly describe how to use $\mathcal{T}(P)$ to report all k points in an axis aligned query rectangle R.

(b) *(5 points)*
Argue that reporting all k points in R takes $O(\sqrt{n} + k)$ time in the worst case.

(c) *(10 points)*
Show that $\mathcal{T}(P)$ **cannot** efficiently report all k points of P in a query parallelogram Q with two horizontal sides. That is, show that there exists a set of n points P and a query parallelogram Q (with two horizontal sides) such that $Q \cap P = \emptyset$, yet the query in $\mathcal{T}(P)$ takes $\Omega(n)$ time.