CMSC 754: Spring 2026 Dave Mount

Homework 5

Due by the start of class on Tuesday, Mar 31. (Submissions will be through Gradescope.) Late homeworks are not accepted (unless an extension has been prearranged) so please turn in whatever you have completed by the due date. Unless otherwise specified, you may assume that all inputs are given in *general position*.

Problem 1. You are given a collection of n non-intersecting circular disks in the plane, each of unit radius. Let $P = \{p_1, \ldots, p_n\}$ denote their center points. Preprocess these disks into a data structure to answer the following *escape queries*. Given a unit disk q (designated by its center point), is it possible for this disk to *escape* from the others, meaning that it is possible to move this disk arbitrarily far away from the others without intersecting any of the disks of P (see Fig. 1).

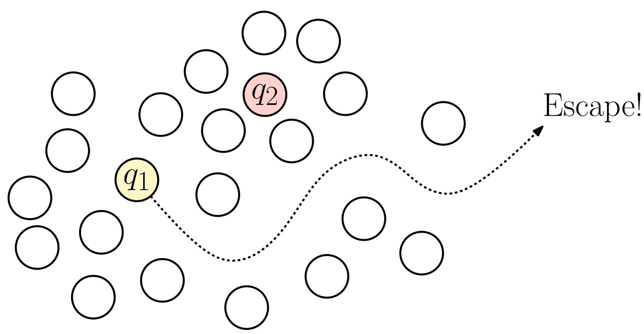

Figure 1: Determining whether a disk can escape. The disk q_1 can escape but q_2 cannot.

Present an efficient data structure for answering escape queries. Your data structure should use $O(n)$ space, answer queries (yes or no) in $O(\log n)$ time, and should be possible to build in $O(n \log n)$ given the set P. (**Hint:** Voronoi diagrams and point location. The space an query times are the more important element of the complexity bounds. If the construction takes longer than $O(n \log n)$, we will give partial credit.)

(a) Explain your data structure (how it is organized and what it stores), and how it is constructed. Derive its space and construction time complexities.

(b) Explain how queries are answered. (You may assume that the initial placement of the query disk does not overlap any of the disks of P.) Derive the query time complexity.

Problem 2. A robotic frog wants to cross a stream without getting wet. The stream is bounded by two horizontal lines $y = y^-$ and $y = y^+$. On the surface there are n circular lily pads whose centers are at the points $P = \{p_1, \ldots, p_n\}$. The frog crosses by moving across the circular lily pads. The lily pads grow at the same rate, so that at time $t \geq 0$ they all have radius t (see Fig. 2(a)).

Present an efficient algorithm that determines the earliest time $t^* > 0$ such that there exists a path from one side of the stream to the other by traveling along the lily pads (see Fig. 2(b)). (You do not need to determine the path, just the time of its existence.)

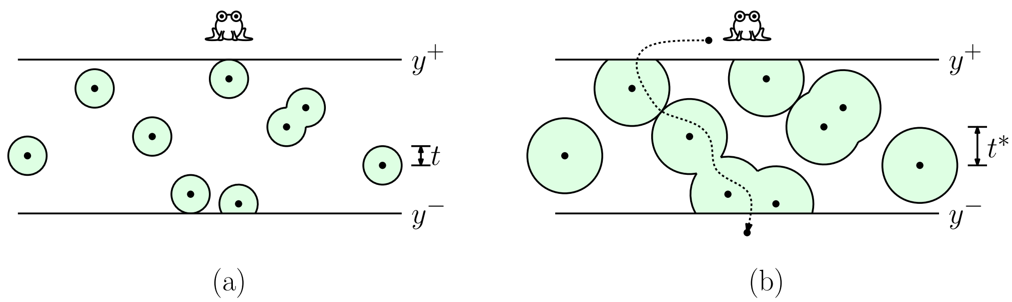

Figure 2: Stream crossing.

Explain your algorithm and derive its running time. (**Hint:** $O(n \log n)$ time is possible. This can be done by various approaches, including Voronoi diagrams, Delaunay triangulations, and Euclidean minimum spanning trees.)

Problem 3. In class we proved that the Delaunay triangulation of a set of sites in the plane maximizes the minimum angle. (It is the max-min angle triangulation.) Unfortunately, it is not the best triangulation for the following two criteria.

(a) Give an example of a set of point sites in the plane such that the Delaunay triangulation of this set *does not* minimize the sum of edge lengths, among all possible triangulations. In other words, the Delaunay triangulation is *not* the minimum-weight triangulation.

(b) Give an example of a set of point sites in the plane such that the Delaunay triangulation of this point set *does not* minimize the maximum angle, among all possible triangulations. In other words, the Delaunay triangulation is *not* the min-max angle triangulation.

In each case briefly explain your construction. Your example should be in general position (e.g., no four sites should be cocircular).

Hint: In both cases, it is possible build a counterexample consisting of just four points that are nearly co-circular. It suffices to present a single, specific example.

Problem 4. (Optional–Ungraded) We start with two (seemingly unrelated) definitions. A triangulation of a set of points in the plane is *acute* if the angles of all its triangles are strictly acute, that is, less than $\pi/2$. (See Fig. 3(a) for an example of a triangulation that is *not* acute.) The Voronoi diagram of a set of points in the plane is said to be *medial* if each edge of the diagram contains in its interior the midpoint of the two defining sites for the edge. (See the Fig. 3(b) for an example of a Voronoi diagram that is *not* medial.)

(a) Prove that any acute triangulation of a set of points in the plane is a Delaunay triangulation, that is, it satisfies the empty circumcircle property.

(b) Prove that if the Voronoi diagram of a set of points in the plane is medial, then the corresponding Delaunay triangulation is acute.

Hint: The following basic geometric fact may be useful. Let $\overline{pq}$ be a chord of a circle and let r and s be two points lying on the circle, one on either side of $\overline{pq}$. Then $\angle prq + \angle psq = \pi$. (See Fig. 3(c).)

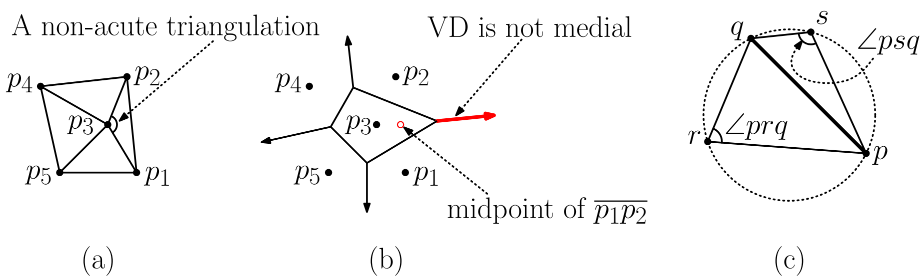

Figure 3: Acute triangulations and medial Voronoi diagrams.

Note: Challenge problems are not graded as part of the homework. The grades are recorded separately. After final grades have been computed, I may "bump-up" a grade that is slightly below a cutoff threshold based on these extra points. (But there is no formal rule for this.)

Challenge Problem: You are given a set of n points $P = \{p_1, \ldots, p_n\}$ all having strictly positive x- and y-coordinates. For each point, you draw two segments, one horizontally to the y-axis and the other vertically to the x-axis (see Fig. 4).

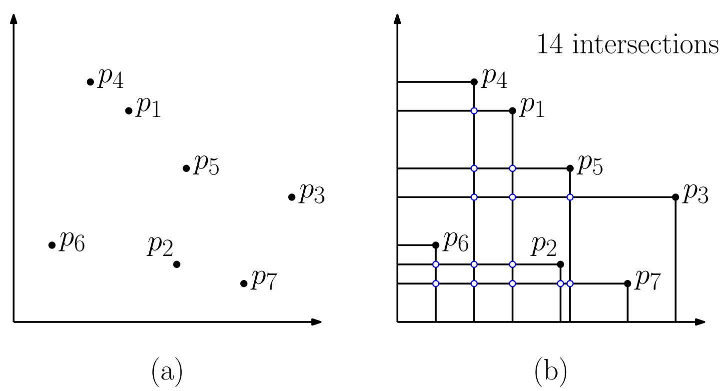

Figure 4: Counting intersections.

Present an efficient algorithm which, given P, returns a *count* of the number of intersection points among these segments (not counting the intersections occuring at each point of P). Your algorithm should running in strictly less than $O(n^2)$ time, and in fact $O(n \log n)$ time is achievable by a fairly simple algorithm. You may assume there are no duplicate x- or y-coordinates.