CMSC 754: Final Exam

In all problems, unless otherwise stated, you may assume that inputs are in *general position*. If you are asked for an $O(T(n))$ time algorithm, you may give a *randomized* algorithm with *expected* time $O(T(n))$. If you are asked to give an algorithm, also explain how it works and derive its running time. (Unless otherwise stated, formal proofs of correctness are not required.)

Problem 1. (25 points) Give a short answer (a few sentences at most) to each question. Except where requested, explanations are not required.

(a) Given a set of n points in the plane such that the convex hull contains h points, what is the running time of Chan's algorithm?

(b) Given n line segments in the plane with m intersecting pairs of segments, what are the time and space complexities of the plane-sweep algorithm for reporting all the intersections?

(c) Given a set of n lines in $\mathbb{R}^2$ in general position, as an exact function of n, what is the number of vertices in the associated line arrangement?

(d) Consider point set P of size n in $\mathbb{R}^d$ and a separation factor $s \geq 1$. What is the number of pairs in an s-WSPD of P, assuming the construction from class? Express your answer as an *asymptotic function* of n, s, and d, where d is a constant.

(e) You are given a collection of n axis-aligned squares in the plane. What is the maximum number of vertices on the boundary of the union of these squares? (An asymptotically correct answer is sufficient.)

Problem 2. (15 points) You are given a set $S = \{s_1, \ldots, s_n\}$ of n non-intersecting line segments in the plane, all lying above the x-axis, such that no line segment is horizontal or vertical (see Fig. 1(a)). The objective is to construct an efficient data structure for answering the following queries. Given any point $q \in \mathbb{R}^2$ that lies above the x-axis, suppose that a drop of water falls at q and drips from one segment to the next until reaching the x-axis. The answer to the query is the x-coordinate, $f(q)$, where the drop finally lands (see Fig. 1(b)). The data structure should have $O(n)$ space and answer queries in $O(\log n)$ time.

(a) (10 points) Explain how to construct the data structure from the set S of segments and justify its space and construction time. (You may use results proved in class lectures. The space and construction time should be $O(n)$ and $O(n \log n)$, respectively. The construction may be randomized. I would recommend providing a small figure illustrating your idea.)

(b) (5 points) Explain how queries are answered in $O(\log n)$ time.

Problem 3. (10 points) You are given three point sets A, B, and C in $\mathbb{R}^2$, which lie in a corridor bounded by two horizontal "walls" separated by a distance w (see Fig. 2(a)). You want to know whether there exists a starting point s lying to the left of the corridor and an angle θ such that resulting beam satisfies the following conditions (see Fig. 2(b)):

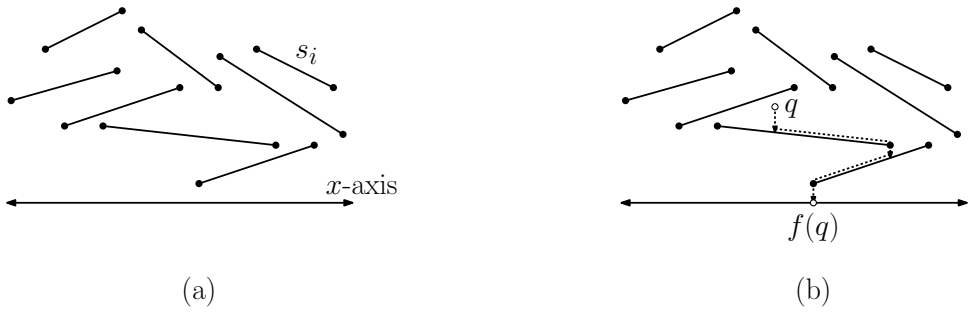

Figure 1: Water-drop queries.

- The beam is shot from s at angle θ, and it enters the corridor through the left side and first hits the lower wall.
- The beam reflects off the walls, with angle of incidence equal to angle of reflection
- We consider the first four segments of the reflecting beam. The points of A lie above the first two segments, the points of B lie below the second and third, and the points of C lie above the third and fourth.

If multiple valid solutions exist, report any one.

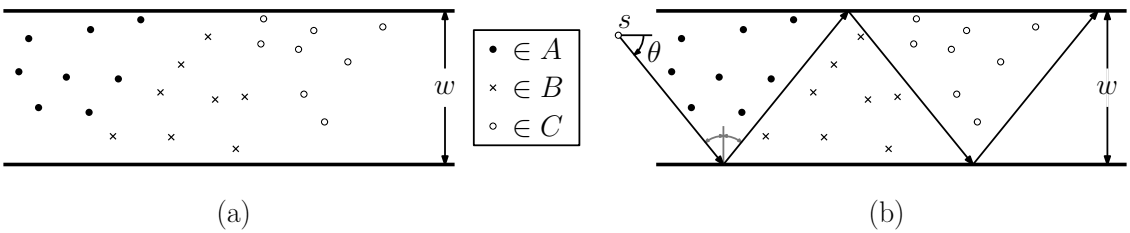

Figure 2: Aiming the laser beam.

Present an efficient algorithm for answering this problem, given A, B, C, and w. You may assume that the left end of the corridor is at $x = 0$ and the horizontal lines lie on $y = 0$ and $y = w$. Letting $n = |A| + |B| + |C|$, aim for a running time of $O(n)$. Justify your algorithm's correctness and derive its running time.

Problem 4. (15 points) You are given a set $P = \{p_1, \ldots, p_n\}$ of n points in $\mathbb{R}^2$ in general position. For $1 \leq i \leq n-1$, define the ith *critical radius*, denoted r_i, to be the smallest value such that, if we place a Euclidean ball of radius r_i centered at each point of P, the resulting collection of balls forms a set with $n - i$ connected components. Intuitively, the balls grow at an equal rate, and we want to find the events where two disconnected groups of balls bump into each other (see Fig. 3).

Present an efficient algorithm which, given P, computes the entire sequence $\langle r_1, \ldots, r_{n-1} \rangle$. Justify your algorithm's correctness and derive its running time. (**Hint:** Aim for a running time of $O(n \log n)$. Be sure that your justification explains clearly why the selected radius values are correct.)

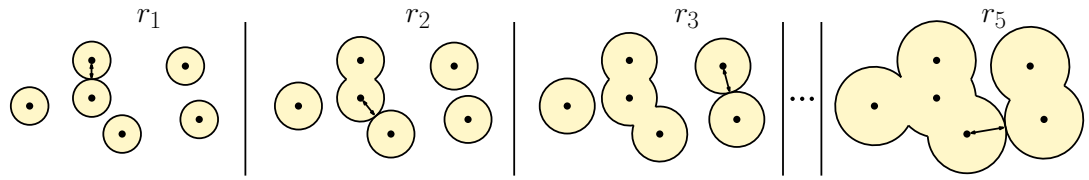

Figure 3: Critical radii.

Problem 5. (15 points) We are given two point sets A and B in the plane, which we assume are not separable from each other by any line. Let $n_A = |A|$, $n_B = |B|$, and $n = n_A + n_B$. Present an algorithm which, given A and B, computes two parallel lines ℓ^+ and ℓ^- such that all the points of A lie on or above ℓ^- and all the points of B lie on or below ℓ^+, and the vertical distance between these lines is minimum (see Fig. 4). Briefly explain your algorithm and derive its running time. (**Hint:** Aim for a running time of $O(n^2)$ and space of $O(n)$.)

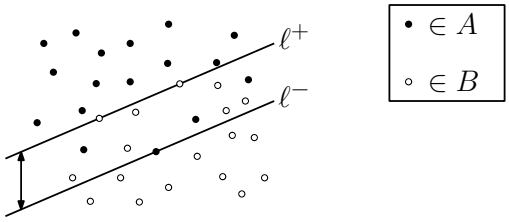

Figure 4: Partial linear separation.

Problem 6. (20 points) Given a point set $P = \{p_1, \ldots, p_n\}$ in $\mathbb{R}^d$, and given any query point $q \in \mathbb{R}^d$ (which need not be in P), its *farthest distance*, denoted $\text{far}_P(q)$ is the maximum distance from q to any point in P (see Fig. 5(a)). We will consider how to construct a coreset for this problem.

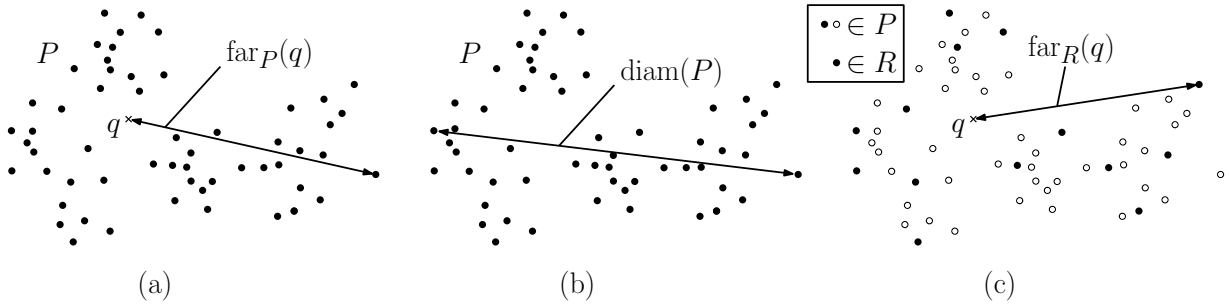

Figure 5: Coreset for farthest distance queries.

(a) (10 points) We first need a crude approximation of the farthest point distance. Prove that given any query point $q \in \mathbb{R}^d$, the distance between q and its farthest point in P is at least $\text{diam}(P)/2$. (Recall that the *diameter* of a point set, denoted $\text{diam}(P)$, is the maximum distance between any two points of the set (see Fig. 5(b)).) **Hint:** Use the triangle inequality.

(b) (10 points) Given $0 < \varepsilon < 1$, an ε-coreset for the farthest-point problem is a subset $R \subseteq P$ such that for any $q \in \mathbb{R}^d$,

$$(1 - \varepsilon) \cdot \text{far}_P(q) \;\leq\; \text{far}_R(q) \;\leq\; \text{far}_P(q).$$

(see Fig. 5(c))

Present an efficient algorithm that, given P and $0 < \varepsilon < 1$, constructs an ε-coreset for farthest-distance queries. Derive the size of your coreset and prove the correctness of your construction. For full credit, give a coreset of size $O\big(1/\varepsilon^{(d-1)/2}\big)$. For 3/4 credit, give a coreset of size $O\big(1/\varepsilon^{d}\big)$. (**Hint:** The construction is closely related to the coreset for directional-width queries. I think the analysis is much simpler for the coreset of size $O\big(1/\varepsilon^{d}\big)$.)