CMSC 754: Midterm Exam

This exam is closed-book and closed-notes. You may use one sheet of notes (front and back). Unless otherwise stated, you may assume that inputs are in *general position*. You may make use of any results presented in class and any well known facts from algorithms or data structures. 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. (20 points) Short-answer questions.

(a) (4 points) Given a planar point set with n points and h vertices on the convex hull. What must be true about the relationship between h and n for Jarvis's algorithm to run at least as fast as Graham's algorithm? (State your answer asymptotically.)

(b) (4 points) Given a simple polygon with $n \geq 3$ vertices, what is the maximum number of *reflex vertices* it might have. (Remember that a reflex vertex is one whose interior angle is greater than 180°.) For full credit, give an exact answer as a function of n.

(c) (4 points) Consider Fortune's algorithm to compute the Voronoi diagram of n sites in the plane. What is the maximum number of arcs that any one site can contribute to the beach line?

(d) (8 points) Which of the following assertions about the Delaunay triangulation (DT) of a set P of n points in the plane are true?

(1) The closest pair of points are connected by an edge in the DT.

(2) The farthest pair of points cannot be connected by an edge in the DT.

(3) Among all triangulations of P, the DT maximizes the minimum angle.

(4) Among all triangulations of P, the DT minimizes the maximum angle.

(5) Among all triangulations of P, the DT minimizes the total sum of edge lengths.

Problem 2. (15 points) Recall the plane-sweep algorithm for decomposing a simple polygon into monotonic pieces.

(a) (8 points) Give the values of helper(e_i), for $i = 1, \ldots, 4$ in Fig 1(a).

(b) (7 points) Show all the new diagonals that would be added as a result of the event at vertex v_i, for $i = 1, \ldots, 4$ in Fig 1(b).

Problem 3. (15 points) We are given a collection of n (non-vertical) line segments $S = \{s_1, \ldots, s_n\}$ in the plane *in no particular order*. Each segment is given by its endpoints $s_i = \overline{p_i q_i}$, where p_i has the smaller x-coordinate. We want to know whether altogether, these segments form the edges of a single simple polygon (see Fig. 2(a)). That is, we need to check the following things:

- No two segments intersect, except at their endpoints (see Fig. 2(b)).

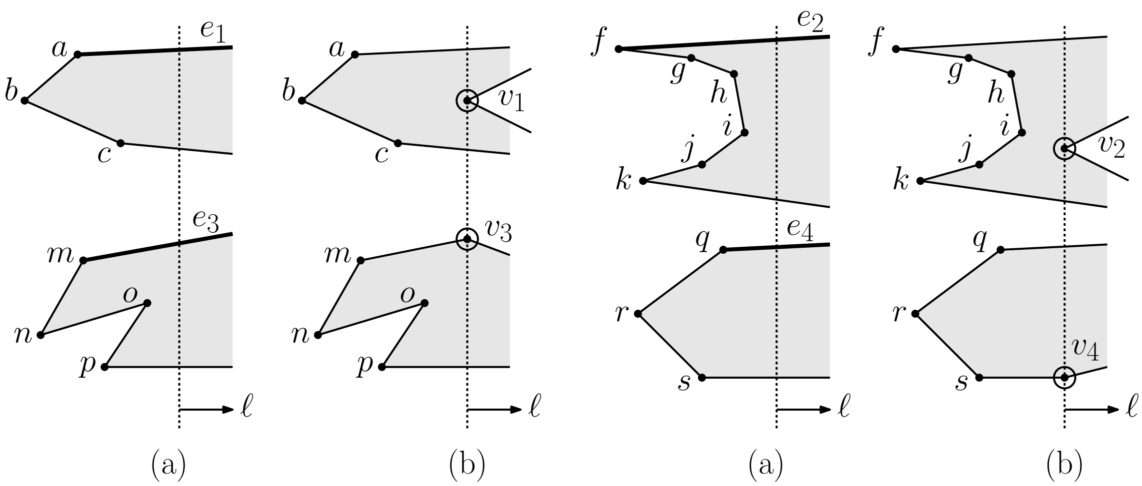

Figure 1: Plane-sweep monotone decomposition.

- Each vertex should have degree two, meaning that there are exactly two segments sharing each endpoint (see Fig. 2(b)).
- The segments form a single closed loop (see Fig. 2(c)).

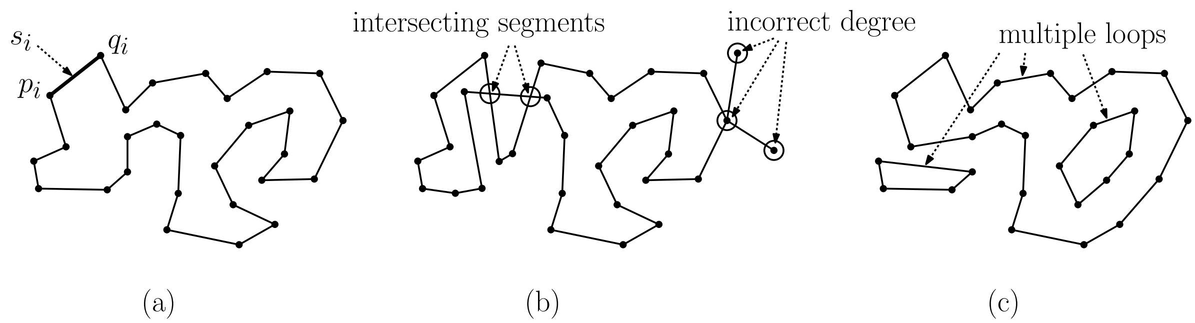

Figure 2: Does a set of segments define a simple polygon?

Present an efficient algorithm to check that the segments form a simple polygon. If the segments violate any of the above conditions, print the first such violation and terminate. Your algorithm should run in $O(n \log n)$ time. (Partial credit will be given if you correctly detect any of the three violations.)

Problem 4. (25 points) Explain how to solve each of the following problems in linear (expected) time. Each can be modeled by reduction to linear programming (LP), perhaps involving multiple instances along with some additional pre- and/or post-processing.

(a) (15 points) In your new career as an archer (shooting arrows), you want to shoot a single arrow through a series of targets. You may stand anywhere you want on the positive y-axis and may shoot in any direction you like. The targets are vertical line segments all in the positive x, y-plane (see Fig. 3(a)). There are n targets. The ith target is specified by giving its center point $c_i = (c_{i,x}, c_{i,y})$ and its height h_i.

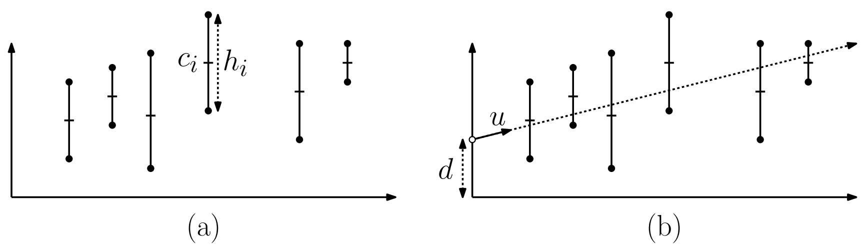

Figure 3: Hitting all targets with a single shot.

Present an algorithm that determines whether there exists a single (straight-line) shot that passes through all n targets. If *any* such a shot exists, output the height d where the shot originates on the y-axis and a directional vector $u = (u_x, u_y)$ that indicates the direction of the shot (see Fig. 3(b)). If there is no shot, indicate this. Your algorithm should run in $O(n)$ time.

(b) (10 points) Consider the same problem as (a) but with the following modification. As in (a), your shot must hit all the targets, but in addition it should come as close as possible (on average) to hitting the centers of the targets. More formally, given any shot, let p_i denote the point where the shot crosses the ith target (see Fig. 4). The objective is to minimize the *absolute value* of $\frac{1}{n}\sum_{i=1}^{n}(p_{i,y} - c_{i,y})$. Explain your algorithm and derive/explain its running time.

Hint: This is a bit tricky, and if you don't see the answer right away, you may want to come back to this later.

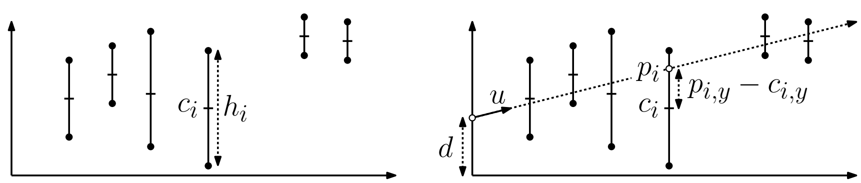

Figure 4: Hitting all targets close to the centers.

Problem 5. (25 points) Our objective is to develop a data structure to determine the first target (if any) that is missed by a shot in the archery problem (Problem 4). The data structure is constructed based on the target centers c_i and their heights h_i (see Fig. 5(a)). A query is given the y-coordinate d along the y-axis where the shot starts and a directional vector u (see Fig. 5(b)).

(a) (15 points) Present a data structure which, given d and $u = (u_x, u_y)$ determines the first target, if any, that is *missed* by the shot. You may assume that $d \geq 0$ and $u_x > 0$. Your data structure should use $O(n)$ space and answer queries in time $O(\log n)$.

(b) (5 points) Suppose that your shot hits all the targets. Assuming that you maintain the same directional vector for the shot, what is the minimum and maximum range of d values such that the shot hits all the targets? Explain how to modify your solution to (a) to answer this in the same space and query time bounds (see Fig. 6(b)).

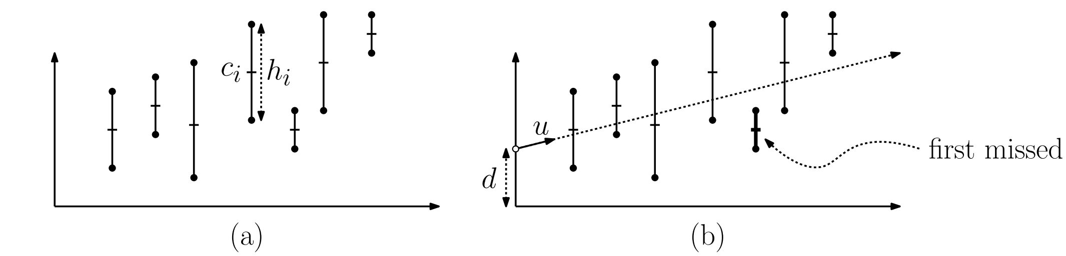

Figure 5: Arrow-shooting queries.

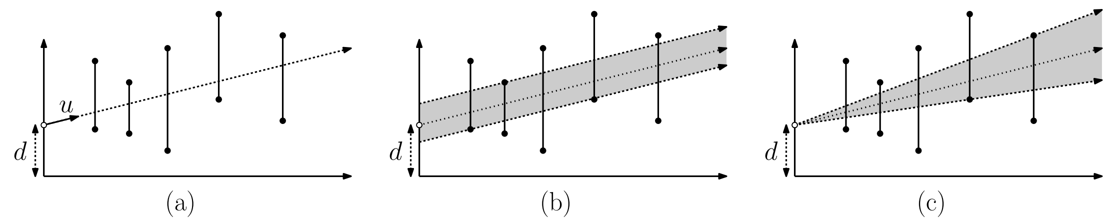

Figure 6: Arrow-shooting variants.

(c) (5 points) Suppose that your shot hits all the targets. Assuming that you maintain the same d value on the y-axis for the shot, what is the minimum and maximum range slopes such that the shot hits all the targets? Explain how to modify your solution to (a) to answer this in the same space and query time bounds (see Fig. 6(c)).

Hint: This is a bit tricky, and if you don't see the answer right away, you may want to come back to this later.