CMSC 754 Quiz 2

This quiz is closed-book and closed-notes, but you may use one sheet of notes, front and back. You may use any algorithms or results given in class. The total point value is 50 points. Good luck!

Problem 1. (5 points) Consider the two segments s_1 and s_2 shown in Fig. 1(a).

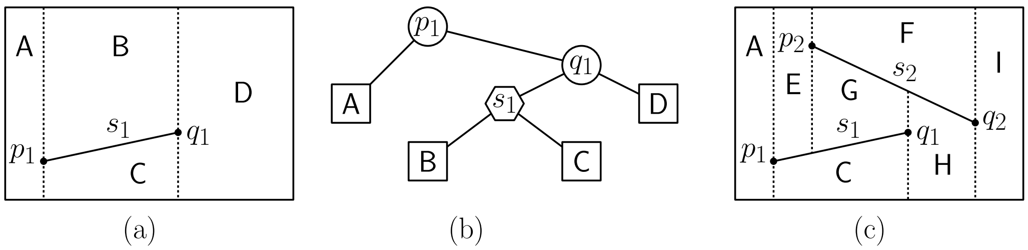

Figure 1: Trapezoidal map and point location.

We show the trapezoidal map and point-location data structure after adding segment s_1 (see Fig. 1(b) and (c)). Show the *final point-location data structure* that results after adding both segments. (Be sure to clearly distinguish the left child from the right child of each node.)

Problem 2: (10 points) Recall that Euler's formula states that in any cell complex, $v - e + f = 2$, where v, e, and f denote the number of vertices, edges, and faces, respectively. (Recall that f includes the unbounded face that extends to infinity.)

(a) (5 points) Given a triangulation with v vertices, where h of these lie on the convex hull, derive an (exact) formula for the number of triangles t in the triangulation as a function of v and h. (**Hint:** Use Euler's formula.)

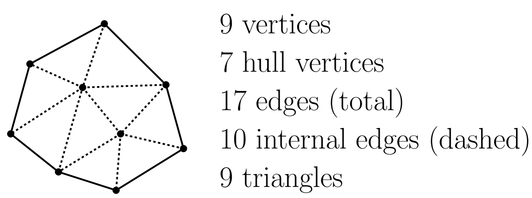

Figure 2: Applications of Euler's formula.

(b) (5 points) Consider the same set-up as (a). We say that an edge is *internal* if it is not on the boundary of the convex hull. Derive a formula for the number of *internal edges* in the triangulation as a function of v and h. (**Hint:** Use Euler's formula.)

Problem 3. (20 points) Explain how to solve each of the following problems in linear (expected) time by reducing it to a linear-programming (LP) problem. This may involve multiple instances and/or additional pre- or post-processing.

(a) (10 points) You are given a set of vertical line segments all in the positive quadrant of the x, y-plane, where the ith segment is specified by giving its center point $c_i = (c_{i,x}, c_{i,y})$ and its height h_i (see Fig. 3(a)).

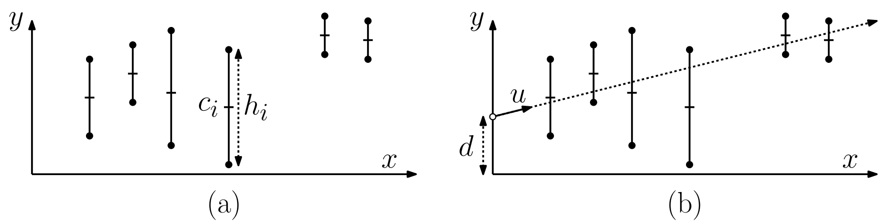

Figure 3: Hitting all segments with a single ray.

Present an algorithm that determines whether there exists a ray that originates on the y-axis and intersects all the segments. If such a ray exists, your algorithm should output the y-coordinate d where the ray originates and the ray's directional vector $u = (u_x, u_y)$ (see Fig. 3(b)). Explain briefly. (**Hint:** This is similar to a homework problem, but give your answer from scratch, that is, *do not* reference the homework solution.)

(b) (10 points) Consider the same problem as (a) but with the following modification. If there exists a shot that hits all the segments, output the shot that comes as close as possible (on average) to hitting the centers of the segments. More formally, given any shot, let p_i denote the point where the shot crosses the ith segment (see Fig. 4). The objective is to minimize the *absolute value* of $\frac{1}{n}\sum_{i=1}^{n}(p_{i,y} - c_{i,y})$. (Note that this is the absolute value of the average, not the average of the absolute values.) Your algorithm should run in $O(n)$ time. (**Hint:** You may simply explain how to modify your solution from (a).)

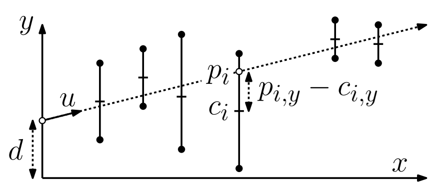

Figure 4: Hitting all segments close to the centers.

Problem 4. (15 points) This problem uses the same setup as the previous one. You are given the centers and heights of a set of vertical line segments. The objective is to develop a *data structure* to efficiently determine the leftmost segment (if any) that is *missed* by a given ray (see Fig. 5(a)).

(a) (10 points) The data structure is based on the centers c_i and heights h_i of the segments. A query consists of the y-coordinate d and direction vector $u = (u_x, u_y)$ of the ray. The query output is the index of the first (leftmost) segment that the ray *misses*. You may assume that $d \geq 0$ and $u_x > 0$. Your data structure should use $O(n)$ space and answer queries in time $O(\log n)$. Briefly explain. (**Hint:** This is similar to a homework problem, but give your answer from scratch, that is, *do not* reference the homework solution.)

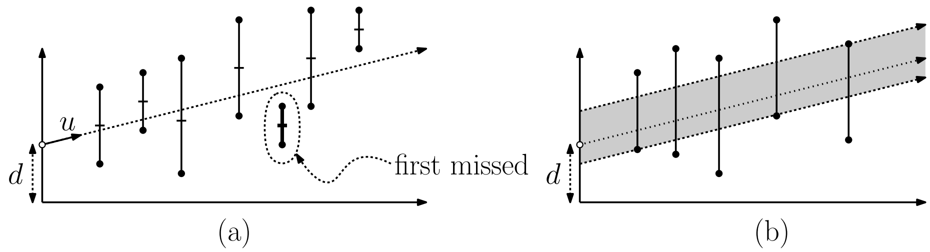

Figure 5: Ray-shooting queries.

(b) (5 points) Suppose that your shot hits all the segments. Assume that the ray intersects all the segments. Explain how to modify your solution to (a) in order to output the minimum ($d_{\min}$) and maximum ($d_{\max}$) y-coordinates of parallel rays that intersect *all* the segments (see Fig. 5(b)). Your data structure should use $O(n)$ space and answer queries in time $O(\log n)$. Briefly explain.