Homework 2: (Preliminary)

Handed out Tuesday, Oct 3. Due: 9:30am, Tuesday, Oct 17 (submission through Gradescope). No late homeworks will be accepted, 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. Also, when asked to give an algorithm with running time $O(f(n))$, it is allowed to give a randomized algorithm with expected running time $O(f(n))$.

Problem 1. 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. See the remarks at the end of the homework on how to present LP reductions.

Note that both problems involve obstacle avoidance. We allow the trajectory to pass through the boundary points of the obstacles. (This is necessary for LP, since it assumes that the constraints are closed halfspaces.)

(a) In your new career as a professional miniature golf player, you are working on a program to compute your best initial shot on a tricky hole. Let’s model this as a rectangle of height $w$ and length $\ell$ whose lower left coordinate is the origin (see Fig. 1(a)). There are a series of line segment obstacles that need to be avoided, each growing perpendicularly out from one of the rectangle’s sides. The input consists of points $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$, where $a_i = (a_{i,x}, a_{i,y})$, and $b_j = (b_{j,x}, b_{j,y})$. The points of $A$ are the endpoints of the segments ascending up from the rectangle’s bottom edge and the points of $B$ are the endpoints of the segment descending from the top edge. The ball must travel from the left edge to the right edge of the rectangle without intersecting any of the obstacles (see Fig. 1(b)).

The final trajectory of the shot is given by two scalars, $c$ and $d$, which denote the $y$-coordinates along the rectangle’s left and right sides, respectively, through which the ball passes (see Fig. 1(a) lower). You do not expect to hit the hole on the first shot. Instead, your objective is to get the ball as close to the center as possible. In particular, you want the value of $d$ where the ball exits the rectangle to be as close to $w/2$ as possible.

Present an algorithm which determines whether a straight-line path exists from the left to right edge of the rectangle that avoids all the obstacles. If such a path exists, determine the trajectory that places $d$ as close as possible to $w/2$. Your algorithm should run in expected time $O(m + n)$.

Figure 1: Miniature golf.
After your golf career failed, you turn to (indoor) tennis. You want to determine how best to hit a lob shot, which travels high enough to avoid the players on the other side of the net but low enough to avoid hitting lighting fixtures hanging from the ceiling. Let’s just consider the problem in two-dimensional space, where the floor is the $x$-axis, and the $y$-axis is vertical (see Fig. 2(a)). The lob needs to clear the net, which is of height $h$ along the $y$-axis. The lob needs to pass above positions likely occupied by the opposing player, which are given as a set of points $P = \{p_1, \ldots, p_m\}$, and it must pass below a set of locations of light fixtures $Q = \{q_1, \ldots, q_n\}$, where $p_i = (p_{i,x}, p_{i,y})$, and $q_j = (q_{j,x}, q_{j,y})$. Finally, it needs to land within the court, which is of length $\ell$.

By standard physics, the shot will travel along a parabolic path, given by the equation $y = -gx^2 + bx + c$ of a parabola, where $g$ is a fixed positive constant determined by the force of gravity (e.g., 32 ft/sec$^2$), and reals $b$ and $c$ are based on the initial location, direction, and speed with which the ball is hit. Present an algorithm which determines whether it is possible to find real values $b$ and $c$ to satisfy all the following requirements (see Fig. 2(b)). The ball’s trajectory must:

(a) pass above the net of height $h$ along the $y$-axis
(b) pass above all the opposing player points in $P$
(c) pass below all the lighting fixture points in $Q$
(d) land on the ground within the court (within distance $\ell$ of the net)

If such a shot is exists, return the one that travels as high as possible above the net along the $y$-axis. Your algorithm should run in expected time $O(m + n)$.

**Problem 2.** Consider the segments shown in Fig. 3.

(a) Show the (final) trapezoidal map for these segments, assuming they are inserted in the order $\langle s_1, s_2, s_3 \rangle$. (We have given you the map after inserting the first two segments, so you only need to show the result after inserting $s_3$.)

(b) Show the point-location data structure resulting from the construction given in class, assuming the insertion order from part (a). (We have given you the point-location data structure after inserting $s_1$ and $s_2$, so you only need to show the result after inserting $s_3$.) We will give partial credit if your data structure works correctly, even though it does not match the construction given in class.

Please follow the convention given in class for the node structure. (In particular, for $y$-nodes, the left (resp., right) child corresponds to the region above (resp., below) the segment.)
Problem 3. (TBD)
Problem 4. (TBD)
Problem 5. (TBD)
Challenge Problem. (TBD)
Guidance for Writing LP Reductions: In a linear programming (LP) reduction, you should explain the following:

- how the solution space is modeled as a vector (and in what dimension),
- what are the constraints,
- what is the objective function (and express it as a vector)
- how to interpret the result (including the cases feasible, unbounded, infeasible)

Here is an example.

Sample Problem: Present an efficient algorithm which given two sets of points $R = \{r_1, \ldots, r_n\}$ and $B = \{b_1, \ldots, b_n\}$, both in $\mathbb{R}^3$, determines whether there exists a plane $h$ in $\mathbb{R}^3$ such that all the points of $R$ lie on or above $h$ and all the points of $B$ lie on or below $h$.

Sample solution: We reduce the problem to linear programming in $\mathbb{R}^3$. Let’s assume that each $r_i \in R$ is given in coordinate form as $(r_{i,x}, r_{i,y}, r_{i,z})$ and similarly for $B$. Let’s model $h$ by the equation $z = ax + dy + e$, for some real parameters $a$, $d$, and $e$. To enforce the condition that each $r_i$ lies on or above $h$ and each $b_j$ lies on or below it, we add the constraints

\[
\begin{align*}
    r_{i,z} &\geq ar_{i,x} + dr_{i,y} + e, \quad \text{for } 1 \leq i \leq n \\
b_{j,z} &\leq ab_{j,x} + db_{j,y} + e, \quad \text{for } 1 \leq j \leq n.
\end{align*}
\]

We then invoke LP with $2n$ constraints in $\mathbb{R}^3$ (with the variables $(a, d, e)$). Since this is a yes-no answer, we don’t really care about the objective function. We can set it arbitrarily, for example, “maximize $e$” (which is equivalent to using the objective vector $c = (0, 0, 1)$).

If we wish to be even more formal (which is not usually required), we can express the LP in standard form as maximizing $c^T$ form as $Ax \leq b$, where $x$ is the symbolic vector $(a, d, e) \in \mathbb{R}^3$, and $A$ and $b$ can be expressed as

\[
\begin{bmatrix}
    r_{1,x} & r_{1,y} & 1 \\
    \vdots & \vdots & \vdots \\
    r_{m,x} & r_{m,y} & 1 \\
    -b_{1,x} & -b_{1,y} & -1 \\
    \vdots & \vdots & \vdots \\
    -b_{n,x} & -b_{n,y} & -1
\end{bmatrix}
\begin{bmatrix}
a \\
d \\
e
\end{bmatrix}
\leq
\begin{bmatrix}
r_{1,z} \\
\vdots \\
r_{m,z} \\
-b_{1,z} \\
\vdots \\
-b_{n,z}
\end{bmatrix}
\]

We interpret the LP’s result as follows. If the result is “infeasible”, then we know that no such plane exists. If the answer is “feasible” or “unbounded”, then we assert that such a plane exists (assuming general position). This is clearly true if the result is “feasible”, since we can just take $h$ to be the plane associated with the optimum vertex $(a, d, e)$.

If the result is “unbounded”, then the plane is vertical, but there exists a perturbation such that $R$ lies above and $B$ lies below.