CS 498 TC ✧ Computational Geometry ✧ Spring 2026

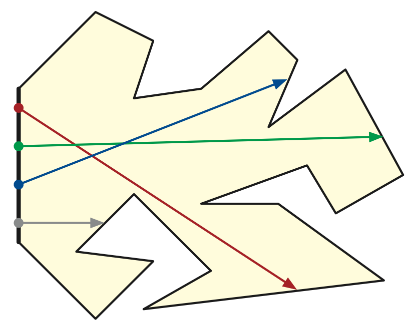
Homework 3

Due Tuesday, April 14, 2026 at 8pm

---

1. Suppose you are given a simple polygon P with a single vertical edge e. Describe and analyze a data structure that can efficiently answer queries of the following form: Given a ray r whose basepoint lies on e, find the first edge of P that r hits. For full credit, your data structure should use $O(n)$ expected space, and your query algorithm should run in $O(\log n)$ expected time. *[Hint: Reduce to a problem you've seen before!]*

   Without loss of generality, you can assume that the edge e lies on the x-axis, the query ray r points into the interior of P, and the query ray r does not contain any vertex of P. Don't worry about the time to construct the data structure.



2. Suppose you are given n points in the plane, in general position. These points are the centers of circular disks, all with the same radius r.

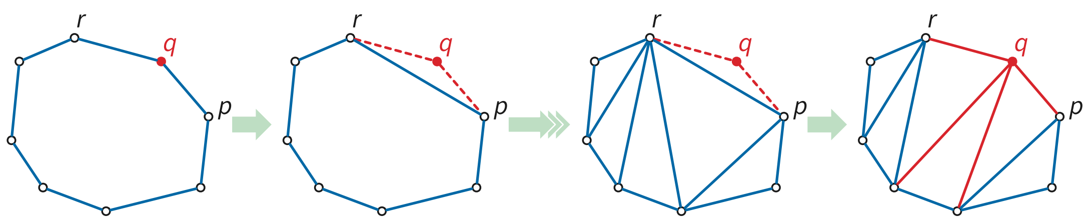
   (a) Suppose you are also given the radius r. Describe and analyze an efficient algorithm to determine whether the union of these disks is connected.

   (b) Now suppose you are *not* given the radius r. Find the minimum radius r such that the union of radius-r disks centered at the given points is connected. (Yes, this algorithm immediately implies a one-line solution to part (a).)

   For full credit, both algorithms should run in $O(n \log n)$ time. Don't forget to prove that your algorithms are correct!

3. Let P be a convex polygon in the plane with n vertices, represented as a circular doubly-linked list of vertices in counterclockwise order. The following randomized incremental algorithm constructs the Delaunay triangulation of the vertices of P.

```
CONVEXDELAUNAY(P):
  if P is a triangle
      return P
  else
      q ← random vertex of P
      p ← pred(q);  r ← succ(q)
      P′ ← P − pq − qr + pr
      T′ ← CONVEXDELAUNAY(P′)
      T ← T′ + pq + qr
      repair T by flipping non-Delaunay edges opposite q
      return T
```

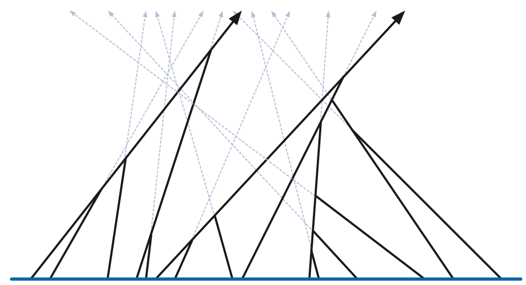


(a) Prove that the expected number of flips performed by this algorithm is $O(n)$.

(b) We already know from Lawson's algorithm that we can perform the necessary flips in $O(1)$ time each. But how do we choose the random vertex q in $O(1)$ time?

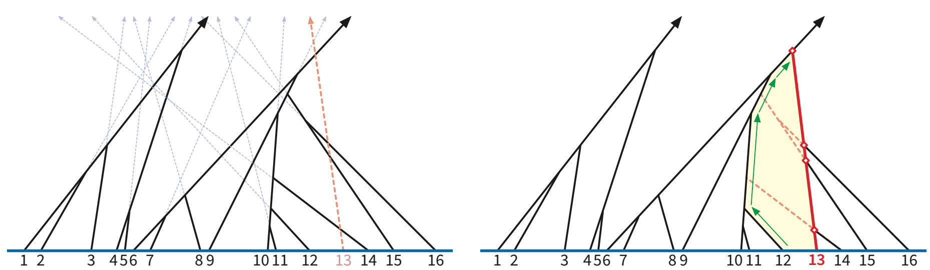
4. One way to solve the "Formula 6–7" problem from the midterm is to build a variant of motorcycle graphs from Homework 2.

   We model each car as a moving point $p_i(t) = (x_i + t \cdot u_i, t)$ whose x-coordinate is the position of car i at time t, and the y-coordinate is the current time t. Each moving point p_i traces out a line segment s_i in the plane. Car i passes car j at time t if and only if $p_i(t) = p_j(t)$ and $u_i > u_j$ (or equivalently, $x_i < x_j$). Car j can never be in the lead after being passed, so we can delete the moving point p_j, terminating its segment s_j. If a car is never passed, its segment is actually a ray extending to infinity. Call the planar straight-line graph consisting of the x-axis and the segments/rays s_i the *6-7-graph* of the moving points.



   A sweep-line algorithm to construct 6-7-graphs in $O(n \log n)$ time follows almost immediately from its definition. The goal of this problem is to work through part of the analysis of a natural randomized incremental algorithm.

   If there are no moving points, there is nothing to do. Otherwise, choose one moving point p_j at random, recursively construct the 6-7-graph of the other $n-1$ moving points, and then insert the final segment s_j. This segment terminates at its lowest intersection with another segment s_i such that $x_i < x_j$ (or equivalently, $u_i > u_j$). Thus, for every other segment s_k that intersects s_j we must have $x_k > x_j$ (or equivalently, $u_k < u_j$); we shorten those segments to terminate on s_j. See the example below.



   (a) What is the *exact* expected number of segments that terminate on s_j, as a function of n? In the example above, three segments terminate on the red segment s_{13}.

   (b) Each segment in the 6-7-graph has a unique face to its left. What is the *exact* expected number of segments s_i that touch the boundary of the face to the left of s_j?. In the example above, the shaded face to the left of s_{13} is bounded by five segments.

   (c) **To think about later, not for submission:** Suppose the moving points are initially sorted by their starting positions. Fill in the details of the randomized incremental construction algorithm, and show that it runs in ***O*(*n*) *expected time***. In particular, describe the data structure that you use to represent the evolving 6-7 graph.

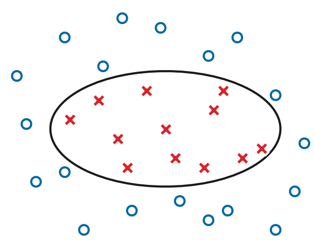
5. Explain how to solve each of the following problems in linear expected time, by reducing them to $O(1)$ instances of fixed-dimensional linear programming. To describe any linear program, you should provide the following information:

   - An explicit list of variables, each with an English description.
   - An explicit objective vector, with an English description.
   - An explicit list of linear inequalities, each with an English description.
   - How to interpret an optimal solution if it exists, and how to interpret unboundedness and infeasibility.

   (a) Suppose you are given a set of n red points and n blue points in the plane. Find an *proper axis-aligned ellipse* that has all red points *strictly* inside and all blue points *strictly* outside, or correctly report that no such ellipse exists. A proper axis-aligned ellipse is the locus of all points satisfying the equation

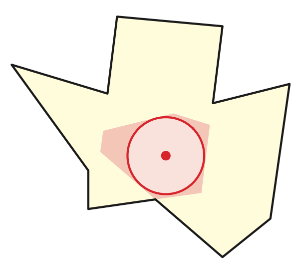
   $$Ax^2 + By^2 + Cx + Dy + E = 0$$

   for some real coefficients A, B, C, D, E, where A and B are *strictly positive*.[1] Points (x, y) such that $Ax^2 + By^2 + Cx + Dy + E$ is positive or negative lie inside or outside the ellipse, respectively. *[Hint: Consider the lifting map* $(x, y) \mapsto (x, y, x^2, y^2)$*.]*



   (b) Let P be a simple polygon in the plane, represented as a cyclic sequence of vertices $p_1, p_2, \ldots, p_n$ in counterclockwise order. Recall that an interior point q is called a *guard* if it can "see" the entire boundary of P, or equivalently, if the triple (q, p_i, p_{i+1}) is oriented counterclockwise for every index i. The *kernel* of P is the set of all guards of P.

   Find the largest circle inside the kernel of P. Equivalently, find a guard of P that is furthest aways from the closest line through an edge $p_i p_{i+1}$. *[Hint: The area of any triangle is* $\frac{1}{2}$*base* $\cdot$ *height.]*



[1] If $A = 0$ and $B > 0$, this equation describes a parabola with a horizontal axis; if $A > 0$ and $B = 0$, it describes parabola with a vertical axis; and if $A = B = 0$, it describes a straight line. These are all *degenerate* axis-aligned ellipses.