CS 498 TC ✧ Computational Geometry ✧ Spring 2021

Final Exam

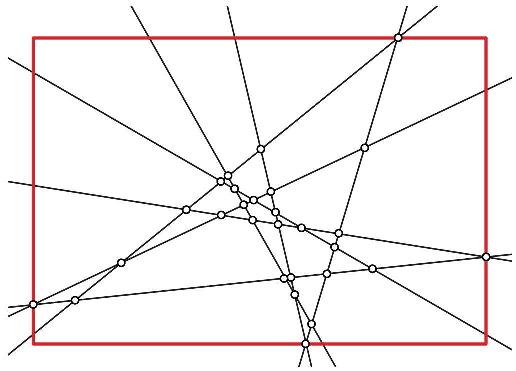
Due Monday, May 10, 2021 at 10pm

✧ Important instructions — Please read carefully! ✧

- **Don't panic!**
- **You have at most three hours to complete this exam and upload your solutions.** The clock started when you opened the Gradescope assignment. All solutions must be uploaded before 10pm (Central Standard Time) on Monday, May 10.
- The exam is designed to be completed on paper in 90 minutes. There are four problems.
- **This is an open-book-but-closed-everything-else exam.** You are welcome to use any of this semester's official course materials (textbook, notes, lecture scribbles, homework solutions) and anything you wrote yourself before starting the exam. All other resources, including materials from other classes and/or previous semesters, other textbooks, other internet resources, and (most importantly) other people, are not permitted.
- If you use a standard algorithm or data structure from this class, or from any prerequisite class (such as CS 374, CS 225, CS 173, or CS 125) as a black box, ***please do not*** give a detailed description; just name the algorithm or data structure. (For example: "balanced binary search tree" or "Gaussian elimination" or "Chan's algorithm" or "the plane-sweep algorithm to count segment intersections".)
- Similarly, if you use a minor variant of an algorithm from this class or a prerequisite class, please describe only the necessary modifications.
- You can implicitly assume general position without comment.
- **I cannot answer questions about the exam while the exam is in progress.** If you need to make additional assumptions to solve an exam problem, please state those assumptions clearly in your solution. Do not discuss the exam with anyone until after Monday at 10pm.
- The exam is not proctored. I am trusting you to take the exam by yourself, with no help from anyone, within the declared time limits. The exam is first and foremost a mechanism to give you honest feedback on your mastery of the course material; please treat it as such. All academic integrity policies are still in place.
- Automatic polygon meter / Analog toy computer item / Triangulate me, moot copy
  Promote magnetic layout / Goatee community portal / Oatmeal recomputing toy
  Get your calm emotion, Pat. / Immature galoot potency / My moot rectangle utopia
  Cut lame emotion; go party! / Young male potato metric / Gourmet emoticon at play
  Trip out, ecology meat-man. / A tiny computer loot game / Cue topology mantra time
  An ugly poetic motor team / Magical poem torn out yet / Automatic mangle poetry

---

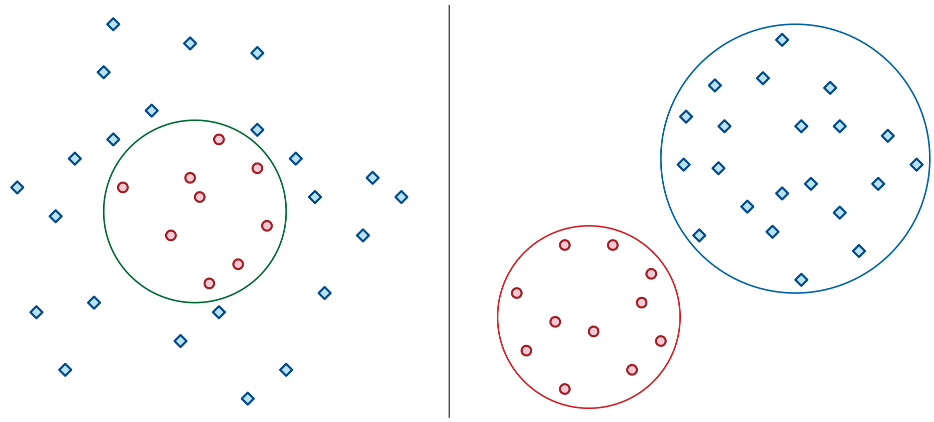
1. Suppose you are given a set L of n lines in the plane in general position. Describe an algorithm to compute the smallest axis-aligned rectangle that contains all $\binom{n}{2}$ vertices of the arrangement of L. For full credit, your algorithm should run in $O(n \log n)$ time. *[Hint: What is the dual of the rightmost arrangement vertex?]*



2. Suppose you are given a set of n red points and n blue points in the plane.

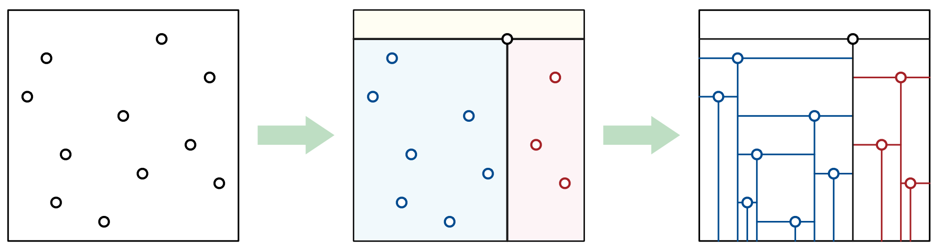
   (a) Describe and analyze an algorithm to determine if there is a circle with all red points inside and all blue points outside.

   (b) Describe and analyze an algorithm to determine if there are two *disjoint* circles, one containing all the red points but no blue points, and the other containing all the blue points but no red points. *[Hint: This is easier than it looks.]*



3. Let P be a set of n points with distinct x- and y-coordinates in the interior of some bounding rectangle R. The *Cartesian subdivision* of P is defined recursively as follows.

   - If $P = \varnothing$, the subdivision is just the rectangle R.
   - Otherwise, let p be the highest point in P.
     - First split R into two smaller rectangles R^+ and R^- using a horizontal segment through p.
     - Then split the lower rectangle R^- into two smaller rectangles $R_<$ and $R_>$ using a vertical segment downward from p.
     - Finally, recursively construct Cartesian subdivisions of the points inside the two lower rectangles $R_<$ and $R_>$.

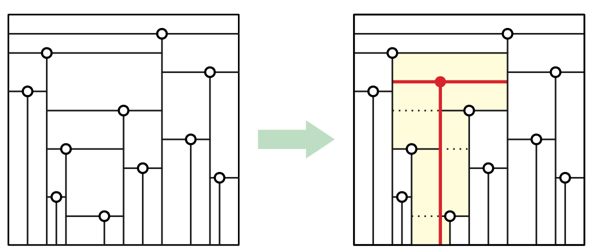


Recursively defining the Cartesian subdivision.

   (a) What is the *exact* number of vertices, edges, and faces in a Cartesian subdivision of n points?

   (b) *Briefly sketch* an algorithm to insert a new point into P and update its Cartesian subdivision. (Assume that you are given both the coordinates of the new point and the face of the subdivision that contains it.)

   (c) Prove that if n points are inserted *in random order* into an initially empty Cartesian subdivision, then the expected number of structural changes for each insertion is $O(1)$.



Inserting a point into a Cartesian subdivision. Modified faces of the subdivision are highlighted.

4. The *width* of a set P of points in the plane is the minimum distance between two parallel lines that enclose P. (The distance between the lines is measured orthogonally, not vertically; rotating a point set does not change its width.) Describe and analyze an algorithm to compute the width of a given set of n points in the plane.