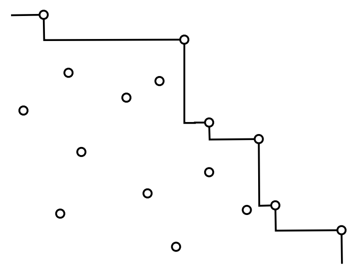
1. Let P be a set of n points in the plane. Recall that a point $p \in P$ is *Pareto-optimal* if no other point is both above and to the right of p. Intuitively, the sorted sequence of Pareto-optimal points describes a *staircase* with all the points in P below and to the left. Your task is to describe some algorithms that compute this staircase.



The staircase of a set of points

   (a) Describe an algorithm to compute the staircase of P in $O(nh)$ time, where h is the number of Pareto-optimal points.

   (b) Describe a divide-and-conquer algorithm to compute the staircase of P in $O(n \log n)$ time. *[Hint: I know of at least two different ways to do this.]*

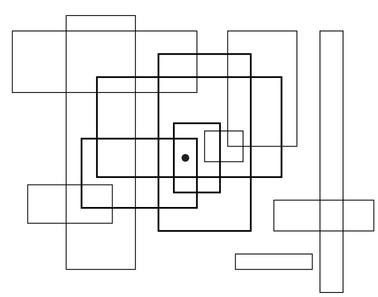
   *(c) Describe an algorithm to compute the staircase of P in $O(n \log h)$ time, where h is the number of Pareto-optimal points. *[Hint: I know of at least two different ways to do this.]*

   (d) Finally, suppose the points in P are already given in sorted order from left to right. Describe an algorithm to compute the staircase of P in $O(n)$ time. *[Hint: I know of at least two different ways to do this.]*

2. Let R be a set of n rectangles in the plane.

   (a) Describe and analyze a plane sweep algorithm to decide, in $O(n \log n)$ time, whether any two rectangles in R intersect.

   *(b) The *depth* of a point is the number of rectangles in R that contain that point. The *maximum depth* of R is the maximum, over all points p in the plane, of the depth of p. Describe a plane sweep algorithm to compute the maximum depth of R in $O(n \log n)$ time.

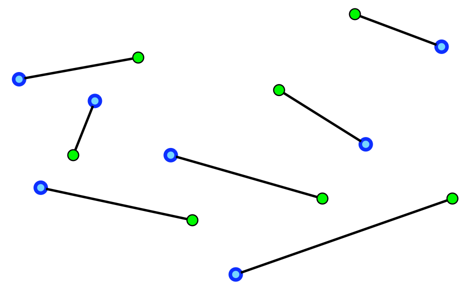


A point with depth 4 in a set of rectangles.

   (c) Describe and analyze a polynomial-time reduction from the maximum depth problem in part (b) to MaxClique: Given a graph G, how large is the largest clique in G?

   (d) MaxClique is NP-hard. So does your reduction imply that P=NP? Why or why not?

3. Let G be a set of n green points, called "Ghosts", and let B be a set of n blue points, called "ghostBusters", so that no three points lie on a common line. Each Ghostbuster has a gun that shoots a stream of particles in a straight line until it hits a ghost. The Ghostbusters want to kill all of the ghosts at once, by having each Ghostbuster shoot a different ghost. It is **very important** that the streams do not cross.



A non-crossing matching between 7 ghosts and 7 Ghostbusters

(a) Prove that the Ghostbusters can succeed. More formally, prove that there is a collection of n non-intersecting line segments, each joining one point in G to one point in B. *[Hint: Think about the set of joining segments with minimum total length.]*

(b) Describe and analyze an algorithm to find a line ℓ that passes through one ghost and one Ghostbuster, so that same number of ghosts as Ghostbusters are above ℓ.

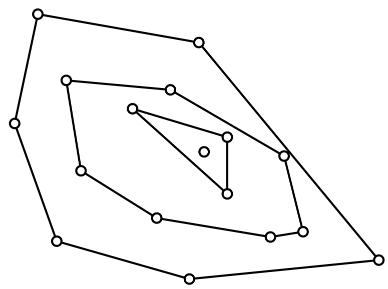
*(c) Describe and analyze an algorithm to find a line ℓ such that exactly half the ghosts and exactly half the Ghostbusters are above ℓ. (Assume n is even.)

(d) Using your algorithm for part (b) or part (c) as a subroutine, describe and analyze an algorithm to find the line segments described in part (a). (Assume n is a power of two if necessary.)

Spengler: *Don't cross the streams.*
Venkman: *Why?*
Spengler: *It would be bad.*
Venkman: *I'm fuzzy on the whole good/bad thing. What do you mean "bad"?*
Spengler: *Try to imagine all life as you know it stopping instantaneously and every molecule in your body exploding at the speed of light.*
Stantz: *Total protonic reversal!*
Venkman: *That's bad. Okay. Alright, important safety tip, thanks Egon.*

— Dr. Egon Spengler (Harold Ramis), Dr. Peter Venkman (Bill Murray), and Dr. Raymond Stanz (Dan Aykroyd), *Ghostbusters*, 1984

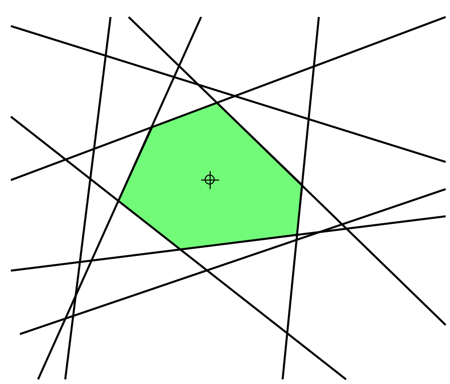
4. The *convex layers* of a point set P consist of a series of nested convex polygons. The convex layers of the empty set are empty. Otherwise, the first layer is just the convex hull of P, and the remaining layers are the convex layers of the points that are not on the convex hull of P.



The convex layers of a set of points.

Describe and analyze an efficient algorithm to compute the convex layers of a given n-point set. For full credit, your algorithm should run in $O(n^2)$ time.

5. Suppose we are given a set of n lines in the plane, where none of the lines passes through the origin $(0, 0)$ and at most two lines intersect at any point. These lines divide the plane into several convex polygonal regions, or *cells*. Describe and analyze an efficient algorithm to compute the cell containing the origin. The output should be a doubly-linked list of the cell's vertices. *[Hint: There are literally dozens of solutions. One solution is to reduce this problem to the convex hull problem. Every other solution looks like a convex hull algorithm.]*



The cell containing the origin in an arrangement of lines.