CS 373: Combinatorial Algorithms, Spring 1999

`http://www-courses.cs.uiuc.edu/~cs373`

Homework 4 (due Thu. Apr. 1, 1999 by noon)

Name:	
Net ID:	Alias:

Everyone must do the problems marked ▶. Problems marked ▷ are for 1-unit grad students and others who want extra credit. (There's no such thing as "partial extra credit"!) Unmarked problems are extra practice problems for your benefit, which will not be graded. Think of them as potential exam questions.

Hard problems are marked with a star; the bigger the star, the harder the problem.

Note: When a question asks you to "give/describe/present an algorithm", you need to do four things to receive full credit:

1. (New!) If not already done, model the problem appropriately. Often the problem is stated in real world terms; give a more rigorous description of the problem. This will help you figure out what is assumed (what you know and what is arbitrary, what operations are and are not allowed), and find the tools needed to solve the problem.
2. Design the most efficient algorithm possible. Significant partial credit will be given for less efficient algorithms, as long as they are still correct, well-presented, and correctly analyzed.
3. Describe your algorithm succinctly, using structured English/pseudocode. We don't want full-fledged compilable source code, but plain English exposition is usually not enough. Follow the examples given in the textbooks, lectures, homeworks, and handouts.
4. Justify the correctness of your algorithm, including termination if that is not obvious.
5. Analyze the time and space complexity of your algorithm.

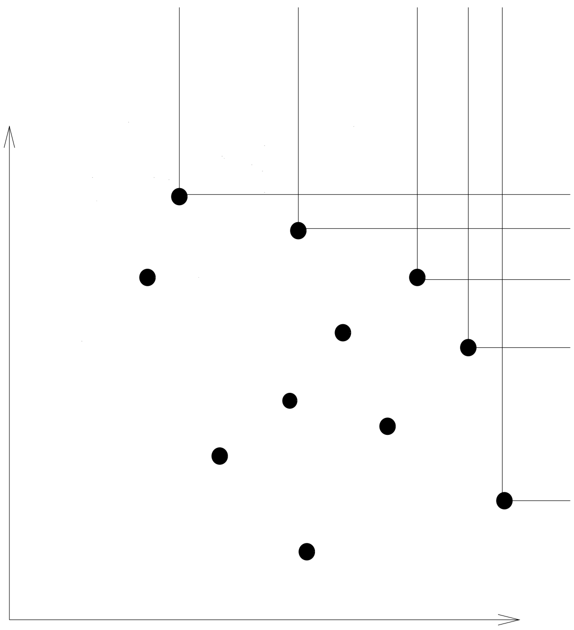
Undergrad/.75U Grad/1U Grad Problems

▶1. (5 pts total) Collinearity
Give an $O(n^2 \log n)$ algorithm to determine whether any three points of a set of n points are collinear. Assume two dimensions and *exact* arithmetic.

▶2. (4 pts, 2 each) Convex Hull Recurrence
Consider the following generic recurrence for convex hull algorithms that divide and conquer:

$$T(n, h) = T(n_1, h_1) + T(n_2, h_2) + O(n)$$

where $n \geq n_1 + n_2, h = h_1 + h_2$ and $n \geq h$. This means that the time to compute the convex hull is a function of both n, the number of input points, and h, the number of convex hull vertices. The splitting and merging parts of the divide-and-conquer algorithm take $O(n)$ time. When n is a constant, $T(n, h)$ is $O(1)$, but when h is a constant, $T(n, h)$ is $O(n)$. Prove that for both of the following restrictions, the solution to the recurrence is $O(n \log h)$:



(a) $h_1, h_2 < \frac{3}{4}h$

(b) $n_1, n_2 < \frac{3}{4}n$

▶3. (5 pts) Circle Intersection
Give an $O(n \log n)$ algorithm to test whether any two circles in a set of size n intersect.

▶4. (5 pts total) Staircases
You are given a set of points in the first quadrant. A *left-up* point of this set is defined to be a point that has no points both greater than it in both coordinates. The left-up subset of a set of points then forms a *staircase* (see figure).

(a) (3 pts) Give an $O(n \log n)$ algorithm to find the staircase of a set of points.

(b) (2 pts) Assume that points are chosen uniformly at random within a rectangle. What is the average number of points in a staircase? Justify. Hint: you will be able to give an exact answer rather than just asymptotics. You have seen the same analysis before.

Only 1U Grad Problems

▷1. (6 pts, 2 each) Ghostbusters and Ghosts
A group of n ghostbusters is battling n ghosts. Each ghostbuster can shoot a single energy beam at a ghost, eradicating it. A stream goes in a straight line and terminates when it hits a ghost. The ghostbusters must all fire at the same time and no two energy beams may cross. The positions of the ghosts and ghostbusters is fixed in the plane (assume that no three points are collinear).

(a) Prove that for any configuration ghosts and ghostbusters there exists such a non-crossing matching.

(b) Show that there exists a line passing through one ghostbuster and one ghost such that the number of ghostbusters on one side of the line equals the number of ghosts on the same side. Give an efficient algorithm to find such a line.

(c) Give an efficient divide and conquer algorithm to pair ghostbusters and ghosts so that no two streams cross.

Practice Problems

1. Basic Computation (assume two dimensions and *exact* arithmetic)

   (a) Intersection: Extend the basic algorithm to determine if two line segments intersect by taking care of *all* degenerate cases.

   (b) Simplicity: Give an $O(n \log n)$ algorithm to determine whether an n-vertex polygon is simple.

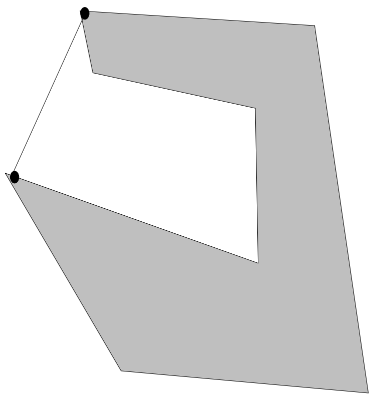
   (c) Area: Give an algorithm to compute the area of a simple n-polygon (not necessarily convex) in $O(n)$ time.

   (d) Inside: Give an algorithm to determine whether a point is within a simple n-polygon (not necessarily convex) in $O(n)$ time.

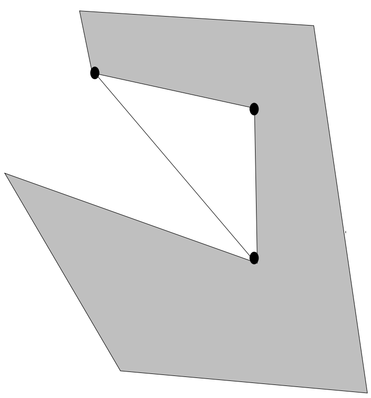
2. External Diagonals and Mouths

   (a) A pair of polygon vertices defines an *external diagonal* if the line segment between them is completely outside the polygon. Show that every nonconvex polygon has at least one external diagonal.

   (b) Three consective polygon vertices p, q, r form a *mouth* if p and r define an external diagonal. Show that every nonconvex polygon has at least one mouth.



An external diagonal



A mouth

3. On-Line Convex Hull
   We are given the set of points one point at a time. After receiving each point, we must compute the convex hull of all those points so far. Give an algorithm to solve this problem in $O(n^2)$ (We could obviously use Graham's scan n times for an $O(n^2 \log n)$ algorithm). Hint: How do you maintain the convex hull?

4. Another On-Line Convex Hull Algorithm
   (a) Given an n-polygon and a point outside the polygon, give an algorithm to find a tangent.
   *(b) Suppose you have found both tangents. Give an algorithm to remove the points from the polygon that are within the angle formed by the tangents (as segments!) and the opposite side of the polygon.
   (c) Use the above to give an algorithm to compute the convex hull on-line in $O(n \log n)$

$^\bigstar$5. Order of the size of the convex hull
   The convex hull on $n \geq 3$ points can have anywhere from 3 to n points. The average case depends on the distribution.
   (a) Prove that if a set of points is chosen randomly within a given rectangle. then the average size of the convex hull is $O(\log n)$.
   (b) Prove that if a set of points is chosen randomly within a given circle. then the average size of the convex hull is $O(\sqrt{n})$.