CS 373: Combinatorial Algorithms, Spring 2001

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

Homework 4 (due Thu. March 29, 2001 at 11:59:59 pm)

Name:		
Net ID:	Alias:	U $^3/_4$ 1

Name:		
Net ID:	Alias:	U $^3/_4$ 1

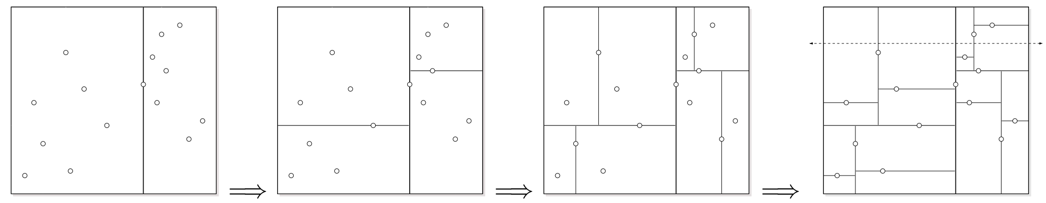
Name:		
Net ID:	Alias:	U $^3/_4$ 1

Homeworks may be done in teams of up to three people. Each team turns in just one solution, and every member of a team gets the same grade. Since 1-unit graduate students are required to solve problems that are worth extra credit for other students, **1-unit grad students may not be on the same team as 3/4-unit grad students or undergraduates.**

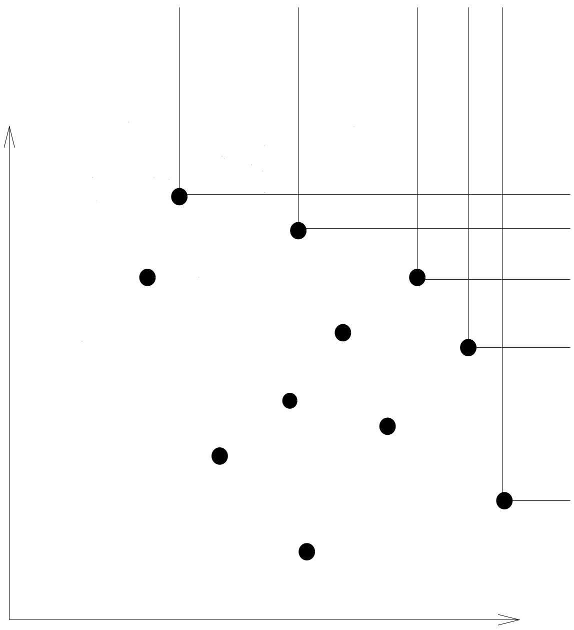
Neatly print your name(s), NetID(s), and the alias(es) you used for Homework 0 in the boxes above. Please also tell us whether you are an undergraduate, 3/4-unit grad student, or 1-unit grad student by circling U, $^3/_4$, or 1, respectively. Staple this sheet to the top of your homework.

Required Problems

1. Suppose we have n points scattered inside a two-dimensional box. A *kd-tree* recursively subdivides the rectangle as follows. First we split the box into two smaller boxes with a *vertical* line, then we split each of those boxes with *horizontal* lines, and so on, always alternating between horizontal and vertical splits. Each time we split a box, the splitting line partitions the rest of the interior points *as evenly as possible* by passing through a median point inside the box (*not* on the boundary). If a box doesn't contain any points, we don't split it any more; these final empty boxes are called *cells*.



Successive divisions of a kd-tree for 15 points. The dashed line crosses four cells.



An example staircase as in problem 3.

(a) How many cells are there, as a function of n? Prove your answer is correct.

(b) In the worst case, *exactly* how many cells can a horizontal line cross, as a function of n? Prove your answer is correct. Assume that $n = 2^k - 1$ for some integer k.

(c) Suppose we have n points stored in a kd-tree. Describe an algorithm that counts the number of points above a horizontal line (such as the dashed line in the figure) in $O(\sqrt{n})$ time.

*(d) *[Optional: 5 pts extra credit]* Find an algorithm that counts the number of points that lie inside a rectangle R and show that it takes $O(\sqrt{n})$ time. You may assume that the sides of the rectangle are parallel to the sides of the box.

2. Circle Intersection *[This problem is worth 20 points]*
Describe an algorithm to decide, given n circles in the plane, whether any two of them intersect, in $O(n \log n)$ time. Each circle is specified by three numbers: its radius and the x- and y-coordinates of its center.

   We only care about intersections between circle boundaries; concentric circles do not intersect. What general position assumptions does your algorithm require? *[Hint: Modify an algorithm for detecting line segment intersections, but describe your modifications very carefully! There are at least two very different solutions.]*

3. 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).

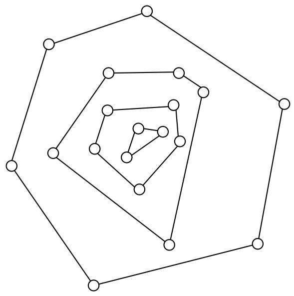
   (a) Prove that left-up points do not necessarily lie on the convex hull.

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

(c) 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.

4. Convex Layers
Given a set Q of points in the plane, define the *convex layers* of Q inductively as follows: The first convex layer of Q is just the convex hull of Q. For all $i > 1$, the ith convex layer is the convex hull of Q after the vertices of the first $i - 1$ layers have been removed.

   Give an $O(n^2)$-time algorithm to find all convex layers of a given set of n points.

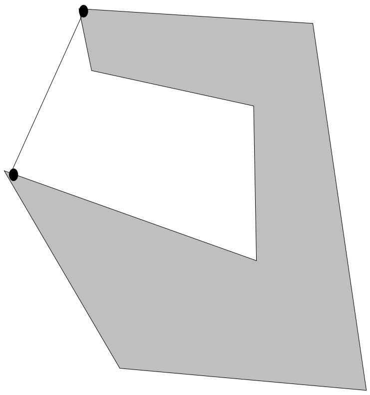


A set of points with four convex layers.

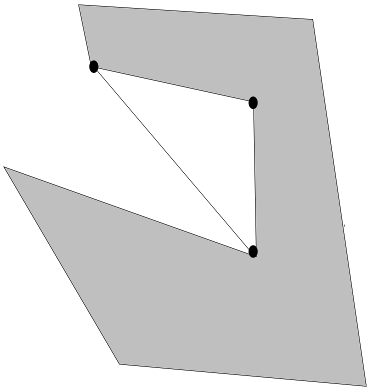
5. *[This problem is required only for graduate students taking CS 373 for a full unit; anyone else can submit a solution for extra credit.]* Solve the travelling salesman problem for points in convex position (ie, the vertices of a convex polygon). Finding the shortest cycle that visits every point is easy – it's just the convex hull. Finding the shortest path that visits evey point is a little harder, because the path can cross through the interior.

   (a) Show that the optimal path cannot be one that crosses itself.

   (b) Describe an $O(n^2)$ time dynamic programming algorithm to solve the problem.



An external diagonal



A mouth

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.

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 $O(n \log n)$

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 $O(n^{1/3})$.

6. 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 (it would be bad). 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 of 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.