CMSC 754: Spring 2026 Dave Mount

Homework 1

Due by the start of class on Tuesday, Feb 10. (Submissions will be through Gradescope.) Late homeworks are not accepted (unless an extension has been prearranged) so please turn in whatever you have completed by the due date. Unless otherwise specified, you may assume that all inputs are given in *general position*.

Problem 1. Suppose that you are given a point set of size n and there are $h \geq 2$ points on the upper hull. As a function of n and h, what is the (exact) maximum number of orientation evaluations that is performed by Graham's scan (for just the upper hull) when run on this input set? (**Note:** Why "maximum"? The number of orientation tests is not strictly a function of n and h. The reason is that in Step 3(b) of the algorithm, the condition "$|S| \geq 2$" might fail, and as a result, the orientation test in the while loop would not be performed during such a step of the algorithm.)

Problem 2. Throughout this problem assume that the only way to access information about points is by computing the orientation of three points. (For example, you cannot access individual coordinates or compute dot products or other vector operations.) In each case, briefly justify your answer.

(a) You are given a set of four points in the plane $\{a, b, c, d\}$. Show how to determine whether d lies within the interior of a triangle $\triangle abc$ in the plane using orientation tests (see Fig. 1(a)). You do *not* know whether $\triangle abc$ is oriented clockwise or counterclockwise, but you may assume that a, b, and c are *not* collinear. Note that d might be collinear with two of the other points. For full credit, show that at most 3 orientation evaluations suffice.

(b) You are given four points in the plane $\{p, q, s, t\}$, no three of which are collinear. Show how to determine whether the line segment $\overline{pq}$ intersects the line segment $\overline{st}$ using orientation tests (see Fig. 1(b)). For full credit, show that this can be done using at most 4 orientation evaluations.

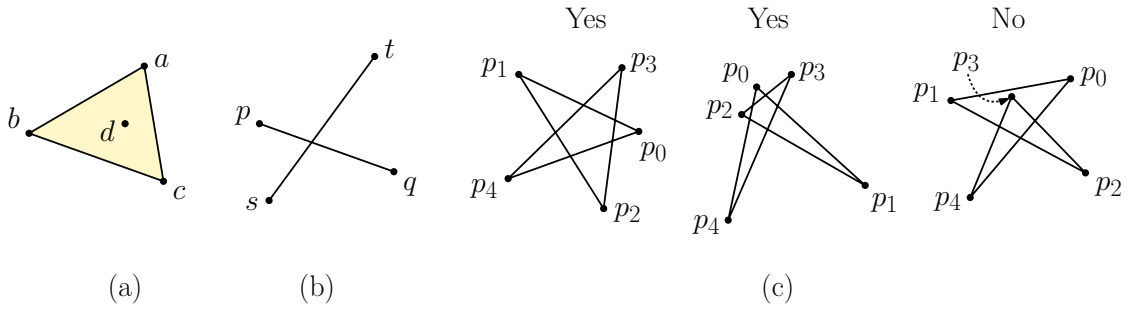

Figure 1: Using orientations for geometric properties.

(c) You are given a sequence of five points in the plane $\langle p_0, \ldots, p_4 \rangle$, no three of which are collinear. These define five line segments $\overline{p_i p_{i+1}}$, for $0 \leq i \leq 4$ (where indices are taken

mod 4, so $p_{4+1} = p_0$). We say that the sequence defines a *pentagram* if every pair of segments intersects, either at their endpoints (which happens for consecutive segments) or in their interiors. Show how to determine whether the configuration of points forms a pentagram using orientation tests (see Fig. 1(c)). For full credit, show that this can be done using at most 10 orientation evaluations. You are allowed to invoke your solution to part (b).

Problem 3. The objective of this problem is to explore possible variations in the guessing sequence for Chan's convex hull algorithm for an n-element point set P. Let h denote the size of the final convex hull. Recall that ConditionalHull(P, h^*) returns the convex hull of P if $h^* \geq h$ and "fail" otherwise, and the algorithm iteratively guesses values of h^* through repeated squaring. Here is a equivalent version of the one given in class.

(1) $i \leftarrow 1$; status $\leftarrow$ fail
(2) while (status == fail)
  (a) $h^* \leftarrow \min\left(2^{2^i}, n\right)$ // repeated squaring
  (b) status $\leftarrow$ ConditionalHull(P, h^*)
  (c) $i \leftarrow i + 1$
(3) return status

What if we do not use repeated squaring? In each of the following cases, indicate what the running time of Chan's algorithm *would be* had we replaced the expression in line (2a). (Express your answer using O-notation as a function of n and h.) In each case, explain how you derived your answer.

(a) $h^* \leftarrow \min(i^2, n)$ // quadratic progression
(b) $h^* \leftarrow \min(2^i, n)$ // repeated doubling

You may assume any standard results on summations, e.g., the Wikipedia page on summations.

Problem 4. (Optional—Ungraded) You are given a set of point in $\mathbb{R}^2$, $P = \{p_1, \ldots, p_n\}$. The points are contained in the 3-sided region, bounded by $y \geq 0$, and $0 \leq x \leq 1$ (see Fig. 2(a)). Your objective is to compute a line segment ℓ along the top that encloses all the points and generates the region with the smallest area.

To simplify the problem, we'll assume that the lower left and right corners of the 3-sided region, $(0, 0)$ and $(1, 0)$, are included as part of the input set P. This effectively means that any solution will have its endpoints on the left and right vertical walls.

(a) Let us assume that we have invoked Graham's scan and we have computed the upper hull U for P. Observe that the line ℓ defining the minimum enclosure must be a supporting line to U. (Otherwise we could decrease the area by lowering ℓ until it passes through a point of U).

Prove the following claim. Let ℓ be the line segment of a supporting line for U passing between the vertical sides of the enclosing region. Let p be the point of ℓ that lies on U

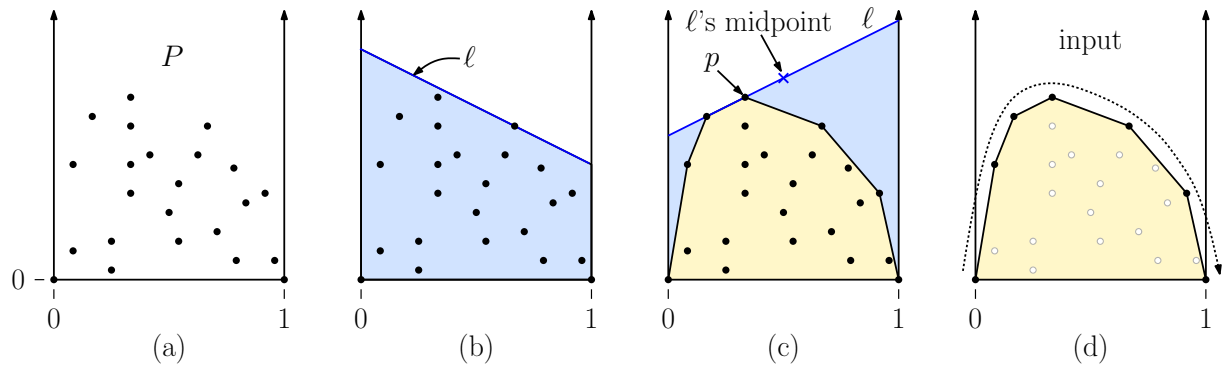

Figure 2: Computing the minimum-area enclosure for a set of points.

and is closest to ℓ's midpoint. (We have a choice in the selection of p only if ℓ is collinear with an edge of U.) Then ℓ is the area-minimizing support line if and only if p coincides with ℓ's midpoint.

For example, in Fig. 2(c), this line cannot be the area-minimizing line because the closest point p to ℓ's midpoint does not coincide with ℓ's midpoint.

(b) Using (a), present an algorithm that computes the line segment ℓ that produces the minimum-area enclosure. Assume that the upper hull U is given to you, sorted from left to right (see Fig. 2(d)). Your algorithm should run in time $O(\log m)$, where m is the number of points on U.

Note: Challenge problems are not graded as part of the homework. The grades are recorded separately. After final grades have been computed, I may "bump-up" a grade that is slightly below a cutoff threshold based on these extra points. (But there is no formal rule for this.)

Challenge Problem 1: Show that Problem 1(c) (pentagram) can be solved with fewer than 10 orientation tests. (**Hint:** I believe that the smallest number is between 6 and 9. If you think you can do it with 5 or fewer, please let me know, since I tried to get this but failed.)

Challenge Problem 2: Returning to Problem 3, what would be the performance of Chan's algorithm if we used the following (rather wonky) sequence of values?

$$h^* \leftarrow \min\left(\sqrt{2}^{\sqrt{2}^i}, n\right)$$

Some tips about writing algorithms: Throughout the semester, whenever you are asked to present an "algorithm," you should present three things: the algorithm, an informal proof of its correctness, and a derivation of its running time. Remember that your description is intended to be read by a human, not a compiler, so conciseness and clarity are preferred over technical details.

Unless otherwise stated, you may use any results from class, or results from any standard textbook on algorithms and data structures. (If the source is from outside of class, you must cite your sources.) Also, you may use results from geometry that: (1) have been mentioned in class,

(2) would be known to someone who knows basic geometry or linear algebra, or (3) is intuitively obvious. If you are unsure, please feel free to check with me.

Giving careful and rigorous proofs can be quite cumbersome in geometry, and so you are encouraged to use intuition and give illustrations whenever appropriate. Beware, however, that a poorly drawn figure can make certain erroneous hypotheses appear to be "obviously correct."

Throughout the semester, unless otherwise stated, you may assume that input objects are in *general position*. For example, you may assume that no two points have the same x-coordinate, no three points are collinear, no four points are cocircular. Also, unless otherwise stated, you may assume that any geometric primitive involving a constant number of objects each of constant complexity can be computed in $O(1)$ time.

Some tips on Gradescope submission: Homework submissions must be written clearly and easily readable after scanning. To guarantee this, please either typeset your solutions, write them using a note-taking application (like OneNote or Notability), or handwritten and scanned. If you hand-write your submission, please be sure that the text contrast is high. This can be assisted through the use of scanning software, like CamScanner. If you are unsure, you can do a trial submission through Gradescope, and ask us to check it.