CMSC 754: Fall 2026 Dave Mount

Homework 1

Due by the start of class on Tuesday, Sep 15. (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. 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 at most 4 orientation evaluations suffice.*

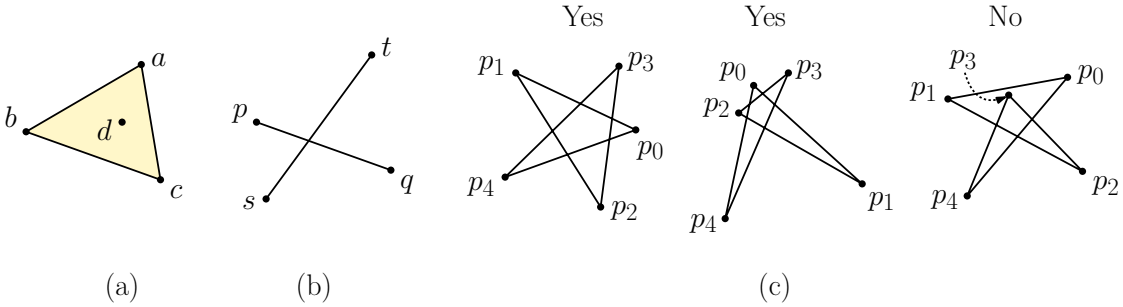

Figure 1: Using orientations for geometric properties.

(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.* (The optimum number is smaller. See the Challenge Problem below.) You are allowed to invoke your solutions to the previous parts.

Problem 2. In this problem, we will consider a simplified version of the Tangent Lemma from Lecture 3. Consider an upper convex chain P in the plane presented as sequence of vertices $\langle p_1, \ldots, p_n \rangle$ in increasing order of x-coordinates, where $p_i = (p_{i,x}, p_{i,y})$. Let q be a point that is to the right of all the points of P, that is, $q_x > p_{n,x}$.

Present an $O(\log n)$-time algorithm which, given P and q, computes the index j of a vertex p_j such that $\overrightarrow{qp_j}$ is an upper supporting line for the hull (see Fig. 2). You may assume that the points of P are stored in an array, so that index operations can be performed in constant time. For full credit, your algorithm should access points only through *orientation tests*.

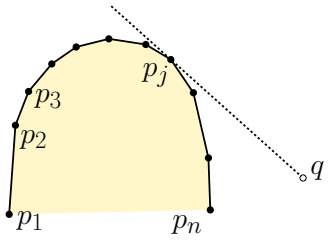

Figure 2: Tangent Lemma (simplified).

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.

- $i \leftarrow 0$; status $\leftarrow$ fail
- while (status = fail)
  - $i \leftarrow i + 1$
  - $h^* \leftarrow \min\left(2^{2^i}, n\right)$ // repeated squaring
  - (status, hull) $\leftarrow$ ConditionalHull(P, h^*)
- return hull

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) This problem essentially asks you to complete the missing step of Jarvis's convex hull algorithm. You are given a set of points $P = \{p_1, \ldots, p_n\}$ in $\mathbb{R}^2$ and two additional points q_0 and q_1. Among the points of P that lie strictly to the left of the

directed line $\overrightarrow{q_0q_1}$, compute the point p_j that minimizes the counterclockwise angle between the rays $\overrightarrow{q_0q_1}$ and $\overrightarrow{q_1p_j}$ (see Fig. 1(b)). If no point of P lies to the left of $\overrightarrow{q_0q_1}$, your algorithm should report this. Your solution should access the points only through orientation tests, and should run in $O(n)$ time. (Note: The notion of being "left" of a directed line means on the left side of the line with respect to an observer facing the line's direction.)

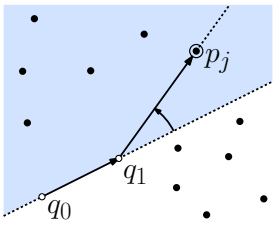

Figure 3: Jarvis implementation

Problem 5. (Optional—Ungraded) Consider a set $P = \{p_1, \ldots, p_n\}$ of points in the plane, where $p_i = (x_i, y_i)$. A *Pareto set* for P, denoted Pareto(P), (named after the Italian engineer and economist Vilfredo Pareto), is a subset of points p_i such that there is no $p_j \in P$ $(j \neq i)$ such that $x_j \geq x_i$ and $y_j \geq y_i$. That is, each point of Pareto(P) has the property that there is no point of P that is both to the right and above it (see Fig. 4).

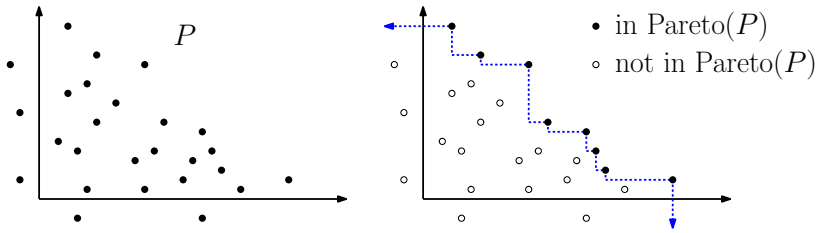

Figure 4: Pareto set

Pareto sets and convex hulls in the plane are similar in many respects. In this problem we will explore some of these connections.

(a) A point p lies on the convex hull of a set P if and only if there is a line passing through p such that all the points of P lie on one side of this line. Provide an analogous assertion for the points of Pareto(P) in terms of a different shape.

(b) Devise an analog of Graham's convex-hull algorithm for computing Pareto(P) in $O(n \log n)$ time. Briefly justify your algorithm's correctness and derive its running time. (You do not need to explain the algorithm "from scratch", that is, you can explain what modifications would be made the Graham's algorithm.)

(c) Devise an analog of the Jarvis march algorithm for computing Pareto(P) in $O(h \cdot n)$ time, where h is the cardinality of Pareto(P). (As in part (b), you can just explain the differences with Jarvis's algorithm.)

In each case, briefly justify the correctness of your algorithm, explain any data structures

that you make use of (unless they are standard), and briefly derive your algorithm's running time.

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.)

Challenge Problem 2: Returning to Problem 5 (Pareto sets), present an algorithm for computing Pareto(P) in $O(n \log h)$ time, where h is the cardinality of $Pareto(P)$. (I know of two solutions. One involves modifying Chan's algorithm. The other is conceptually simpler, but harder to analyze.)

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.