

Problem 1 (10 points)

Let $P = \{p_1, \dots, p_n\}$ and $P' = \{p'_1, \dots, p'_n\}$ be the vertex sets of two upper hulls in the plane. Each set is presented as a sequence of points sorted from left to right. Let $p_i = (x_i, y_i)$ and $p'_j = (x'_j, y'_j)$ denote the point coordinates. We assume that P lies entirely to the left of P' , meaning that there exists a value z such that for all i and j , $x_i < z < x'_j$.

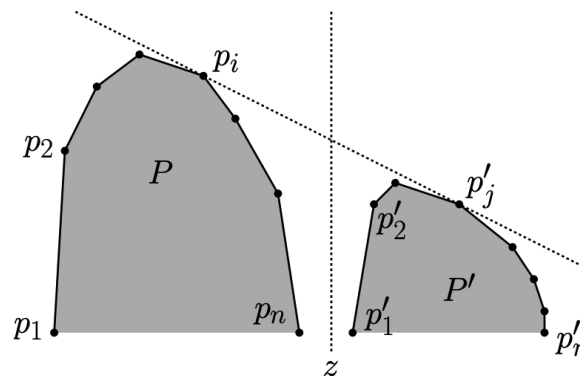


Figure 1: Problem 1: Computing the upper tangent of two hulls

Present an $O(\log n)$ -time algorithm which, given P and P' , compute the two points $p_i \in P$ and $p'_j \in P'$ such that their common support line passes through these two points.

Briefly justify your algorithm's correctness and drive its running time. (**Hint:** The correctness proof involves a case analysis. Please be careful, a poorly drawn figure may lead to an incorrect hypothesis.)

Problem 2 (20 points)

Consider a set $P = \{p_1, \dots, p_n\}$ of points in the plane, where $p_i = (x_i, y_i)$. A *Pareto set* for P , denoted $\text{Pareto}(P)$, (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 $\text{Pareto}(P)$ has the property that there is no point of P that is both to the right and above it.

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

1. (5 points) 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 $\text{Pareto}(P)$ in terms of a different shape.

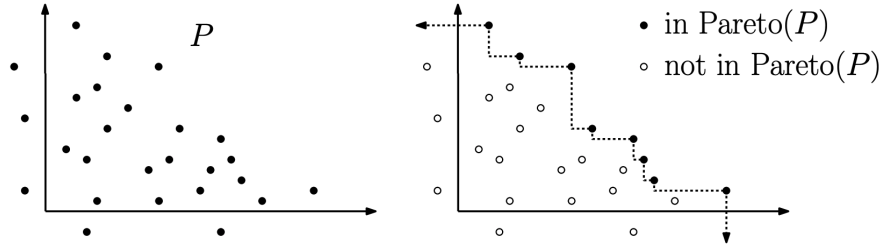


Figure 2: Problem 2: Pareto set

2. (5 points) Devise an analogue of Graham's convex-hull algorithm for computing $\text{Pareto}(P)$ 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 with modifications would be made to Graham's algorithm.)
3. (5 points) Devise an analogue of the Jarvis march algorithm for computing $\text{Pareto}(P)$ in $O(h \cdot n)$ time, where h is the cardinality of $\text{Pareto}(P)$. (As with the previous part, you can just explain the differences with Jarvis's algorithm.)
4. (5 points) Devise an algorithm for computing $\text{Pareto}(P)$ in $O(n \log h)$ time, where h is the cardinality of $\text{Pareto}(P)$.

Problem 3 (10 points)

Assume you have an orientation test available which can determine in constant time whether three points make a left turn (i.e., the third point lies on the left of the oriented line described by the first two points) or a right turn. Now, let a point q and a convex polygon $P = \{p_1, \dots, p_n\}$ in the plane be given, where the points of P are stored in an array in counter-clockwise order around P and q is outside of P . Give pseudo-code to determine the tangents from q to P in $O(\log n)$ time.

Problem 4 (10 points)

Given a set S of n points in the plane, consider the subsets

$$\begin{aligned}
 S_1 &= S, \\
 S_2 &= S_1 \setminus \{\text{set of vertices of } \text{conv}(S_1)\} \\
 &\dots \\
 S_i &= S_{i-1} \setminus \{\text{set of vertices of } \text{conv}(S_{i-1})\}
 \end{aligned}$$

until S_k has at most three elements. Give an $O(n^2)$ time algorithm that computes all convex hull $\text{conv}(S_1), \text{conv}(S_2), \dots, S_k$. [Extra credit, provide an algorithm that is faster than $O(n^2)$].

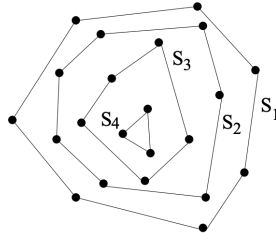


Figure 3: Problem 4: Onion peeling

Tips and Acknowledgements

David Mount’s tips for writing up homework solutions: 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. 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

Acknowledgements: Homework problems adapted from assignments of David Mount and Carola Wenk.