CMSC 754: Spring 2026 Dave Mount

**Sample Problems for the Final Exam**

The final exam will be **Tue, May 12, 10:30am–12:30pm** in class. The exam will be closed-book and closed-notes. You may use *two* sheets of notes (front and back). The following problems have been collected from old homeworks and exams. They do not necessarily reflect the actual difficulty or coverage of questions on the final exam. The final will be comprehensive, but will emphasize material since the midterm.

In all problems, unless otherwise stated, you may assume general position, and you may use of any results presented in class or any well-known result from algorithms and data structures.

**Disclaimer:** The following sample problems have been collected from old homeworks and exams. Because the material and order of coverage varies each semester, these problems *do not* necessarily reflect the actual length, coverage, or difficulty of the midterm exam.

**Problem 1.** Give a short answer (a few sentences) to each question. Unless explicitly requested, explanations are not required, but may be given for partial credit.

(a) Let $P$ be a simple polygon with $n$ sides, where $n$ is a large number. As a function of $n$, what is the maximum number of *scan reflex vertices* that it might have? Draw an example to illustrate.

(b) A convex polygon $P_1$ is enclosed within another convex polygon $P_2$ (see Fig. 1(a)). Suppose you dualize the vertices of each of these polygons (using the dual transform given in class, where the point $(a, b)$ is mapped to the dual line $y = ax - b$). What can be said (if anything) about the relationships between the resulting two sets of dual lines.

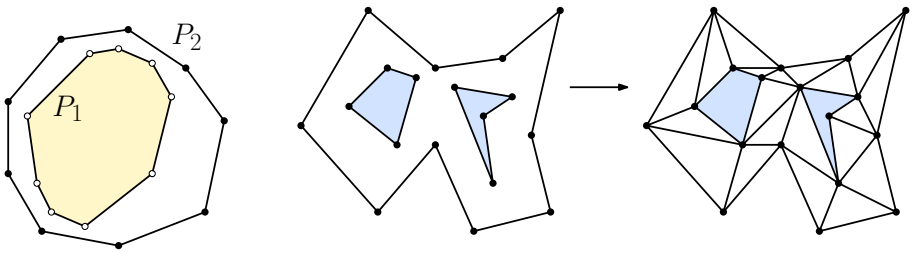


Figure 1: (a) Duals of nested polygons and (b) triangulating a polygon with holes.

(c) You are given a set of $n$ disjoint line segments in the plane that have $m$ intersection points. (For example, in Fig. 2, $n = 4$ and $m = 3$). Suppose that you build a trapezoidal map of the segments, but whenever two segments intersect, there are two bullet paths shot (up and down) from such a point. As a function of $n$ and $m$, what is the maximum number of trapezoids in the final trapezoidal map? Explain briefly. (Give an exact, not asymptotic, answer.)

(d) Consider the linear-programming algorithm given in class for $n$ constraints in dimension 2. In class we showed that the *expected-case* running time of the algorithm is $O(n)$.

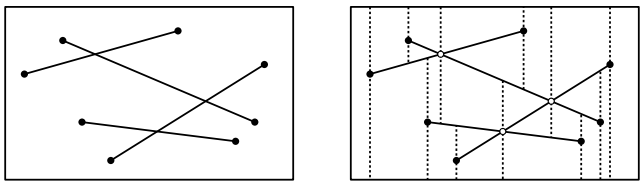

Figure 2: Trapezoidal map with intersections.

What is the *worst-case* running time of the algorithm? Briefly justify your answer (in a sentence or two).

(e) For each of the following assertions about the Delaunay triangulation (DT) of a set $P$ of $n$ points in the plane, which are True and which are False?

(i) The Euclidean minimum spanning tree of $P$ is a subgraph of the DT.

(ii) Among all triangulations of $P$, the DT maximizes the minimum angle.

(iii) Among all triangulations of $P$, the DT minimizes the maximum angle.

(iv) Among all triangulations of $P$, the DT minimizes the total sum of edge lengths.

(v) The DT is a $t$-spanner, for some constant $t$.

(f) A *dodecahedron* is a convex polyhedron that has 12 faces, each of which is a 5-sided pentagon. Every vertex has degree 3. How many vertices and edges does the dodecahedron have? Show how you derived your answer.

(g) Given a set $P$ of $n$ points in the plane, what is the maximum number of edges in $P$'s Voronoi diagram? (For full credit, express your answer up to an additive constant.)

(h) When the $i$th site is added to the Delaunay triangulation using the randomized incremental algorithm, what is the worst-case number of edges that can be incident on the newly added site? What can you say about the expected-case number of such edges (assuming that points are inserted in random order)?

(i) An arrangement of $n$ lines in the plane has exactly $n^2$ edges. How many edges are there in an arrangement of $n$ planes in 3-dimensional space? (Give an exact answer for full credit or an asymptotically tight answer for half credit.) Explain briefly.

(j) Let $P$ and $Q$ be two simple polygons in $\mathbb{R}^2$, where $P$ has $m$ vertices and $Q$ has $n$ vertices. What is the maximum number of vertices on the boundary of the Minkowski sum $P \oplus Q$ (asymptotically) assuming:

(i) $P$ and $Q$ are both convex

(ii) $P$ is convex but $Q$ is arbitrary

(iii) $P$ and $Q$ are both arbitrary

(k) In each of the following cases, what is the asymptotic worst-case complexity (number of vertices) on the boundary of the union of $n$ of the following objects in $\mathbb{R}^2$:

(i) axis-parallel squares

(ii) axis-parallel rectangles (of arbitrary heights and widths)

(iii) rectangles (of arbitrary heights and widths which need not be axis parallel)

(iv) axis-parallel rectangles, where the width to height ratio is either $4 \times 1$ or $1 \times 4$.

**Problem 2.** A simple polygon $P$ is *star-shaped* if there is a point $q$ in the interior of $P$ such that for each point $p$ on the boundary of $P$, the open line segment $\overline{qp}$ lies entirely within the interior of $P$ (see Fig. 3). Suppose that $P$ is given as a counterclockwise sequence of its vertices $\langle v_1, v_2, \ldots, v_n \rangle$. Show that it is possible to determine whether $P$ is star-shaped in $O(n)$ time. (Note: You are *not* given the point $q$.) Prove the correctness of your algorithm.

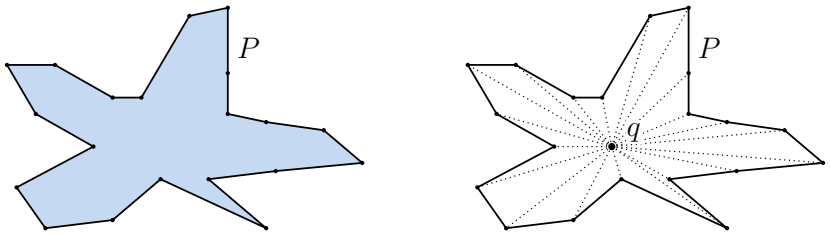


Figure 3: Determining whether a polygon is star-shaped.

**Problem 3.** This problem is inspired from applications in surveillance. Given a simple polygon $P$, we say that two points $p$ and $q$ are *visible* to each other if the open line segment between them lies entirely within $P$'s interior. We allow for $p$ and $q$ to lie on $P$'s boundary, but the segment between them cannot pass through any vertex of $P$ (see Fig. 4(a)).

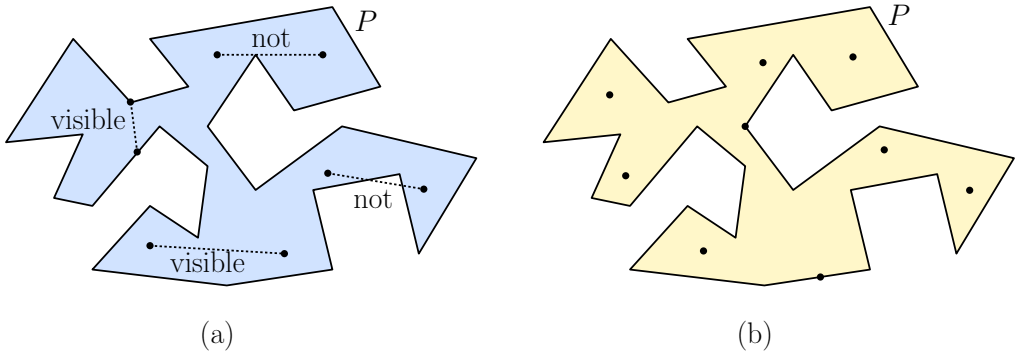


Figure 4: (a) Visibility and (b) a guarding set of size 9 for $P$.

A *guarding set* for $P$ is any set of points $G$, called *guards*, lying in $P$ (either on its boundary or in its interior) such that every point in $P$'s interior is visible to at least one guard of $G$. Note that guards may be placed on vertices, along edges, or in $P$'s interior (see Fig. 4(b)).

Prove that there exists a constant $c \geq 1$ such that (for all sufficiently large $n$) every $n$ vertex simple polygon $P$ has a guarding set of size at most $n/c$. For full credit, show that $c = 3$ works.

**Problem 4.** You are given two sets of points in the plane, the red set $R$ containing $n_r$ points and the blue set $B$ containing $n_b$ points. The total number of points in both sets is $n = n_r + n_b$. Give an $O(n)$ time algorithm to determine whether the convex hull of the red set intersects the convex hull of the blue set. If one hull is nested within the other, then we consider them to intersect. (Hint: It may be easier to consider the question in its inverse form, do the convex hulls *not* intersect.)

**Problem 5.** Consider an $n$-sided simple polygon $P$ in the plane. Let us suppose that the leftmost edge of $P$ is vertical (see Fig. 5(a)). Let $e$ denote this edge. Explain how to construct a data structure to answer the following queries in $O(\log n)$ time with $O(n)$ space. Given a ray $r$ whose origin lies on $e$ and which is directed into the interior of $P$, find the first edge of $P$ that this ray hits. For example, in the figure below the query for ray $r$ should report edge $f$. (Hint: Reduce this to a point location query in an appropriate planar subdivision.)

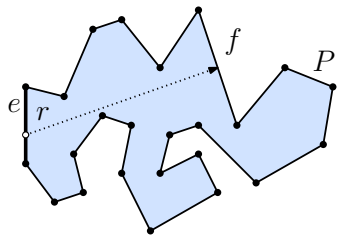


Figure 5: Ray-shooting queries.

**Problem 6.** You are given a set $P$ of $n$ points in $\mathbb{R}^2$. A nonvertical line $\ell$ partitions $P$ into two (possibly empty) subsets: $P^+(\ell)$ consists of the points lie on or above $\ell$ and $P^-(\ell)$ consists of the points of $P$ that lie strictly below $\ell$ (see Fig. 6(a)).

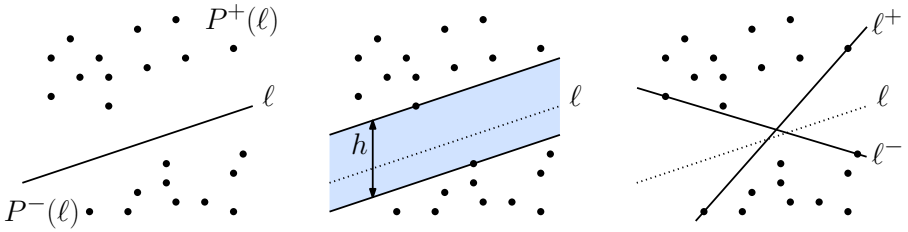


Figure 6: Query problem.

Given the point set $P$, present data structures for answering the following two queries. In each case, the data structure should use $O(n^2)$ space, it should answer queries in $O(\log n)$ time. (You do not need to explain how to build the data structure, but it should be constructable in polynomial time in $n$.)

(a) The input to the query is a nonvertical line $\ell$. The answer is the maximum vertical distance $h$ between two lines parallel to $h$ that lie between $P^+(\ell)$ and $P^-(\ell)$ (see Fig. 6(b)). For simplicity, you may assume that neither set is empty (implying that $h$ is finite).

(b) Again, the input to the query is a nonvertical line $\ell$. The answer to the query are the two lines $\ell^-$ and $\ell^+$ of minimum and maximum slope, respectively, that separate $P^+(\ell)$ from $P^-(\ell)$ (see Fig. 6(c)). You may assume that $P^+(\ell)$ from $P^-(\ell)$ are *not* separable by a vertical line (implying that these two slopes are finite).

**Problem 7.** You are given a set $P$ of $n$ points in the plane and a path $\pi$ that visits each point exactly once. (This path may self-intersect. See Fig. 7.)

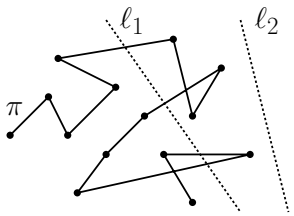


Figure 7: Path crossing queries.

Explain how to build a data structure from $P$ and $\pi$ of space $O(n)$ so that given any query line $\ell$, it is possible to determine in $O(\log n)$ time whether $\ell$ intersects the path. (For example, in Fig. 7 the answer for $\ell_1$ is "yes," and the answer for $\ell_2$ is "no.") (**Hint:** Duality is involved, but the solution requires a bit of lateral thinking.)

**Problem 8.** You are given a set $P$ of $n$ points in $\mathbb{R}^d$ and an approximation factor $\varepsilon > 0$. An (exact) *distance query* is defined as follows. You are given a real $\delta > 0$, and you are to return a count of all the pairs of points $(p, q) \in P \times P$, such that $\|pq\| \geq \delta$. In an *$\varepsilon$-approximate distance query*, your count *must* include all pairs $(p, q)$ where $\|pq\| \geq \delta(1+\varepsilon)$ and it *must not* include any pairs $(p, q)$ where $\|pq\| < \delta/(1+\varepsilon)$. Pairs of points whose distances lie between these two bounds may or may not be counted, at the discretion of the algorithm.

Explain how to preprocess $P$ into a data structure so that $\varepsilon$-approximate distance counting queries can be answered in $O(n/\varepsilon^d)$ time and $O(n/\varepsilon^d)$ space. (Hint: Use a well-separated pair decomposition. Explain clearly what separation factor is used and any needed modification to the WSPD construction.)