Sample Problems for the Midterm Exam

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 midterm exam. (They certainly do not reflect the length of the exam!) Note that some topics we covered this semester were not covered in previous semesters. (In particular, the upper bound theorem and Chan's algorithm are new this semester.)

The exam will be closed-book and closed-notes. You may use one sheet of notes (front and back). 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.

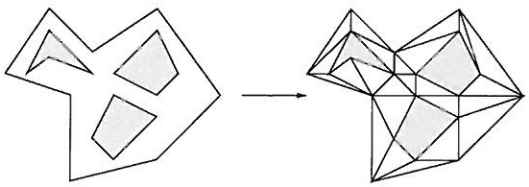
Problem 1. Give a short answer (a few sentences) to each question.

(a) In the analysis of the randomized incremental point location we argued that the expected depth of a random query point is at most $12 \ln n$. Based on your knowledge of the proof, where does the factor 12 arise? (Hint: It arises from two factors. You can explain either for partial credit.)

(b) In the primal plane, there is a triangle whose vertices are the three points p, q, and r and there is a line ℓ that intersects this triangle. What can you infer about the relationship among the corresponding dual lines p^*, q^*, r^*, and the dual point ℓ^*? Explain.

(c) Recall the orientation primitive $Orient(a, b, c)$, which given three points in the plane, returns a positive value if the points are ordered counterclockwise, zero if they are collinear, and negative if clockwise. Show how to use this primitive (one or more times) to determine whether a point d lies within the interior of the triangle defined by the points a, b, and c. (You may assume that a, b, and c are oriented counterclockwise.)

(d) In class we showed that any triangulation of any n-sided simple polygon has exactly $n - 2$ triangles. Suppose that the polygon has h polygonal holes each having k sides. (In the figure below $n = 8$, $h = 3$, and $k = 4$). As a function of n, h and k, how many triangles will such a triangulation have?

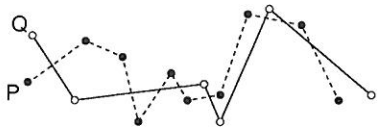


(e) In Fortune's algorithm for planar Voronoi diagrams, given n sites, what is the maximum number of distinct arcs that a *single site* can contribute to the beach line? (You may use O-notation.)

Problem 2.

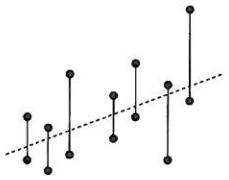
(a) You are given a convex polygon in the plane having n_c sides and an x-monotone polygon having n_m sides. What is the maximum number of intersections that might occur between the edges of these two polygons? (You may use O-notation.)

(b) You are given two x-monotone polygonal chains P and Q with a total of n vertices between then. Prove that P and Q can intersect at most $O(n)$ times.



(c) Does the bound proved in (b) still apply if the two polygonal chains that are monotone with respect to *different* directions (e.g., one monotone with respect to x and the other monotone with respect to y)? Justify your answer.

Problem 3. You are given a set of n vertical line segments in the plane. Present an efficient an algorithm to determine whether there exists a line that intersects all of these segments. An example is shown in the figure below. (Hint: $O(n)$ time is possible.) Justify your algorithm's correctness and derive its running time.

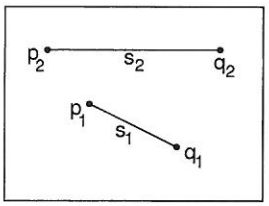


Problem 4. Consider the segments shown in the figure below. Consider the insertion order $\langle s_1, s_2 \rangle$.

(a) Show the trapezoidal map after the insertion of s_1.

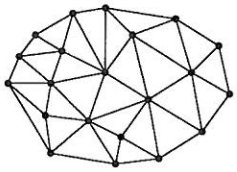
(b) Show the trapezoidal map after the insertion of both segments. (Label the trapezoids for part (c).)

(c) Show the *final* history DAG for this insertion order.



Problem 5. Given a set of n points in the plane, a *triangulation* of these points is a planar straight line graph whose outer face is the convex hull of the point set, and each of whose internal faces is a triangle. There are many possible triangulations of a set of points. Throughout this problem you may assume that no three points are collinear.

(a) Among the n points, suppose that h lie on the convex hull of the point set. As a function of n and h, what is the number of triangles (internal faces) t in the triangulation. Show

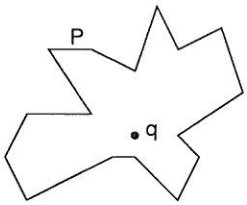


how you derived your answer. (It is a fact, which you do not need to prove, that the number of triangles depends only on n and h.) You may give an asymptotic answer for partial credit.

(b) Describe an $O(n \log n)$ algorithm for constructing *any* triangulation (your choice) of a set of n points in the plane. Explain your algorithm and analyze its running time. You may assume that the result is stored in any reasonable representation. (Hint: There is a simple plane-sweep algorithm.)

Problem 6. 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.

Problem 7. 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 the figure below.) 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 in linear time whether P is star-shaped in $O(n)$ time. (Note: You are *not* given the point q.) Prove the correctness of your algorithm.



Problem 8. Consider the following randomized incremental algorithm, which computes the smallest rectangle (with sides parallel to the axes) bounding a set of points in the plane. This rectangle is represented by its lower-left point Lo and the upper-right point Hi.

(1) Let $P = \{p_1, p_2, \ldots, p_n\}$ be a random permutation of the points.

(2) Let $Lo[x] = Hi[x] = p_1[x]$. Let $Lo[y] = Hi[y] = p_1[y]$.

(3) For $i = 2$ through n do:

(a) if $p_i[x] < Lo[x]$ then $(*)$ $Lo[x] = p_i[x]$.

(b) if $p_i[y] < Lo[y]$ then $(*)$ $Lo[y] = p_i[y]$.

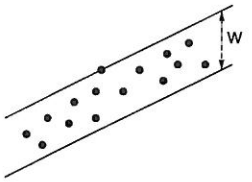
(c) if $p_i[x] > Hi[x]$ then $(*)$ $Hi[x] = p_i[x]$.

(d) if $p_i[y] > Hi[y]$ then $(*)$ $Hi[y] = p_i[y]$.

Clearly this algorithm runs in $O(n)$ time. Prove that the total number of times that "then" clauses of statements 3(a)–(d) (each indicated with a $(*)$) are executed is $O(\log n)$ on average. (We are averaging over all possible random permutations of the points.) To simplify your analysis you may assume that no two points have the same x- or y-coordinates. (Hint: Use backwards analysis.)

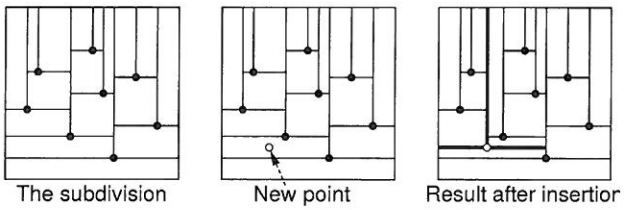
Problem 9. Define a *strip* to be the region bounded by two (nonvertical) parallel lines. The *width* of a strip is the vertical distance between the two lines.

(a) Suppose that a strip of vertical width w contains a set of n points in the primal plane. Dualize the points and the two lines. Describe (in words and pictures) the resulting configuration in the dual plane. Assume the usual dual transformation that maps point (a, b) to the line $y = ax - b$, and vice versa.



(b) Give an $O(n)$ time algorithm, which given a set of n points in the plane, finds the nonvertical strip of minimum width that encloses all of these points.

Problem 10. Given a set of n points P in the plane, we define a subdivision of the plane into rectangular regions by the following rule. We assume that all the points are contained within a bounding rectangle. Imagine that the points are sorted in increasing order of y-coordinate. For each point in this order, shoot a bullet to the left, to the right and up until it hits an existing segment, and then add these three bullet-path segments to the subdivision. (See the figure below left for an example.)



(a) Show that the resulting subdivision has size $O(n)$ (including vertices, edges, and faces).

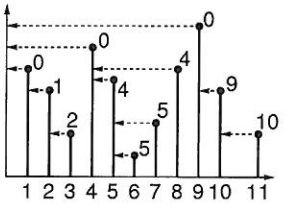
(b) Describe an algorithm to add a new point to the subdivision and restore the proper subdivision structure. Note that the new point may have an arbitrary y-coordinate, but the subdivision must be updated as if the points were inserted in increasing order of y-coordinate. (See the figure above center and right.)

(c) Prove that if the points are added in random order, then the expected number of structural changes to the subdivision with each insertion is $O(1)$.

Problem 11. You are given a collection of vertical line segments in the first quadrant of the x, y plane. Each line segment has one endpoint on the x-axis and the other endpoint has a positive y-coordinate. Imagine that from the top of each segment a horizontal bullet is shot to the left. The problem is to determine the index of the segment that is first hit by each of these bullet paths. If no segment is hit, return the index 0. (See the figure below.)

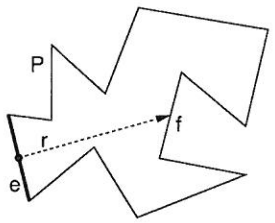
The input is a sequence of top endpoints of each segment $p_i = (x_i, y_i)$, for $1 \leq i \leq n$, which are *sorted in increasing order* by x-coordinate. The output is the sequence of indices, one for each segment.

Present an $O(n)$ time algorithm to solve this problem. Explain how your algorithm works and justify its running time.



Output: 0,1,2,0,4,5,5,4,0,9,1

Problem 12. Given an n-vertex simple polygon P and an edge e of P, show how to construct a data structure to answer the following queries in $O(\log n)$ time and $O(n)$ 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. You may assume that e is rotated into a convenient position, if it helps to simplify things. (Hint: Reduce this to a point location query.)



CMSC 754: Midterm Exam

This exam is closed-book and closed-notes. You may use one sheet of notes, front and back. Write all answers in the exam booklet. If you have a question, either raise your hand or come to the front of class. Total point value is 100 points. Good luck!

In all problems, unless otherwise stated, you may assume that points are in *general position*. You may make use of any results presented in class and any well known facts from algorithms or data structures. If you are asked for an $O(T(n))$ time algorithm, you may give a *randomized* algorithm with *expected* time $O(T(n))$.

Problem 1. (25 points) Give a short answer (a few sentences) to each question.

(a) There is a lower bound of $\Omega(n \log n)$ (from the element-uniqueness problem) on computing the closest pair of points in the plane. Nonetheless, we saw a grid-based algorithm that runs in $O(n)$ time. Why is this not a contradiction?

(b) How many edges (1-faces) does a d-dimensional simplex have? How many edges are incident to a single vertex in a d-dimensional simplex? Explain briefly.

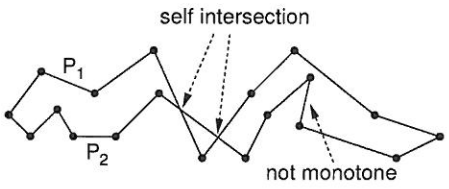
(c) You are given a set of n disjoint line segments in the plane. How many trapezoids are there in a trapezoidal decomposition of these segments? Explain briefly. (Give an exact, not asymptotic, answer.)

(d) Suppose the line segments of part (c) are not disjoint, but have I intersection points. Each intersection point results in two bullet paths. As a function of n and I, how many faces are there now? Explain briefly.

(e) What are the two types of events that arise in Fortune's plane sweep algorithm for Voronoi diagrams. Briefly explain what each event means. (You do not need to explain how each event is processed.)

Problem 2. (10 points)

Let P_1 and P_2 denote two polygonal chains that share the same start and end points (see the figure to the right). They are supposed to form the upper and lower parts of a horizontally monotone polygon, which we are to triangulate. However, we do not trust the integrity of the input. Explain how to modify the monotone polygon triangulation algorithm given in class so that *as it runs* it also tests the following two conditions:



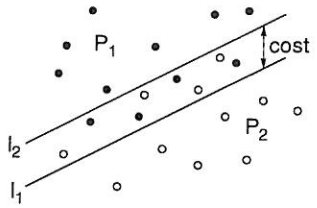
(1) The chains P_1 and P_2 are indeed horizontally monotone.
(2) The polygon is simple, meaning that it does not self intersect.

Your modified algorithm should run in $O(n)$ time and can terminate as soon as either error is detected. Briefly justify the correctness and running time of your algorithm. (It is not necessary to give the entire triangulation algorithm, just explain the modifications.)

Note: You can get *half credit* by presenting any $O(n)$ time algorithm, even if it is not based on the monotone triangulation algorithm given in class.

Problem 3. (20 points)

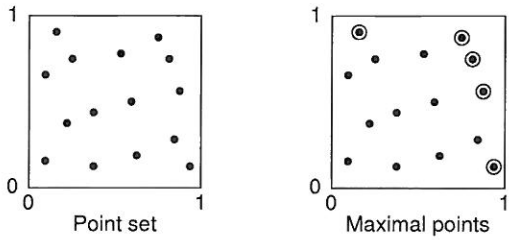
You are given two sets of points in the plane, P_1 and P_2. Let n denote the total number of points in $P_1 \cup P_2$. A *partial classifier* is a pair of parallel lines ℓ_1 and ℓ_2, such that all the points of P_1 lie on or above ℓ_1 and all the points of P_2 lie on or below ℓ_2. The *cost* of the partial classifier is the vertical distance between these lines.



(a) Assuming the standard dual transformation, which maps the point (a, b) to the dual line $y = ax - b$ (and vice versa), give a geometric explanation of what a partial classifier means in the dual plane. What is the meaning of cost in the dual setting?

(b) Give an algorithm which, given point sets P_1 and P_2, computes a partial classifier of minimum cost. $O(n)$ time is possible. Briefly justify the correctness and running time of your algorithm.

Problem 4. (20 points) Given two points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ in the plane, we say that p_2 *dominates* p_1 if $x_1 \leq x_2$ and $y_1 \leq y_2$. Given a set of points $P = \{p_1, p_2, \ldots, p_n\}$, a point p_i is said to be *maximal* if it is not dominated by any other point of P. (See the figure below.)



Suppose further that the points of P have been generated by a random process, where the x-coordinate and y-coordinate of each point are independently generated random real numbers in the interval $[0, 1]$.

(a) Assume that the points of P are sorted in increasing order of their x-coordinates. As a function of n and i, what is the probability that p_i is maximal? (Hint: Consider the points p_j, where $j \geq i$.)

(b) Prove that the expected number of maximal points in P is $O(\log n)$.

Problem 5. (25 points) Recall that given an n element point set in the plane, Chan's convex hull algorithm iterates over values $t = 1, 2, \ldots$. For each value of t, it partitions the points into subsets of size $m \leftarrow \min(2^{2^t}, n)$. For each partition it invokes a procedure PartialHull(P, m), which runs in $O(n \log m)$ time, but fails if $m < h$, where h is the number of points on the final convex hull. In summary:

- For $t \leftarrow 1, 2, 3, \ldots$ do:
  - Let $m \leftarrow \min(2^{2^t}, n)$.
  - Let $L \leftarrow$ PartialHull(P, m).
  - If $L \neq$ "Fail" then return L.

In each of the following cases, indicate what the running time of Chan's algorithm would be had we used a different progression for m. (Use O-notation as a function of n and h.) Briefly explain each answer. For full credit, attempt to express each running time in simplest terms.

(a) What would the running time be if we chose $m \leftarrow \min(t, n)$?

(b) What would the running time be if we chose $m \leftarrow \min(2^t, n)$?

(c) What would the running time be if we chose $m \leftarrow \min(2^{2^{2^t}}, n)$?

Hint: The following identities may be useful. $\sum_{i=1}^{N} \log i = \Theta(N \log N)$, $\sum_{i=1}^{N} i^c = \Theta(N^{c+1})$, $\sum_{i=1}^{N} f(i) = \Theta(f(N))$, where $f(i)$ is any function that grows at an exponential rate or higher.

Sample Problems for the Final Exam

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. Note that some topics we covered this semester (upper bound theorem, Chan's convex hull algorithm, VC-dimension) were not covered in previous semesters. The final will be comprehensive, but will emphasize material since the midterm.

The exam will be closed-book and closed-notes. You may use two sheets of notes (front and back). 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.

Problem 1. Give a short answer (a few sentences) to each question.

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

(b) A kd-tree of n points in the plane defines a subdivision of the plane into n cells, each of which is a rectangle. Is this subdivision a *cell complex*? Explain briefly.

(c) Given a k-d tree with n points in the plane, what is the (asymptotic) maximum number of cells that might be stabbed by an line that is not axis-parallel? Explain briefly.

(d) What is a *zone* in an arrangement? Given an n-line arrangement A in the plane and given an arbitrary line ℓ, what is the (asymptotic) maximum complexity (number of edges) of the zone of ℓ relative to A? *(No explanation needed.)*

(e) Which of the following statements regarding the Delaunay triangulation (DT) of a set of points in the plane are true? *(No explanation needed.)* Among all triangulations...

(i) ...the DT minimizes the maximum angle.

(ii) ...the DT maximizes the minimum angle.

(iii) ...the DT has the minimum total edge length.

(iv) The largest angle in a DT cannot exceed 90 degrees.

(f) 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 partial credit.) Explain briefly.

Problem 2. In class we argued that the number of parabolic arcs along the beach line in Fortune's algorithm is at most $2n - 1$. The goal of this problem is to prove this result in a somewhat more general setting.

Consider the beach line at some stage of the computation, and let $\{p_1, p_2, \ldots, p_n\}$ denote the sites that have been processed up to this point in time. Label each arc of the beach line with its the associated site. Reading the labels from left to right defines a string. (In Fig. 1 below the string would be $p_1p_2p_1p_3p_6p_4p_6p_7p_9$.)

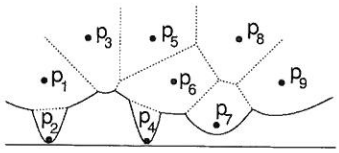


Figure 1: Problem 2.

(a) Prove that for any i, j, the following alternating subsequence cannot appear anywhere within such a string:

$$\ldots p_i \ldots p_j \ldots p_i \ldots p_j \ldots$$

(b) Prove that any string of n distinct symbols that does not contain any repeated symbols ($\ldots p_i p_i \ldots$) and does not contain the alternating sequence of the type given in part (a) cannot be of length greater than $2n-1$. (Hint: Use induction on n.) These are important sequences in combinatorics, known as *Davenport-Schinzel sequences.*

Problem 3. Consider a set P of n points in the plane. For $k \leq \lfloor n/2 \rfloor$, point q (which may or may not be in P) is called a *k-splitter* if every line L passing through q has at least k points of P lying on or above it and at least k points on or below it. (For example the point q in Fig. 2 is a 3-splitter, since every line passing through q has at least 3 points of P lying on either side. But it is not a 4-splitter since a horizontal line through q has only 3 points below it.)

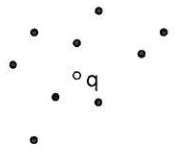


Figure 2: Problem 3.

(a) Show that for all (sufficiently large) n there exists a set of n points with no $\lfloor n/2 \rfloor$-splitter.

(b) Prove that there exists a k-splitter if and only if in the dual line arrangement, levels $\mathcal{L}_k$ and $\mathcal{L}_{n-k+1}$ can be separated by a line.

(c) Prove that any set of n points in the plane has a $\lfloor n/3 \rfloor$-splitter.

(d) Describe an $O(n^2)$ algorithm for computing a $\lfloor n/3 \rfloor$-splitter for point set P.

Problem 4. Given a set P of n points in the plane, and given any slope θ, project the points orthogonally onto a line whose slope is θ. The order (say from top to bottom) of the projections is called an *allowable permutation.* (If two points project to the same location, then order them arbitrarily.) For example, in Fig. 3 for the slope θ the permutation would be $\langle p_1, p_3, p_2, p_5, p_4, p_6 \rangle$.

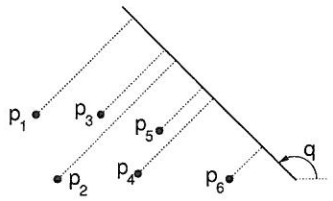


Figure 3: Problem 4.

(a) Prove that there are $O(n^2)$ distinct allowable permutations. (Hint: What does an allowable permutation correspond to in the dual plane?)

(b) Give an $O(n^3)$ algorithm which lists all the allowable permutations for a given n-point set. ($O(n^2)$ time to find the permutations and $O(n)$ time to list each one.)

Problem 5. You are given a set of n triangles in the plane $\mathcal{T} = \{T_1, \ldots, T_n\}$, where triangle T_i has vertices $\langle a_i, b_i, c_i \rangle$. Present an algorithm that computes a line ℓ that stabs the greatest number of triangles of $\mathcal{T}$. (You may assume general position.) For example, in Fig. 4 there exists a line that intersects 4 of the 5 triangles. Your algorithms should run in $O(n^2)$ time and use at most $O(n^2)$ space.

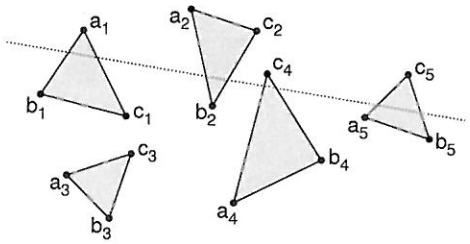


Figure 4: Problem 5.

Problem 6. This problem involves a range space (P, R) where P is a set of n points in the plane, and R is the set of all ranges arising by intersecting P with a closed halfplane.

(a) Show that the VC-dimension of halfplane ranges is at least 3 by giving an example of a set Q of 3 points in the plane that are shattered by the set of halfplane ranges.

(b) Show that the VC-dimension of halfplane ranges is at most 3, by proving that no 4-element set can be shattered by halfplane ranges.

(c) Present a data structure for halfplane range counting queries in the plane that uses $O(n^2)$ space and can answer queries in $O(\log n)$ time. Recall that the objective is to preprocess a set P of n points in the plane, so that given any query halfplane h, we can return a count of the number of points of $P \cap h$ in $O(\log n)$ time.

Problem 7. Given a set of n points P in $\mathbb{R}^d$, and given any point $p \in P$, its *nearest neighbor* is the closest point to p among the remaining points of P. Given an approximation factor $\varepsilon > 0$, we say that a point $p' \in P$ is an *ε-approximate nearest neighbor* to p if $\|pp'\| \leq (1+\varepsilon)\|pp''\|$, where p'' is the true nearest neighbor to p. Show that in $O(n \log n + (1/\varepsilon)^d n)$ time it is possible to compute an ε-approximate nearest neighbor for every point of P. Justify the correctness of your algorithm.

CMSC 754: Final Exam

This exam is closed-book and closed-notes. You may use two sheets of notes, front and back. Write all answers in the exam booklet. If you have a question, either raise your hand or come to the front of class. Total point value is 100 points. Good luck!

In all problems, unless otherwise stated, you may assume that points are in *general position*. You may make use of any results presented in class and any well known facts from algorithms or data structures. If you are asked for an $O(T(n))$ time algorithm, you may give a *randomized* algorithm with *expected* time $O(T(n))$.

Problem 1. (20 points; 3-6 points each) Give a short answer (a few sentences) to each question.

(a) Is Jarvis's March ever faster than Graham's scan (in terms of its asymptotic running time)? If so, indicate under what circumstances.

(b) What are the three possible outcomes of any linear programming instance? Explain each briefly.

(c) When the ith segment is added to a trapezoidal map...

(i) ...what is the worst case number of structural changes to the map?

(ii) ...assuming a random insertion order, what is the expected number of structural changes?

No explanation required. Asymptotic answers are fine.

(d) Assuming d is a constant:

(i) What is the worst-case complexity of the Delaunay triangulation of n points in d-space?

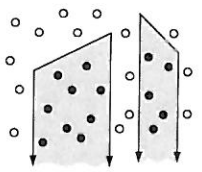
(ii) What is the number of vertices in an arrangement of n hyperplanes in d-space?

(No explanation needed.)

(e) Let P be a set of n points in the plane, where each point is associated with a numerical weight. Given any axis-aligned rectangle the query problem is to return the minimum weight among the points in this rectangle. **True or false:** It is possible to answer such queries using a kd-tree in $O(\sqrt{n})$ time. (Give a very brief explanation.)

Problem 2. (15 points)

Define a *semi-trapezoid* to be the closed region bounded on the left and right by vertical lines and above by a nonvertical line. (Two examples are shown in the figure to the right. Note that the range extends to $-\infty$ in the y-direction.) What is the VC-dimension of the range space of closed semi-trapezoids in the plane? Provide a brief proof.



Problem 3. (15 points) Consider a set P of n points in the plane. The distances between each pair of distinct points defines a set of $\binom{n}{2}$ *interpoint distances*. Present an efficient algorithm to compute an approximation the *second largest* interpoint distance.

More formally, your algorithm is given a set P of n points in the plane and a constant approximation parameter $\varepsilon > 0$. Let Δ denote the true second largest interpoint distance among the points of P. Your algorithm may output any value Δ' where

$$\frac{\Delta}{1+\varepsilon} \leq \Delta' \leq (1+\varepsilon)\Delta.$$

(Hint: Use WSPDs. First give the algorithm for an arbitrary separation parameter s, then derive a value of s that provides the approximation bound.)

For full credit, you should *derive* the value of s, and not just restate a bound that we proved in class.

Problem 4. (15 points) You are given a set of n sites P in the plane. Each site of P is the center of a circular disk of radius 1. The points within each disk are said to be *safe*. We say that P is *safely connected* if, given any $p, q \in P$, it is possible to travel from p to q by a path that travels only in the safe region. (For example, the disks of Fig. 1(a) are connected, and the disks of Fig. 1(b) are not.)

Present an $O(n \log n)$ time algorithm to determine whether such a set of sites P is safely connected. Justify the correctness of your algorithm and derive its running time.

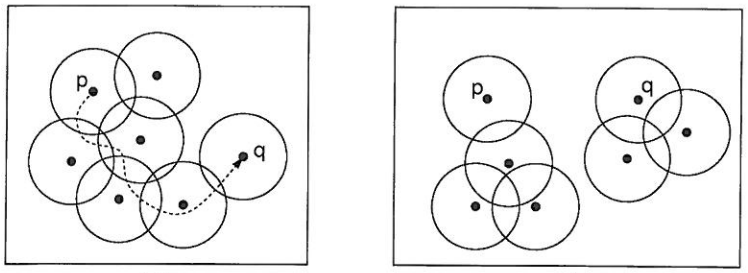


Figure 1: Problem 4.

Problem 5. (15 points) You are given a set $P = \{p_1, p_2, \ldots, p_n\}$ of n points in the plane. Consider all the $\binom{n}{2}$ lines passing through each pair of distinct points $p_i, p_j \in P$, and let $\ell_{\max}$ to be the line of this set with the maximum slope. We are interested in computing $\ell_{\max}$.

(a) What is the interpretation of $\ell_{\max}$ in the dual plane?

(b) Give an $O(n \log n)$ algorithm that computes $\ell_{\max}$. Justify your algorithm's correctness.

Problem 6. (20 points) You are given a set P of n points in the plane and a path π that visits each point exactly once. (This path may self-intersect.) (See Fig. 2(a) and (b).)

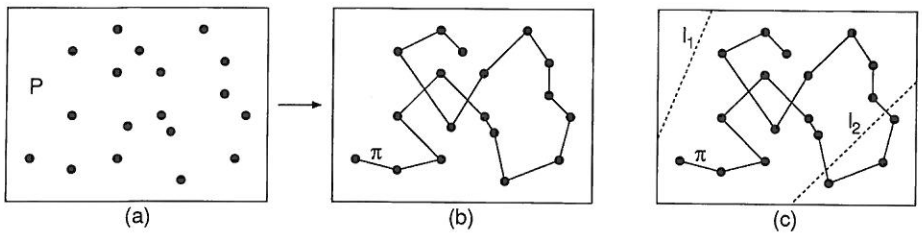


Figure 2: Problem 5.

(a) (10 points) Explain how to build a data structure from P and π of space $O(n)$ so that given any query line ℓ, it is possible to determine in $O(\log n)$ time whether ℓ intersects the path. (For example, in Fig. 2(c) the answer for ℓ_1 is "no," and the answer for ℓ_2 is "yes.")

(b) (10 points) This is a generalization to part (a). Explain how to build a data structure from P and π so that given any line ℓ, it is possible to report all the segments of π that intersect ℓ. The space should be at most $O(n \log n)$ and the query time should be at most $O(k \log^2 n)$, where k is the number of segments reported. (Hint: Even if you did not solve (a), you can solve this by applying the data structure of part (a) as a "black box.")