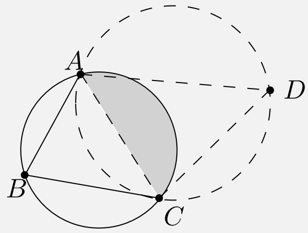
Problem 1. Let D be a point outside $\odot ABC$ (circumcircle of triangle ABC) such that points B and D fall on different sides of the line passing through segment $\overline{AC}$. Then, $\odot ACD$ covers territory of arc $\overset{\frown}{AC}$ inside $\odot ABC$ (the shadow region in the figure below).



Hint: Pick any point E on the arc $\overset{\frown}{AC}$ ($E \neq A$ and $E \neq C$). Prove that the angle $\angle AEC > \angle ADC$.

Proof. Pick any point E on the arc $\overset{\frown}{AC}$.

Consider the circumcircle $\odot ABC$. Since D is a point outside $\odot ABC$, $\angle AEC > \angle ADC$.

Now consider the circumcircle $\odot ACD$. Now, D is a point on $\odot ACD$. Since $\angle AEC > \angle ADC$, E is inside the circle $\odot ACD$.

Thus, the entire arc $\overset{\frown}{AC}$ is covered by $\odot ACD$. □

Problem 2. Given a triangle ABC and a point p, determine in $O(1)$ time if $\odot ABC$ covers p.

Solution. Let $A = (x_1, y_1), B = (x_2, y_2), C = (x_3, y_3)$. Let $q = (x, y)$ be a point on or in the interior of the circumcircle of $\triangle ABC$.

Consider the set of equations:

$$\begin{cases} a(x^2+y^2) + bx + cy + d \leq 0 \\ a(x_1^2+y_1^2) + bx_1 + cy_1 + d = 0 \\ a(x_2^2+y_2^2) + bx_2 + cy_2 + d = 0 \\ a(x_3^2+y_3^2) + bx_3 + cy_3 + d = 0 \end{cases}$$

To obtain a valid non-trivial solution (a, b, c, d),

$$\begin{vmatrix} x^2+y^2 & x & y & 1 \\ x_1^2+y_1^2 & x_1 & y_1 & 1 \\ x_2^2+y_2^2 & x_2 & y_2 & 1 \\ x_3^2+y_3^2 & x_3 & y_3 & 1 \end{vmatrix} \leq 0.$$

This determinant can be computed in $O(1)$ time.

Problem 3. Let P be a set of n points in $\mathbb{R}^2$. A Euclidean spanning tree of P is a tree where every vertex is a point in P and every edge is a line segment connecting two points of P. The tree's weight equals the total (Euclidean) length of all the segments. The *Euclidean minimum spanning tree* (EMST) is a Euclidean spanning tree of the smallest weight. Give an algorithm to find an EMST of P in $O(n \log n)$ expected time.
Remark: The expected $O(n \log n)$ time is due to the randomized Delaunay triangulation algorithm. Deterministic $O(n \log n)$ time algorithm exists.

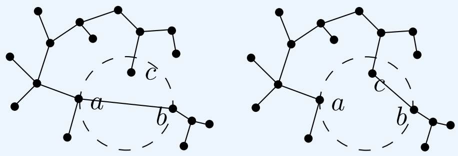
Solution. First, we compute the point set Delaunay triangulation in expected $O(n \log n)$ time.

Then compute the MST of the triangulation by Kruskal's algorithm and return the result.

Note that there are at most $3n-6$ edges in a point set triangulation. Therefore, Kruskal's algorithm can be done in $O(n \log n)$ time.

We now prove that the MST of P is a subgraph of the Delaunay triangulation.

Let T be the MST for P, let $w(T)$ denote the total weight of T. Let a and b be any two sites such that ab is an edge of T. Suppose to the contrary that ab is not an edge in the Delaunay triangulation. This implies that there is no empty circle passing through a and b, and in particular, the circle whose diameter is the segment $\overline{ab}$ contains a site, call it c.



The removal of $\overline{ab}$ from the MST splits the tree into two subtrees. Assume without loss of generality that c lies in the same subtree as a. Now remove the edge $\overline{ab}$ from the MST and add the edge $\overline{bc}$ in its place. This will incur in a spanning tree T' whose weight is

$$w(T') = w(T) + |bc| - |ab| < w(T).$$

Since ab is the diameter of the circle, $|bc| < |ab|$. This contradicts the hypothesis that T is the MST, completing the proof.

Problem 4. Let P be a set of n points in $\mathbb{R}^2$. A tour is a sequence of n segments $\overline{p_1p_2}, \overline{p_2p_3}, \ldots, \overline{p_{n-1}p_n}, \overline{p_np_1}$, where each $p_i (i \in [1, n])$ is a distinct point in P. The *length* of the tour is the total length of all the n segments. Let ℓ^* be the shortest length of all possible tours. Design an algorithm to find a tour with length at most $2\ell^*$ in $O(n \log n)$ expected time.

Solution. In $O(n \log n)$ expected time, one can obtain the MST T of P. Then, a closed walk π where all edges on T appear twice can be computed by pre-order traversal in $O(n)$ time. Then, a Hamiltonian cycle σ can be found by scanning $\pi = (\pi_1, \pi_2, \ldots, \pi_{2n-1})$. Namely, add π_i to σ if π_i has not been visited before, also add π_1 again to the end of the sequence.

Lemma 1: $\ell^* \geq \mathrm{MST}(T)$. Consider removing an edge e from the cycle attaining the cost ℓ^*, we obtained a spanning tree of P. By no means that $\ell^* - \mathrm{cost}(e) < \mathrm{MST}(T)$ and thus $\ell^* \geq \mathrm{MST}(T)$.

Lemma 2: $\left(\sum_{i=1}^{n-1} \mathrm{dist}(\sigma_i, \sigma_{i+1})\right) + \mathrm{dist}(\sigma_n, \sigma_1) \leq 2 \cdot \mathrm{MST}(T)$.

Let $\sigma_1 = \pi_{j_1}, \sigma_2 = \pi_{j_2}, \ldots, \sigma_n = \pi_{j_n}$. It is apparent that $1 = j_1 < j_2 < \ldots < j_n = 2n-1$.

Therefore by the triangle inequality,

$$dist(\sigma_i, \sigma_{i+1}) \leq \sum_{k=j_i}^{j_{i+1}-1} dist(\pi_k, \pi_{k+1}),$$

meaning that the length of cycle $\sigma \leq$ length of cycle $\pi = 2 \cdot MST(T) \leq 2\ell^*$.

The sequence σ is what we want to find.

Problem 5. Let P be a set of n points in $\mathbb{R}^2$. A k-clustering of P is a partition $P_1, P_2, \ldots, P_k$ of P such that

- each $P_i (i \in [1, k])$ is a non-empty subset of P,
- $P_i \cap P_j = \emptyset$ for any different $i, j \in [1, k]$,
- $P_1 \cup P_2 \cup P_3 \cup \ldots \cup P_k = P$.

We will refer to each $P_i (i \in [1, k])$ as a cluster. For any different $i, j \in [i, k]$, define the distance between clusters P_i and P_j as

$$dist(P_i, P_j) = \min_{p \in P_i, q \in P_j} dist(p, q)$$

where $dist(p, q)$ is the Euclidean distance between p and q. The quality of the k-clustering is defined to be the smallest distance between all $\binom{k}{2}$ cluster pairs.

- Prove: If the quality of P is determined by two points $p, q \in P$, then $\overline{pq}$ is an edge in the Delaunay triangulation of P.
- Given a real value $r > 0$ and an integer $k \geq 2$, give an algorithm to determine whether there exists a k-clustering of P whose quality is at least r. Your algorithm needs to finish in $O(n \log n)$ expected time.

Hint: The first question is easy. For the second question, what happens if we remove all the edges of the Delaunay graph of P that have lengths less than r?

Solution.

- Suppose the otherwise that $\overline{pq}$ is not an edge in the Delaunay triangulation of P. There must exist a point r lying inside the circle having $\overline{pq}$ as the diameter. Let o be the center of this circle. Then, without loss of generality, suppose that r does not belong to the cluster with p, $|op| + |or| < 2R = |op| + |oq| = |pq|$. This is a contradiction as the cross-pair distance of the given clustering is at least $|pq|$.
- We first spend $O(n \log n)$ time to compute the Delaunay triangulation of P. Then, we can sort the list of edges L in ascending order in the triangulation in $O(n \log n)$ time. Start by having every point as disjoint n clusters. Now, consider each edge $(u, v) \in L$, we merge the two clusters with u and v only if they do not belong to the same cluster. We terminate the process,
  - if we have to merge more than $k - 1$ times, and report "no";
  - otherwise proceed until we have merged $k - 1$ times, and report "yes".

  It is easy to see the algorithm's correctness as

  1. If we have reported "yes", no quality less than r is possible as each point pair connected by such edges are in the same cluster now.
  2. If we have reported "no", there must exist an edge (u, v) with $dist(u, v) < r$ which fails to be merged. It then means that u and v is a cross-cluster pair, the quality must count this pair in, and is upper-bounded by the distance of this pair.