CS 498TC, Spring 2018
Timothy Chan

Assignment 3 (due March 30 Friday 2pm (in class))

You may work in a group of at most 3 students. Carefully read `https://courses.engr.illinois.edu/cs498tc3/policies.html` and `https://courses.engr.illinois.edu/cs498tc3/integrity.html`.

1. [*7 pts*] Given a convex polygon P, we want to find the largest circle contained in P. Show that this problem can be solved in $O(n)$ time. (Hint: Use linear programming.)

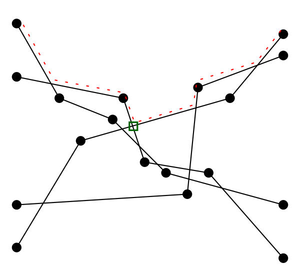
2. [*25 pts*] An *increasing* polygonal curve of size k is a (not necessarily convex) polygonal curve $p_0p_1p_2\cdots p_k$ such that $0 = p_0.x < p_1.x < p_2.x < \cdots < p_k.x = 1$ and $p_0.y < p_1.y < p_2.y < \cdots < p_k.y$. A *decreasing* polygonal curve is a polygonal curve $q_0q_1q_2\cdots q_k$ such that $0 = q_0.x < q_1.x < q_2.x < \cdots < q_k.x = 1$ and $q_0.y > q_1.y > q_2.y > \cdots > q_k.y$. Given n increasing polygonal curves and decreasing polygonal curves, each of size k, we want to find the lowest point on their upper envelope.

   [See the figure below; the upper envelope is shown in red dotted lines; its lowest point is shown in green. Note that two increasing curves may intersect a large number of times, but an increasing and a decreasing curve may intersect at most once. Recall that the intersection between an increasing and a decreasing curve can be computed in logarithmic time by Assignment 1 Question 1.]

   (a) [*10 pts*] Give a randomized algorithm with expected running time $O(n \log k)$, by modifying Seidel's LP algorithm.

   (b) [*15 pts*] Give a deterministic algorithm with worst-case running time $O(n \log^2 k \log n)$ or better, by modifying Megiddo/Dyer's LP algorithm.

   [Hint: consider the middle x-value of each curve, and take the median x_m of these middle x-values. How can we decide whether the solution is to the left or right of x_m? How many iterations are needed to reduce the size of *every* curve by a half?]



3. [*13 pts*] The L_1-*distance* (or "Manhattan distance") between two points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ is defined as $d_1(p_1, p_2) = |x_1 - x_2| + |y_1 - y_2|$. The L_1-distance between two objects A and B is defined as $d_1(A, B) = \min_{a \in A, b \in B} d_1(a, b)$.

(a) [*3 pts*] For a triangle t and a number r, what does the region $\{q \in \mathbb{R}^2 : d_1(q,t) \leq r\}$ look like?

(b) [*10 pts*] Given a set T of n disjoint triangles in 2D and a number r, consider the problem of checking whether there exists a pair of triangles in T with L_1-distance at most r. Show that this problem can be solved in $O(n \log n)$ time by directly using an algorithm from class.

[Bonus (3 pts): consider the more challenging problem of finding the minimum L_1-distance between all pairs of triangles in T. Give a (deterministic or randomized) algorithm with running time close to $O(n \log n)$.]