Spring 1996 **CMSC 754: Homework 5** Samir Khuller

Due in class: Apr 24.

(1) Given n disjoint line segments in the plane, we wish to answer queries of the following form: for a query point p, determine the first segment hit by a bullet that is fired to the right along a horizontal ray. Give an algorithm with $O(n \log n)$ preprocessing and $O(n)$ space that allows the queries to be answered in $O(\log n)$ time.

(2) The L_1 distance (also called manhattan metric) between two points $p = (x_p, y_p)$ and $q = (x_q, y_q)$ is defined by $d_1(p,q) = |x_p - x_q| + |y_p - y_q|$. Characterize the voronoi diagram according to the L_1 metric of n points in the plane (in other words, describe the shape of the bisectors). Do the same for the L_∞ metric, defined by $d_\infty(p,q) = \max(|x_p - x_q|, |y_p - y_q|)$.

(3) The objective of this problem is to devise an algorithm for for finding the maximum perimeter triangle with its three vertices selected from a point set S of n points in the plane.

(a) Consider a collection of m points that are the vertices of a convex polygon, and let a be one such point. Show that the diameter of the point set always intersects the maximum length segment joining a to another point (the maximum length chord rooted at a). Is this intersection property true if we consider two maximal chords rooted at two different points a and b?

(b) Use this observation to develop an $O(m \log m)$ algorithm for finding the diameter of m points that form a convex polygon.

(c) Show that the vertices of a max perimeter triangle with vertices selected from S, must have its vertices on the convex hull.

(d) Formulate a prove a result analogous to part (a), relating the max perimeter triangle and a max perimeter triangle rooted at a particular vertex a. Use this to develop an efficient algorithm for finding the max perimeter triangle with vertices from S.

(4) Problem (2) Exercises 6.6.3 (page 222).

(5) Given n points in the plane, and a parameter d, we wish to construct a graph $G = (V, E)$ where the vertices correspond to the points, and there is an edge between a pair of points if the Euclidean distance between the points is at most d. Clearly, such a graph can be constructed in $O(n^2)$ time. Often this graph may actually be quite sparse (number of edges is $o(n^2)$). Design an algorithm whose running time is a function of the number of edges in the graph (in other words, the algorithm should run faster when the output has fewer edges). How well can you do? (These graphs are also called Unit Disk Graphs.)