Problem Set 7

This problem set is due **AT THE BEGINNING OF** class on *May 9, 2002.*

Both exercises and problems should be solved, but *only the problems* should be turned in. Exercises are intended to help you master the course material and will be useful in solving the assessed problems. You are responsible for material covered by the exercises. Each problem is to be done on a separate sheet (or sheets) of paper. Mark the top of each sheet with your name, 6.046J/18.410J, the problem number, your TA and recitation section, the date, and the names of any students with whom you collaborated.

Problem 7-1. Finding a Close Pair

In the lecture, you have seen a randomized algorithm that given a set of n points and $r > 0$, checks if there is any pair of distinct points p_i, p_j which are "close" to each other, i.e., such that the distance between p_i and p_j is at most r.

Modify the algorithm so that it also outputs a pair p_i, p_j of close points (if it exists), in addition to returning YES or NO. The algorithm should still run in $O(n)$ time, and return the correct answer with probability $\geq 1/2$.

Problem 7-2. TA Location Problem

It has been recently discovered (journal publication to appear soon) that during 6.046 office hours, the students usually occupy random places in the theory lounge of the LCS building. In order to maximize the interaction, the TA must find a place in the lounge such that the total distance from the TA to the students is minimized.

(a) Assume that the students are sitting in a row, i.e., that the student positions are defined by n points $p_1 \ldots p_n$ on a real line $\Re$. Give an algorithm that finds a TA position $q \in \Re$ which minimizes the value of

$$\sum_{i=1}^{n} |p_i - q|$$

Your algorithm should run in $O(n)$ time.

(b) Now assume that the student positions are defined by n points $p_1 \ldots p_n$ in a *plane*[1] $\Re^2$. Give an algorithm that finds a TA position $q \in \Re^2$ which minimizes the value of

$$\sum_{i=1}^{n} |p_i^x - q^x| + |p_i^y - q^y|$$

[1]Note that this does not capture the full complexity of the problem - some students were recently reported to sit on the ceiling, so a 3-dimensional representation is needed in general.

where p^x (and p^y, resp.) is the x-coordinate (y-coordinate resp.) of the point p.

Again, your algorithm should run in $O(n)$ time.

Hint: Think median.

Problem 7-3. Orthogonal intersection counting

In the lecture you have seen an algorithm, which given n horizontal/vertical segments in the plane, reports all intersections among those segments in time $O(n \log n + P)$, where P is the number of intersections.

Show that by modifying that algorithm, one can obtain a procedure which reports the *number* of intersections P in time $O(n \log n)$.