Problem Set 12

Due: Monday, November 24, 2003.

NONCOLLABORATIVE Problem 1. In many applications, one wants to do range searching among objects other than points. In this problem, we will see that we can reduce several problems of this flavor to normal orthogonal range searching.

(a) Let S be a set of n axis-parallel rectangles in the plane. We want to be able to report all rectangles in S that are completely contained in a query rectangle $[x : x'] \times [y : y']$. Describe a data structure for this problem that uses $O(n \log^3 n)$ storage and has $O(\log^4 n + k)$ query time, where k is the number of reported answers. (Hint: Transform the problem to an orthogonal range searching problem in some higher-dimensional space.)

(b) Let P consist of a set of n polygons in the plane. Again, describe a data structure that uses $O(n \log^3 n)$ storage and has $O(\log^4 n + k)$ query time to report all polygons completely contained in the query rectangle, where k is the number of reported answers.

Problem 2. Consider the *MTF-every-other* online linear-search algorithm which moves an accessed item on every even request for that item (including the first). Prove that MTF-every-other is $O(1)$-competitive.

Problem 3. Consider the problem of finding a lifetime companion. Among the pool of k potential partners at the university (MIT students being snobby about this) you can choose and date any one for a while and by doing so measure their suitability compared to all your previous significant others'. You then decide whether to stay together forever or break up forever. Your goal is to find the most suitable companion, as measured by a rank ordering of all the possibilities.

(a) Show that any deterministic strategy for choosing dates and deciding whether to break up is terrible from a competitive perspective: you can be forced by fate to end up with the absolutely worst possible choice.

(b) Devise a randomized strategy that does better, giving you a constant probability of ending up with the absolute best companion.

(c) Suppose that you want to play for slightly less high stakes. Give an algorithm minimizing the *expected rank* of your final choice. Note that a constant is achievable, but (say) $O(\log k)$ is easier to achieve and is worth submitting if you can't achieve a constant.

(d) Comment on potential ramifications for dating at MIT.

OPTIONAL Problem 4. A paging strategy incurs *Belady's anomaly* if, for some sequence of page accesses and two cache sizes $k < k'$, the strategy makes fewer page faults on the smaller cache size k compared to the larger cache size k'. Prove that LRU (Least Recently Used) never incurs Belady's anomaly but that FIFO (First In First Out) incurs Belady's anomaly.