15-859(D) Randomized Algorithms

Homework # 6 **due: December 3, 1998**

You're driving in heavy traffic on some highway, and your lane is going slower than all the rest. At some point you decide to put in the effort to change lanes, and soon after you do it, your original lane speeds up again. What's a good lane-switching strategy? In this exercise we will look at this problem from the point of view of on-line competitive analysis.

We will model this problem as a 2-state "Metrical Task System". There are two states LEFT and RIGHT corresponding to the two lanes of traffic. At any point in time, the online algorithm is in one of these states. The algorithm may at any time move to the other state, but must pay a cost of d for doing so. (d is supposed to model the "effort" for changing lanes). At each time step, the algorithm receives a cost vector (c_l, c_r) (modeling how slow each lane is) where $c_l, c_r \geq 0$. If the algorithm is currently in the LEFT state, it may stay there, paying a cost of c_l, or it may move to the RIGHT state, paying a cost of $d + c_r$. Similarly, if the algorithm is currently in the RIGHT state, it may either stay, paying c_r, or move, paying $d + c_l$.

At any point in time, let's define OPT_L to be the optimal cost in hindsight (the optimal offline cost) of servicing the cost vectors received so far and ending in the state LEFT. Define OPT_R to be the optimal cost in hindsight of servicing the cost vectors received so far and ending in the state RIGHT.

Exercises:

1. Show that OPT_L and OPT_R can never differ from each other by more than d. Also, describe an algorithm for computing OPT_L and OPT_R from a sequence of cost vectors.

2. Suppose you have a randomized algorithm whose probability distribution (p_l, p_r) is a function *only* of OPT_L and OPT_R. (p_l is the probability the algorithm is currently at LEFT; $p_r = 1 - p_l$ is the probability the algorithm is currently at RIGHT.) Prove that in order for the algorithm to have a finite competitive ratio, it must be the case that if $\mathrm{OPT}_L - \mathrm{OPT}_R = d$ then $p_l = 0$, and it must be the case that if $\mathrm{OPT}_R - \mathrm{OPT}_L = d$ then $p_r = 0$.

3. Suppose we have an algorithm A that achieves competitive ratio α for the special case that each cost vector (c_l, c_r) has either $c_l = 0$ or $c_r = 0$. Show how this can be used to get competitive ratio α without these restrictions. I.e., show that in constructing an algorithm, we may assume without loss of generality that each vector has $c_l = 0$ or $c_r = 0$.

Problems: (problem 5 is worth double)

4. Give a deterministic algorithm with competitive ratio 3 for this problem.

5. Give a randomized algorithm with competitive ratio 2 against an oblivous adversary for this problem. Hint 1: use results of exercises 1,2,3. Hint 2: define $\Delta = p_r - 1/2 = 1/2 - p_l$ and use the potential function $\Phi = d\Delta^2$. Hint 3: this is the hardest of the five problems on this assignment.

   For half credit, give a randomized algorithm with competitive ratio 5/2.