CS 787: Advanced Algorithms	**Homework 3**
Out: 11/02/09	**Due: 11/23/09**

Guidelines

- This homework consists of a few exercises followed by some problems. The exercises are meant for *your practice only*, and do not have to be turned in. You are required to turn in the problems. We will provide solutions to all the questions.
- Collaboration is not allowed. Some of the problems are difficult, so please get started early. Late submissions do not get any credit.
- Prove all your claims, excluding what we have proved in class.
- Please typeset your answers (preferably using laTeX).

Exercises

1. In class we studied the following two LP relaxations of the min-congestion routing problem, and claimed that they are equivalent. Prove that given a solution to the second LP, you can come up with a solution of equal value to the first one in time polynomial in the size of the graph n. (Note that in order to do this in polynomial time your solution to the first LP must assign non-zero values to at most a polynomial number of variables.)

$$\begin{array}{rcll} \min t & \text{s. t.} & & \\ \sum_{P\in\mathcal{P}_i} x_P & = & 1 & \forall i \\ \sum_i \sum_{P\in\mathcal{P}_i, P\ni e} x_P & \leq & t & \forall e \\ x_P & \geq & 0 & \forall i, P\in\mathcal{P}_i \\ t & \geq & 0 & \end{array} \qquad \left\| \qquad \begin{array}{rcll} \min t & \text{s. t.} & & \\ \sum_{e\in\delta^-(s_i)} x_{i,e} & = & 1 & \forall i \\ \sum_{e\in\delta^-(v)} x_{i,e} & = & \sum_{e\in\delta^+(v)} x_{i,e} & \forall i, v\neq s_i, t_i \\ \sum_i x_{i,e} & \leq & t & \forall e \\ x_{i,e} & \geq & 0 & \forall i, e \\ t & \geq & 0 & \end{array}$$

2. Suppose that the min-cut in a graph is of size k. Modify the randomized min-cut algorithm discussed in class such that for any cut S of size αk, the algorithm returns S with probability roughly $n^{-\alpha}$.
3. Prove that FIFO is k-competitive for caching.

Problems

1. Give a deterministic approximation algorithm for the max-cut problem that achieves a 2-approximation. (It should not be necessary to use the SDP-relaxation for this problem.)
2. Consider adapting the randomized min-cut algorithm from class to finding an s-t min-cut in a graph—at every stage, we have an s-supernode and a t-supernode. Of all the edges that do not go between the s-supernode and the t-supernode, we pick a random edge and contract it.
   (a) Show that there are graphs for which this algorithm produces an s-t min-cut with exponentially low probability.
   (b) What is the largest number of distinct s-t min-cuts that a graph can have? Give a graph that achieves this number.

3. Recall that in class we prove that randomized 1-bit LRU is $2H_k$ competitive for the caching problem (k is the size of the cache, and $H_k = \sum_{i \leq k} 1/i$.)

   (a) Prove that when the number of pages N is $k+1$, randomized 1-bit LRU is H_k-competitive.

   (b) Give an example of a request sequence for $k = 2$ and $N = 4$ where this algorithm is *not* H_k competitive.

4. The **online bin-packing** problem is a variant of the knapsack problem. We are given an unlimited number of bins, each of size 1. We get a sequence of iems one by one (each of size at most 1), and are required to place them into bins as we receive them. Our goal is to minimize the number of bins we use, subject to the constraint that no bin to should be filled to more than its capacity. In this question we will consider a simple online algorithm for this problem called **First-Fit** (FF). FF orders the bins arbitrarily, and places each item into the first bin that has enough space to hold the item. Prove that FF has competitive ratio 2. (Extra credit: Prove that its competitive ratio is $7/4$.)

5. In the **online call admission** problem, we are given a graph (think of it as a telephony network). We get a request sequence of calls each of which is a path in the graph. Our task is to either admit any given call, and remove the used up edges from the graph (they cannot be reused for any subsequent calls), or reject the call. Our objective is to admit as many calls as we can. We will discuss this problem in line graphs with D edges—these are graphs with $D+1$ nodes $v_0, v_1, \cdots, v_D$, and an edge between v_i and v_{i+1} for every $i \in [0, D-1]$.

   (a) Prove that any deterministic algorithm must have a competitive ratio of $\Omega(D)$ on this graph.

   (b) Assume $D = 2^k$ for some k, and consider the following randomized algorithm. Pick an integer $i \in [1, k-1]$ u.a.r. If a call contains between 2^i and 2^{i+1} edges, and does not overlap with any previously admitted call, then admit the call, otherwise reject it. Prove that this algorithm has a competitive ratio of $O(k) = O(\log D)$. *Hint: What is the c.r. of this algorithm if* **all** *calls have between 2^i and 2^{i+1} edges?*

6. **(Extra credit: Tracking a moving target.)** Here is a variation on the deterministic Weighted- Majority algorithm, designed to make it more adaptive.

   (a) Each expert begins with weight 1 (as before).

   (b) We predict the result of a weighted-majority vote of the experts (as before).

   (c) If an expert makes a mistake, we penalize it by dividing its weight by 2, but only if its weight was at least $1/4$ of the average weight of experts.

   Prove that in any contiguous block of trials (e.g., the 51st example through the 77th example), the number of mistakes made by the algorithm is at most $O(m + \log n)$, where m is the number of mistakes made by the best expert in that block, and n is the total number of experts.