Algorithmic aspects of communication networks	Assignment 2
OR 739 Spring 1997	February 10, 1997

Due date: Friday, February 21.

1. Give an $O(\log n)$-competitive algorithm for admission control on a tree with parallel edges, by modify the $O(\log n)$-competitive admission control algorithm for a full binary tree.

   Hint: Show that there exists a node v with the property that deleting v no remaining subtree has more than $n/2$ nodes. Instead of paths whose top node is the root of the tree, take the paths that go through node v. Use this idea recursively to replace the other levels of the full binary tree in the algorithm.

2. Give an $\Omega(n)$ lower bound on the competitive ratio of any *deterministic* on-line algorithm for admission control on a path of n nodes, where each edge can be used at most once. Extend this bound to a $\Omega(n^{1/k})$ lower bound if edges can be used at most k times. **Hint** for the extension: try $k = 2$ first.

3. Consider a network consisting of a central high bandwidth part of m_c edges, where each link can support k connections, where $k \approx \log_2(2m_c)$. From this high bandwidth central part there are lower bandwidth connections to outside parts. These outside connections are hanging off from several nodes in the central part, they each form tree structures, and the tree structures have no more than n_t nodes each. Give an $O(\log(n_t m_c))$ competitive algorithm for routing and admission control on this network. You may assume a solution to Problem 1 even if you did not solve that problem.

4. Consider the congestion control problem where connections do not last forever, and their duration is not known in advance. The model is that calls (pair of nodes to be connected) arrive on-line, must be routed; and you only discover how long a call lasts when it terminates. In this case no decent routing policy can be developed without rerouting (i.e., there is a $\Omega(\sqrt{n})$ lower bound on the competitive ratio of any on-line algorithm). In this problem we will develop an $O(\log n)$ competitive algorithm that only uses limited rerouting.

   The algorithm is as follows: Let $f_\tau(e)$ denote the number of paths using edge e at time τ. We will use the cost function $c_\tau(e) = (3/2)^{f_\tau(e)}$. When a call arrives (terminals s and t) at some time τ, it gets routed on the minimum cost path using the cost function c_τ at the current time τ. Each connection should record the cost of this path, that we will denote by $cost(s,t)$, i.e., the cost at the time of routing. In addition of routing paths the algorithm will also do rerouting: for any on-going connection between two nodes s and t at time τ if the minimum cost path connecting them using the current cost c_τ is at least a factor of 2 cheaper than $cost(s,t)$ than we reroute onto the current min-cost path.

(a) Assume that at every moment in time all on-going connections can be routed via disjoint paths. Show that the algorithm will never have congestion more than $O(\log m)$, where m is the number of edges.

(b) Under the same assumption show that no connection will be rerouted more than $O(\log m)$ times.

(c) Devise a version of this method without the assumption that at all times all on-going connections can be routed via disjoint paths.