Approximation and Network Algorithms	**Problem Set 1**
CS 684 Fall 2001	**Due Friday, November 16, 2001**

(1) "Just in time" scheduling concerns scheduling of different operations so that material does not have to wait at any time too long. In this problem we consider a special case of such a scheduling problem, given by a graph G. Assume that the nodes of the graph indicate actions that need to get scheduled, and the problem is to assign a time $t(v)$ for each node v. The edges represent dependencies of the operations. In particular, for each edge $e = (v, w)$, operation v must occur before operation w. Moreover, we are given an upper and lower bound, l_e and m_e, on the allowed difference. Think of l_e as the time needed to get the output of operation v to where w takes place, and $m_e - l_e$ is the maximum allowed extra time the material is allowed to wait for operation w. More formally, we need to assign a time $t(v)$ for each node $v \in V$, so that $l_e \leq t(w) - t(v) \leq m_e$ for all edges $e = (v, w)$. Show how to solve this problem by reducing it to a minimum cost flow problem (or its dual).

(2) In this problem, we develop another primal-dual algorithm for the minimum cost maximum flow problem. Let $G = (V, E)$ be a flow network, with source s and think t, and capacities u_e and costs c_e on the edges $e \in E$. Assume also that the costs are nonnegative and integral, and let α denote the value of the maximum (s, t)-flow in G. As usual in primal-dual algorithms, we will start with 0 prices, i.e., let $p(v) = 0$ for all $v \in V$, and we will use the reduced cost of $c_e^p = c_e + p(v) - p(w)$ for edge $e = (v, w)$. The basic algorithm is as follows.

Find a maximum flow from s to t using only edges with zero reduced cost. If this flow has value α we are done. Else let S be the source side of the min cut, and raise the prices $p(v)$ by 1 for all nodes $v \notin S$. Now till a flow of value α is found, repeat the following. Form a restricted residual graph G_f^p of the current flow f, that consists of the subset of the regular residual graph using only the edges of 0 reduced cost. Find a max-flow in the residual graph, and use this flow to augment the current flow f.

(a) Show that the prices p and the current flow f are in equilibrium throughout the algorithm (and as a result if a flow of value α is ever found this is the minimum cost max-flow we are looking for).

(b) We want to measure progress by the increase in prices. Note that the price of node s is never changed, while each increases the price of node t. Show that if all edge costs $c_e \in \{0, 1\}$ then $p(t) \leq n$ throughout the algorithm (and hence the algorithm terminates after n maximum flow computations). Hint: consider one of the flow paths in the last ineration.

(c) The previous parts assume that all edges cost 0 or 1. Show that the same conclusion also holds if the costs are nonnegative integers, and there is a feasible flow of value α using only edges whose cost is either 0 or 1 (note that this flow might not be the minimum cost flow).

(d) Use the previous part repeatedly in a scaling algorithm to get a polynomial time algorithm to find the minimum cost maximum flow assuming that the costs are nonnegative integers. If you assume that the largest cost is less than 2^δ, then in iteration i round down all costs to integer multiples of $2^{\delta - i}$.

(3) Consider the following defense problem. You have a network $G = (V, E)$ with source s, sink t. Each edge e has two numbers attached, a capacity u_e and a cost c_e. Here the cost c_e is the cost to destroy the edge e. The network is needed to transport some α amount of flow from s to t. The enemy is assumed to have a budget B and the network is viewed insecure if there is a subset F of the edges of cost at most B such that destroying these edges the remaining network no longer supports a flow of value α.

(a) Prove that the problem if a given network is secure is NP-complete.

(b) Give an algorithm for the following approximate version of this security problem. Your algorithm can either conclude that the network is secure, or else should find a 2-approximate insecurity, meaning a set of edges F of total cost at most $2B$ such that the maximum flow in the remaining network is at most 2α. Hint: use maximum flows. Consider the special case when $B = \alpha$ first.

(4) Consider the facility location problem discussed in class on Friday, October 26. Here we want to consider the following fault tolerant variant. The input to the problem is the same, a set of facilities F with costs f_i for each $i \in F$, a set of clients, and distances between them. However, we want each client to be connected to two different facility, a prime connection and a secondary connection. If client j has $p(j)$ as its prime connection and $s(j)$ as his secondary connection, we charge connection cost $dist(j, p(i)) + \alpha dist(j, s(i))$ for client j, where $0 < \alpha < 1$ is a given discount factor, we change less for secondary connection, as they are only backups in case the primary connection fails. So the total cost is the sum of the costs of all open facilities, plus the connection costs of all clients. Give a 3-approximation algorithm for this problem.