CS 684: Approximation and Network Algorithms	**November 16, 2001**
Problem Set 2	**Due Wednesday December 5**

(1) Throughout this course we have seen many network design approximation algorithms, however they were all undirected, i.e., the problems were defined on undirected graph. Unfortunately, not much is known about directed problems. Here is one thing that can be done.

Consider a simple directed network design problem. There is a directed graph $G = (V, E)$ and costs of the edges $c_e \geq 0$ as usual. The goal is to find a minimum cost subgraph that is strongly connected (that is a subset $F \subseteq E$ such that, for all sets $S \neq \emptyset, V$ there is at least one edge leaving S and at least one edge entering S in F). Give a 2-approximation algorithm for this problem. Hint: recall that we know how to find the minimum cost directed tree in polynomial time.

(2) Consider the metric traveling salesman problem. You are given n nodes, and distances $d(i, j)$ between them. The goal is to find an ordering of the nodes $i_1, \ldots, i_n$ such that the sum of the distances $\sum_{k=1}^{n} d(i_k, i_{k+1})$ is as small as possible, where i_{n+1} is used to denote i_1. A standard 2-approximation algorithm is as follows. Consider the complete graph on the n nodes with distances as edge weights. Take a minimum cost spanning tree T on the edges. Take a DFS ordering of the tree of T starting at any node, and order the nodes in the order the DFS first visits them. This tour has cost at most twice the optimum.

Here we will give a different 2-approximation algorithm that is conjectured to give a much better bound. Unfortunately, this algorithm will only approximate the value of the solution, but will not provide a solution. Consider the following linear program with variables x_e for each edge of the complete graph.

$$\begin{aligned} \text{minimize} \sum_e c_e x_e & \\ x_e \geq 0 & \qquad \text{for all edges } e \in E \\ \sum_{e \in \delta(S)} x_e \geq 2 & \qquad \text{for all } \emptyset \neq S \subset V \end{aligned}$$

Note that this linear program can be solved in polynomial time despite its exponential size (e.g., it is the same program Jain's algorithm uses).

Show that this the minimum value of this linear program is a lower bound to the traveling salesman value. Show that this lower bound is at least 1/2 of the optimum.

(3) Consider the following network design problem. We will consider here a seemingly very general setting, however less general than Jain's network design problem. On the other hand, we'll prove stronger results.

There is an undirected graph $G = (V, E)$ with costs $c_e \geq 0$ on the edges $e \in E$. In addition, there is a function $f(S)$ for all subsets $S \subseteq V$. We are looking for a minimum cost set of edges $F \subseteq E$ such that $\delta_F(S) \geq f(S)$ for all $S \subseteq V$. To make this problem feasible assume that for all sets $\delta_E(S) \geq f(S)$, i.e., including all edges of G we do satisfy the requirement.

For this problem we assume that $f(.)$ is supermodular, that is for any two sets A and B we have that $f(A) + f(B) \leq f(A \cap B) + f(A \cup B)$. In this problem we want to design an analog to Jain's algorithm for this setting. We consider the corresponding linear program, where there is a variable x_e for each edge $e \in E$,

$$\begin{aligned} \text{minimize} \sum_e c_e x_e & \\ 0 \leq x_e \leq 1 & \qquad \text{for all edges } e \in E \\ \sum_{e \in \delta(S)} x_e \geq f(S) & \qquad \text{for all } S \subseteq V. \end{aligned}$$

Surprisingly, this general linear program can be solved in time polynomial in the size of the graph even in this general setting (f can be any supermodular function as long as given any set S we can compute $f(S)$ in polynomial time). You do not have to worry about how to do this.

Prove that any vertex solution of this linear program has all integer values. To start with, prove that if f is not all 0, there is at least one edge with $x_e = 1$.

(4) The multi-way cut problem is a special case of the classification problem that was discussed in class on 11/2 by Jon Kleinberg's guest lecture. The problem is as follows. There is a graph $G = (V, E)$ and costs $c_e \geq 0$ on the edges. There are k labels. There is also a subset of the nodes $T \subset V$ called terminals, whose labels are known. The problem is to find a labeling of the unlabeled nodes $V \setminus T$, so as to minimize the sum of the costs of all the edges whose ends have different labels. We'll give two different 2-approximation algorithms for this problem.

Select a label j at random. For each label $i \neq j$ define a cut problem as follows. Contract all terminals that have label i into one node s, contract all terminals whose label is not i to another node t. Find the minimum (s, t) cut, and among all minimum (s, t) cuts take the one closest to the source. Label all these nodes by i. At the end, label all unlabeled nodes j.

Prove that the min-cut computations for the labels $i \neq j$ can be done in parallel, i.e., for any two labels $i_1, i_2 \neq j$ the set of nodes labeled i_i and i_2 will be disjoint, even if we run each min-cut computation using the original labeling. Prove that the cost of the resulting labeling is at most twice the minimum possible cost no matter what the choice of label j was. Prove that the expected cost of the solution is at most $2(1 - \frac{1}{k})$ times the minimum possible cost of the label j was chosen uniformly at random.