CO453: Network Design – Winter 2007

Instructor: Chaitanya Swamy

Assignment 2

Due: Friday, February 2, 2007 after class

You must give a proof of correctness of any algorithm you design, and argue briefly why it runs in polynomial time. You may use any proof or algorithm covered in class directly.

Throughout $G = (V, E)$ with $|V| = n,\ |E| = m$; for a set $A \subseteq V,\ E[A] = \{(u, v) \in E : u, v \in A\}$.

Q1 [Kleinberg-Tardos]:

(a) Suppose you are given an undirected graph G with costs $c_e \geq 0$ on the edges that are *not* necessarily distinct. You are given a spanning tree T with the property that every edge $e \in T$ belongs to *some* MST of G (recall that if the edge costs are not distinct then the MST need not be unique). Can one conclude that T itself is an MST? Give a proof, or a counterexample. **(5 marks)**

(b) Consider the same question for the minimum-cost arborescence (MCA) problem when the underlying directed graph G is a *directed, acyclic graph* (DAG), that is, G has no directed cycles. We are given an arborescence T rooted at r such that every edge $e \in T$ belongs to some MCA (rooted at r) of G. Is T then an MCA? Give a proof, or a counterexample. **(5 marks)**

Q2 [Kleinberg-Tardos]:

In this question we consider a variation of Edmonds' minimum-cost arborescence algorithm done in class. We are given a directed graph $G = (V, E)$ with a root $r \in V$ and costs $c_e \geq 0$ on the edges.

(a) Recall that for every vertex v, we defined $y_v = \min_{(u,v)\in E} c_{uv}$, the cost of the cheapest edge entering v, and we set $c'_e = c_e - y_v$ for every edge e entering v. Now consider instead the following modified costs: define $c''_e = \max(0, c_e - 2y_v)$ for every edge e entering v. This new change is likely to make many more edges have 0 c''-cost. Suppose we find an arborescence T among the 0 c''-cost edges. Show that $c(T) \leq 2 \cdot c(T_{\mathsf{opt}})$, where T_{opt} is an MCA of G (wrt. original costs c). **(3 marks)**

(b) Unlike the equivalence shown in class between costs c and c', it need not be that a min-cost arborescence wrt. costs c is a min-cost arborescence wrt. costs c'', or vice-versa. Give an example to show that an MCA wrt. costs c'' is *not* an MCA wrt. costs c. **(2 marks)**

(c) Nevertheless, prove that the following correspondence holds: for any arborescence T, we have $c''(T) + \sum_{v \neq r} y_v \leq c(T) \leq c''(T) + 2\sum_{v \neq r} y_v$. **(2 marks)**

(d) Now suppose that we do not find an arborescence among the 0 c''-cost edges. We proceed as in the algorithm done in class. There must exist a 0 c''-cost cycle C. We contract this cycle C, recursively apply the same procedure on the contracted graph to obtain an arborescence in the contracted graph, and then we "open out" the cycle C (as in the algorithm from class) to obtain an arborescence T in G. Prove that the resulting arborescence T has c-cost at most twice the c-cost of an MCA T_{opt} of G (wrt. costs c), that is, $c(T) \leq 2 \cdot c(T_{\mathsf{opt}})$. **(8 marks)**

Q3: The METRIC FACILITY LOCATION problem is defined as follows. We are given an undirected complete graph $G = (V, E)$ (i.e., there is an edge between every pair of vertices) with edge costs $c_e \geq 0$. These costs form a metric, that is, they satisfy the triangle inequality: for any three nodes u, v, w we have $c_{uw} \leq c_{uv} + c_{vw}$. There is a special set of nodes $F \subseteq V$ called *facilities*, and another set of nodes $C \subseteq V$ called *clients*. We are also given a number K. We have to decide if there is a subset $F' \subseteq F$ of facilities, and a way of assigning each client j to a facility $i(j) \in F'$ such that $|F'| + \sum_{j \in C} c_{i(j)j} \leq K$. Prove that METRIC FACILITY LOCATION is *NP*-complete. **(5 marks)**

(We can think of F' as the set of facilities opened, and there being a *facility-opening* cost of 1 for opening any facility. Then the cost-term above is the total cost incurred in opening facilities, and assigning or connecting clients to facilities.)

(Hint: Consider a reduction from SET COVER.)

Q4: In this question, we will consider the MST problem from a game-theoretic perspective. One application of MSTs is in building a multicast tree that connects all the users of a service to the service provider. We model this as follows. We are given a complete undirected graph G with edge costs $c_e \geq 0$, and a *root* $r \in V$. Think of the root as the service-provider (e.g., an ISP), and the other nodes as users who require this service and need to be connected to the root via a spanning tree that is used for multicast. Then, the service-provider's problem is to build an MST. Additionally, the service provider wants to charge each user a price so that it can recover its cost incurred in constructing the multicast tree. To make this precise, we define a *cost-sharing* scheme as a function $\xi : 2^V \times V \mapsto \mathbb{R}_{\geq 0}$, where $\xi(S, v)$ denotes the price charged to v if S is the set of nodes (users) that are connected to the root. The root certainly does not get charged and users that are not connected should not pay, so we want

$$\forall S, \qquad \xi(S, r) = 0 \text{ and } \xi(S, v) = 0 \quad \text{if } v \notin S.$$

The service provider should be able to recover his cost, so we want that

$$\text{(Budget-Balance)} \qquad \forall S, \qquad \sum_{v \in S} \xi(S, v) = \mathsf{MST}(S),$$

where $\mathsf{MST}(S)$ denotes the cost of the MST in the graph $(S \cup \{r\}, E[S \cup \{r\}])$. Thus, we can view $\xi(S, v)$ for $v \in S$ as v's share of the total cost incurred in connecting nodes in S to r.

(a) Given an MST on $S \cup \{r\}$, here is one simple way of defining cost-shares. We root the MST at r, and for each $v \in S$, we set $\xi(S, v) = c_{uv}$, where u is the parent of v. (Also $\xi(S, v) = 0$ for every node $v \notin S$.) It is clear that ξ satisfies the budget-balance property. Economic considerations require that the cost-shares satisfy the following competitiveness property:

$$\text{(Competitiveness)} \qquad \forall S,\ A \subseteq S, \qquad \sum_{v \in A} \xi(S, v) \leq \mathsf{MST}(A).$$

The rationale is that if a subset A of users is charged a total price that is greater than the cost to connect them directly to the root, then a competitor could entice the users A to break away from S by charging them a lower price, and still be able to recover the cost $\mathsf{MST}(A)$ incurred in providing service to A. Prove that the above cost-sharing scheme ξ is competitive. **(5 marks)**

(b) Recall the reduction from the MST problem to the MCA problem done in class. We bidirect every edge to get $\bar{G} = (V, \bar{E})$, and for every $u, v \in V$, we set $\bar{c}_{uv} = \bar{c}_{vu} = c_{uv}$. Thus, the costs $\bar{c}$ are symmetric. An MCA for the instance $(\bar{G}, \bar{c})$ (with root r) yields an MST for (G, c) if we ignore the edge directions. (If S is the set of nodes to connect to r, then we find an MCA in the graph $(S \cup \{r\}, \bar{E}[S \cup \{r\}])$.) Edmonds' algorithm for MCA yields another natural way of defining cost-shares. We defined $y_v = \min_{e \text{ enters } v} \bar{c}_e$, and set the new costs $\bar{c}'_e = \bar{c}_e - y_v$ for every $e = (u, v)$. We can think of y_v as the initial cost-share of v, but these cost-shares may increase as we proceed; if we find a directed cycle 0 $\bar{c}'$-cost cycle C, then some/all of the nodes in C will need to pay for the edge that we pick later to enter the cycle C, since this edge is used to connect the nodes in C to r. In general, at any stage, each node a of our contracted graph represents a set $A \subseteq V$ of nodes of the original graph that have been contracted into a as a result of contracting cycles; we call A, a *supernode*. For a supernode A, we set y_A to be the y-value of the corresponding node in the contracted graph. If $A \subseteq V$ is not a supernode, we set $y_A = 0$. In this stage, we pick an edge e of cost y_A entering the set A. This cost will be shared by the nodes of A. A natural cost-sharing scheme is to charge this cost equally to all nodes in A, so the cost-share of every node in A increases by $\frac{y_A}{|A|}$. This yields the cost-shares,

$$(*) \qquad \xi(S, v) = \sum_{A \subseteq V : v \in A} \frac{y_A}{|A|} \quad \text{if } v \in S, \text{ and 0 otherwise.}$$

(Another option is to charge y_A completely to the node v this is entered by e in the original graph; this yields the cost-shares in part (a). Do you see why?)

We will use the above cost-sharing scheme, but will slightly modify Edmonds' algorithm to maintain symmetric edge costs in every stage. In the current algorithm even if the costs $\bar{c}$ may be symmetric, the new costs $\bar{c}'$ need not be symmetric (do you see why?). To handle this, for every node v, including r, we define $y_v = y = \min_{all \text{ edges } e} \bar{c}_e$. Notice that when we set $\bar{c}'_e = \bar{c}_e - y_v$ (for every $e = (u, v)$, every v), there will always be a 0 $\bar{c}'$-cost cycle (of length 2), perhaps one that contains r, and we allow the algorithm to contract such cycles. The rest of the algorithm is the same. We use the new y-values to calculate the cost-shares, as given by (*), except that we always set $y_A = 0$ for the supernode A containing r (since r is not charged). Prove that the cost-shares ξ so defined are budget-balanced and competitive. **(5 marks)**

(c) (Bonus question: can be handed in later, on Mon., Feb. 5)

Suppose now that each user $v \neq r$ has a utility $u(v)$ that it derives from receiving service (i.e., if it is connected to the root). Node v may then choose to "drop out" of the multicast if its cost-share $\xi(S, v)$ is larger than its utility $u(v)$. We would like that a node v drops out as soon as it realizes that its cost-share is too high. In this kind of setting, where no new users join in, the following property is desirable: suppose we are currently serving a set $S \subseteq V$ and a node $v \in S$ drops out, then the cost-shares of the *other users* in S only goes up. That is,

(Cross monotonicity)
$\forall S,\ v, \quad \xi(S \setminus \{v\}, u) \geq \xi(S, u)$ for every $u \neq v$ (if $v \notin S$ or $u \notin S$, the inequality holds trivially).

This is in fact what one would expect — if there are fewer users sharing the cost, then a user's cost-share should only go up. A cross-monotonic scheme prevents a user v from playing the following "waiting game" (and thereby misrepresenting its utility $u(v)$): v realizes that its cost-share is too high, but it waits for some other nodes to drop out and thereby decrease its cost-share; in a cross-monotonic scheme, v's cost-share only goes up as other nodes drop out. Prove that the cost-sharing scheme defined in part (b) is cross-monotonic. **(10 marks)**