CS5234: Combinatorial and Graph Algorithms

Problem Set 3

Solution Sketches

S-1. In this problem, you will improve the approximation ratio for Metric Steiner Tree.

(a) Given a cycle C containing n nodes, let $cost(C)$ be the sum of the edge weights in C. Prove that at least one edge in the cycle C has weight at least $cost(C)/n$.

Solution: This follows immediately from that definition of an average. A simple proof by contradiction would reason as follows: Assume that every edge has weight $< cost(C)/n$; since the cycle contains n edges, the total edge weight is then $< n \cdot cost(C)/n < cost(C)$, which is a contradiction.

(b) Given an instance of Metric Steiner Tree with n required nodes R, prove that a minimum spanning tree of R is a $2(1 - 1/n)$-approximation of the optimal Metric Steiner Tree. *Hint: Think about where a cycle is constructed in the original proof, and use the result from part(a).*

Solution: Consider an instance $G = (V, E)$ of the metric Steiner tree problem, and let n be the number of required nodes. Let T be the optimal steiner tree for the specified instance, and let $C' = DFS(T)$ be the cycle created by performing a depth-first traversal of T. Let C be the cycle C' where we skip Steiner nodes and repeated nodes. Note that C is a cycle on only the required nodes of the problem, and includes each required node exactly once.
By construction, we know that $cost(C') = 2cost(T) = 2 \cdot OPT$, and by the triangle inequality we know that $cost(C) \leq cost(C') \leq 2 \cdot OPT$. By part (a) above, we know that there is at least one edge e with weight at least $cost(C)/n$. Let T' be the tree that is realized by removing edge e from C.
Observe that $cost(T') \leq cost(C) - cost(C)/n \leq cost(C)(1 - 1/n)$. Combining inequalities, we conclude taht $cost(T') \leq 2 \cdot OPT(1 - 1/n)$. Finally, let M be a minimum spanning tree of the required nodes of G. Since M is a minimum spanning tree, we know that $cost(M) \leq cost(T') \leq OPT \cdot 2(1 - 1/n)$, concluding the proof.

S-2. Let $G = (V, E)$ be an undirected graph with nonnegative edge weights. Let S, the senders and R, the receivers, be disjoint subsets of V. The problem is to find a minimum cost subgraph of G such that for every receiver $r \in R$, there is at least one sender $s \in S$ such that there is a path connecting r to s in the subgraph. Give a factor 2 approximation algorithm that runs in polynomial time. *Hint: Consider introducing an additional vertex to the graph, and try building on an approximation algorithm (for a different problem).*

Solution.

Executive summary. We solve the problem by reducing it to the problem of finding a Steiner Tree. The reduction consists of adding a new node v that is connected to all the source nodes, and requiring the Steiner Tree to include both v and all the receivers. The resulting tree ensures that each receiver is connected to some source at a minimal cost.

Details. Let G' be a new graph that is equivalent to G, but also contains an additional node v that has an edge with zero cost connecting it to every source. Let Q be the set containing v and all the nodes in R. Let O be all the other nodes in the graph.

Execute the General Steiner Tree approximation algorithm on the new graph G' where Q is the set of required nodes and O is the set of optional (Steiner) nodes. Let T be the resulting Steiner Tree. Remove the node v from the tree T, along with all adjacent edges. The resulting graph is a 2-approximation of the optimal network connecting each receiver to some source.

Proof First, notice that the result graph connects every receiver to some source: the tree T contains a path from each receiver u to the special node v, since T is a Steiner tree that contains every receiver as well as v. Moreover, the path from a receiver u to v necessary traverses a sender, since all the edges adjacent to v are senders. Hence, when v is removed, the remaining graph still contains a path from u to a sender.

Next, we argue that the resulting graph is a 2-approximation of optimal. Let $OPT(SR)$ be an optimal solution to the sender/receiver problem. Let E be the set of edges in $OPT(SR)$ in graph G', along with all the edges connecting v to the senders. Notice that the resulting set of edges E is a valid Steiner Tree for G', in that it connects all the receivers and the node v.

Moreover, notice that the cost of the set of edges E is exactly equal to $OPT(SR)$, since the edges connecting v to the source nodes have cost zero. Thus, the cost of the optimal Steiner Tree $OPT(ST) \leq OPT(SR)$.

Finally, we observe that the solution returned by the algorithm above has cost equal to the cost of the Steiner tree approximation, i.e., $\leq 2OPT(ST)$. Thus, putting the inequalities together, we observe that the resulting algorithm is a 2-approximation of $OPT(SR)$.

Finally, since the algorithm involves constructing a graph (i.e., time $O(m + n)$), running the Steiner Tree approximation algorithm (i.e., polynomial time), and re-constructing a final graph (i.e., time $O(m + n)$), the resulting algorithm runs in polynomial time. □

Note. We used zero-cost edges in our graph G' on which we ran the Steiner Tree approximation algorithm. First, note that the approximation algorithm works correctly even with zero-cost edges. Second, if you prefer not to use zero-cost edges, you can choose some arbitrary small value ϵ, and notice that the proof proceeds identically (since the cost of these edges is added when constructing the Steiner Tree, but then subtracted later when constructing the final graph). However, it is cleaner to use zero-cost edges.

S-3. In this problem, we will consider the following variant of set cover: assume you are given an underlying set of n elements X and a collection of m sets $S_1, S_2, \ldots, S_m$ where each $S_j \subseteq X$. Assume that each element $x \in X$ is contained in *at most* 3 different sets.

(a) Give an integer linear program (ILP) that solves this set cover problem.

Solution: Assume the input to the set cover problem consists of elements $x_1, x_2, \ldots, x_n$ and sets $S_1, S_2, \ldots, S_m$. We define a variable s_j where $s_j = 1$ if and only if set S_j is included in the cover, and $s_j = 0$ otherwise. (Notice that s_j will be an integer variable with values in the set $\{0, 1\}$.) We can now formulate the following linear program:

$$\begin{aligned} \min \sum_{j=1}^{m} s_j \qquad & \text{where:} \\ \sum_{j:x_k \in S_j} s_j \;&\geq\; 1, \quad \forall k \in \{1, \ldots, n\} \\ s_j \;&\geq\; 0, \quad \forall j \in \{1, \ldots, m\} \\ s_j \;&\leq\; 1, \quad \forall j \in \{1, \ldots, m\} \\ s_j \;&\in\; \mathbb{Z}, \quad \forall j \in \{1, \ldots, m\} \end{aligned}$$

In order to see that this is correct, we need to show two things: every feasible solution to this ILP is a solution to the set cover problem, and every solution to the set cover problem (or, at least the optimal solution to the set cover problem) is a feasible solution to this ILP.

First, assume that $s_1, \ldots, s_m$ is a feasible solution to this ILP, and interpret it as a set cover solution as above (i.e., set S_j is included in the cover if and only if $s_j = 1$.) Since $s_j \in \{0, 1\}$, this interpretation makes sense. Moreover, if the solution to the ILP is feasible, then we know that for every element x_i, at least one set covering x_i must be in the cover due to the constraint: $\sum_{j:x_i \in S_j} s_j \geq 1$.

Conversely, assume that $s_1, \ldots, s_m$ represents a solution to the set cover problem. Then by definition, $s_j \in \{0, 1\}$. And, in addition, for every x_i, it satisfies the constraint: $\sum_{j:x_i \in S_j} s_j \geq 1$ since at least one S_j must cover x_i.

Putting these two facts together, we conclude that this ILP yields a solution to the set cover problem.

(b) Explain how to round the ILP from part (a) to achieve a 3-approximation algorithm. Prove that your algorithm is correct.

Solution: Relax the ILP to a linear program by removing the constraint that $s_j \in \mathbb{Z}$. Let $s_1, \ldots, s_m$ be the solution to this (relaxed) linear program. We now round the solution as follows: for all $j \in \{1, \ldots, m\}$, add set S_j to the set cover C if $s_j \geq 1/3$. We now argue that the resulting algorithm yields a 3-approximation of the minimum set cover.

First, we observe that the result is a valid set cover. Consider some element x_i. Recall that x_i is included in at most 3 different sets. For now, let us call these three sets S_A, S_B, S_C. We know, via the constraint on the LP, that $s_A + s_B + s_C \geq 1$. This implies that: $s_A \geq 1/3$ or $s_B \geq 1/3$ or $s_C \geq 3$. Thus, our algorithm adds one of the three sets to the set cover, guarantee that element x_i is covered.

We now need to argue that the resulting cover is at least a 3-approximation of the minimum-sized set cover. Let OPT the minimum-sized set cover, and let C be the set cover constructed by the approximation algorithm. Notice that $\sum_{j \in \{1,\ldots,m\}} s_j \leq |OPT|$, i.e., the solution to the relaxed LP is no greater than the solution to the original ILP (which is equal to OPT). And, we only include a set S_j if $s_j \geq 1/3$; from this we conclude that $|C| \leq 3 \sum_{j \in \{1,\ldots,m\}} s_j$ (since we effectively increase s_j to 1 if it is $\geq 1/3$, and decrease it to 0 otherwise). Putting these two inequalities together, we conclude that $|C| \leq 3|OPT|$, i.e., this approximation algorithm yields a 3-approximation of the minimum-sized set cover.