Advanced Approximation Algorithms **(CMU 15-854B, Spring 2008)**

Lecture 10: Group Steiner Tree problem

14 February, 2008

Lecturer: Anupam Gupta *Scribe: Amitabh Basu*

1 Problem Definition

We will be studying the Group Steiner tree problem in this lecture. Recall that the classical Steiner tree problem is the following. Given a weighted graph $G = (V, E)$, a subset $S \subseteq V$ of the vertices, and a root $r \in V$, we want to find a minimum weight tree which connects all the vertices in S to r. The weights on the edges are assumed to be positive. We will now define the Group Steiner tree problem.

The input to the problem is

- A weighted graph $G = (V, E)$ with positive edge weights c_e for every edge $e \in E$. As is usual, $n = |V|$.
- A specified vertex r.
- k subsets (or groups) of vertices $g_1, \ldots, g_k$, $g_i \subseteq V$.

We want to output a subtree T such that T "connects" each g_i to r. By "connect" we mean that there should exist a vertex $v \in g_i$ such that there is a path in T from r to v.

We will study the problem when the input graph is a tree itself and the root of the tree is the specified vertex r. So we need to connect each group g_i to the root of the tree. Moreover, we will assume that each group g_i is a subset of the leaf vertices only. No internal vertices are contained in any group. We will refer to the input tree by T and the solution subtree by T'. See Figure 1 for an illustration. Each group is denoted by a different mark (disc, cross, box etc.) and the heavy edges are the edges output.

2 Approximation Results for Group Steiner Tree

It is easy to show the following.

Proposition 2.1. *Group Steiner Tree is at least as hard as Set Cover.*

Proof. Given an instance of set cover, we construct a star with the central vertex as r. Each edge of the star corresponds to a set of the set system, and has weight equal to the cost of that set. Each element $i \in 1, \ldots, N$ of the ground set corresponds to a group g_i in the Group Steiner Tree problem. A group g_i consists of all the leaf vertices incident on edges corresponding to the sets which contain element i.

This reduction shows that Group Steiner Tree can be used to solve Set Cover. $\square$

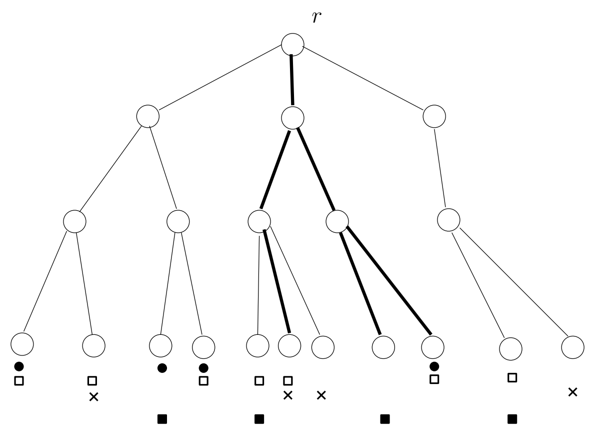

Figure 1: Example illustrating the Group Steiner Tree problem

The main result of today's lecture is the following theorem.

Theorem 2.2. *Group Steiner tree has $O(\log n \log k)$ approximation.*

On the hardness of approximation side, we have the following result.

Theorem 2.3. *Group Steiner Tree problem is $\Omega(\log^{2-\epsilon} n)$-hard, for all $\epsilon > 0$.*

To help prove Theorem 2.2, we will establish the next lemma.

Lemma 2.4. *There exists a randomized algorithm (call it A) that takes an instance $\mathcal{I}$ of the Group Steiner Tree problem and outputs a (random) set of edges $\widehat{E}$ such that*

1. $\mathbf{E}[$*cost of edges in* $\widehat{E}] \leq OPT(\mathcal{I})$
2. *For any group g,* $\mathbf{Pr}[\widehat{E}$ *connects the root to* $g] \geq \frac{1}{\log |g|} \geq \frac{1}{\log n}$

Let us first show how the above lemma implies Theorem 2.2. Run the algorithm A guaranteed by the lemma independently on the instance $\mathcal{I}$ $L = O(\log n \log k)$ times. Say we get $\widehat{E}_1, \hat{E}_2, \ldots, \widehat{E}_L$ as the sets of edges output during the different runs. Let $\widehat{E} = \cup_i \hat{E}_i$.

If some group is not connected to the root in $\widehat{E}$, find the cheapest way to connect to this group to the root by a path. The cost of such connections are bounded in the following simple observation.

Fact 2.5. *The cost of the cheapest path from some vertex in g to the root is at most $OPT((I)$.*

Proof. Since OPT connects some vertex v in g to the root r, the path in OPT that connects v and r is at least the cheapest path that was taken as part of the solution. □

However, the probability that group g does not get connected to r even after L rounds is pretty low because of condition 2 in the lemma. More precisely,

$$\begin{aligned}\mathbf{Pr}[\text{group } g \text{ does not get connected in } L \text{ rounds}] &\leq (1 - \tfrac{1}{\log n})^L \\ &\leq e^{L/\log n} = e^{\log k} \\ &\leq 1/k\end{aligned}$$

The first inequality follows from condition 2 of the lemma. So the expected cost of the final solution is at most

$$\begin{aligned}&\textstyle\sum_i \mathbf{E}[\text{cost of } \widehat{E}_i] + \sum_{\text{groups g}} \mathbf{Pr}[\text{g is not connected}] \cdot OPT(\mathcal{I}) \\ &= L \cdot OPT(\mathcal{I}) + \textstyle\sum_{\text{groups g}} \mathbf{Pr}[\text{g is not connected}] \cdot OPT(\mathcal{I}) \\ &\leq (L+1) \cdot OPT(\mathcal{I})\end{aligned}$$

The second term in the first line follows from Fact 2.5. The first term in the second line follows from condition 1 of Lemma 2.4. The third line follows from the second because of the probability bound proved above. Plugging in the value of L, we get the result of Theorem 2.2.

We now turn to prove Lemma 2.4.

3 Randomized Rounding for Group Steiner Tree

We first write a natural Integer Linear Program for the Group Steiner Tree problem. We will then use the technique of randomized rounding on the LP relaxation to prove Lemma 2.4. The ILP for the problem is as follows.

$$\begin{aligned}&\min \textstyle\sum_e c_e x_e \\ &\text{s.t.} \\ &\textstyle\sum_{e \in \partial S} x_e \quad \geq \quad 1 \qquad \forall S \subseteq V \text{s.t. } S \text{ separates } r \text{ from group } g_i \text{ for some } i\end{aligned}$$

In the above formulation, ∂S denotes the set of edges with *exactly* one end-point in S. Note that there are potentially an exponential number of constraints in the LP. This is a potential problem to solving the LP in polynomial time. However there are two ways around it.

First, there exists a compact formulation for the above ILP, i.e. one with only a polynomial number of constraints and having the same integral feasible solutions.

Second, we can invoke a theorem proved by Grotzchel, Lovasz and Schrijver [1], which states that solving an LP is polynomially equivalent to separation. By separation, we mean that given a point x, we should be able to decide if it feasible and if not, then exhibit a constraint being violated. This theorem uses the ellipsoid algorithm and binary searching to "guess" the value of OPT.

3.1 Rounding the LP relaxation

The first natural idea is to pick each edge in E independently with probability x_e, as we have been doing in previous lectures. As before, by linearity of expectation, the expected cost of the edges picked will be equal to the LP-OPT which is at most $OPT(\mathcal{I})$. So condition 1 of the lemma can be satisfied. However, with this strategy, condition 2 of Lemma 2.4 is typically far from being satisfied, as can be seen in the following example (Figure 2).

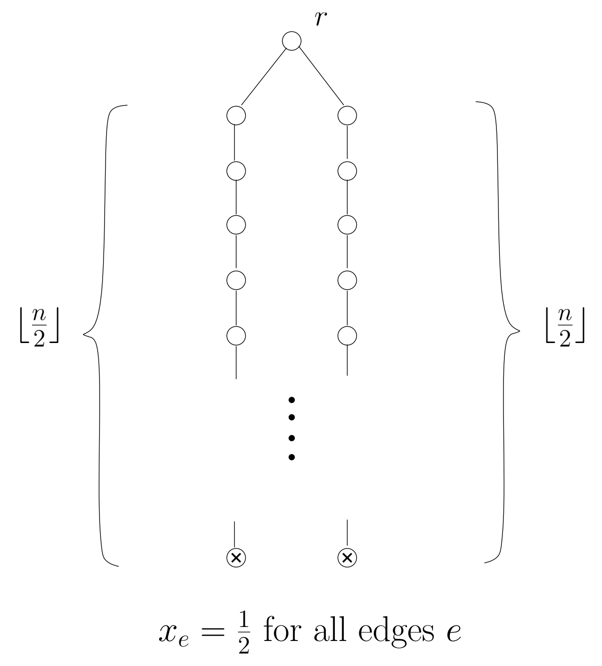
r
$\lfloor \frac{n}{2} \rfloor$
$\lfloor \frac{n}{2} \rfloor$
$x_e = \frac{1}{2}$ for all edges e

Figure 2: Independently picking edges can be bad

In Figure 2, the two leaf nodes are in the same group and the probability that one of them connects to the root is $2 \cdot 2^{-\frac{n}{2}}$, which is extremely tiny. To satisfy condition 2 of the lemma, we needed this probability to be at least $1/\log 2$. So complete independence across edges leads to situations where the connection probability of a group becomes very small.

The following is the fix suggested by Garg, Konjevod and Ravi [2]. For an edge e, we denote its parent edge by $p(e)$.

1. "Mark" each edge with probability $x_e/x_{p(e)}$.[1] If there is no parent edge, then mark the edge with probability x_e.
2. Pick an edge e if all its ancestor edges are marked, and e itself is marked.

Fact 3.1. $\mathbf{Pr}[$*edge e is picked*$] = x_e$

Proof. Consider any edge e. Say it has i ancestor edges. then the probability that it gets picked is the probability that e *and* all its ancestors are marked. Therefore,

$$\mathbf{Pr}[\text{edge } e \text{ is picked}] = \frac{x_e}{x_{p(e)}} \cdot \frac{x_{p(e)}}{x_{p(p(e))}} \cdot \frac{x_{p(p(e))}}{x_{p(p(p(e)))}} \cdots \frac{x_{p^{i-1}(e)}}{x_{p^i(e)}} \cdot x_{p^i(e)} = x_e$$

[1]Note that the constraints of the LP force $x_e \le x_{p(e)}$, because otherwise, we can reduce x_e to the value of $x_{p(e)}$ and get a lower cost LP. If we cannot reduce it to $x_{p(e)}$, then at some point some constraint becomes tight, and still $x_{p(e)} < x_e$. Now for this constraint, the "top" vertex of e is included in it (since e is in the cut). Now consider the cut without this vertex. $p(e)$ is now included into the cut and e (and possibly some other edges) exit. But if $x_{p(e)} < x_e$, this constraint is now violated, and was being violated even before reducing x_e.

□

So this implies that the expected cost of the set of edges picked will still be equal to the LP-OPT and hence at most $OPT(\mathcal{I})$.

Now we turn to prove that the above procedure provides the lower bound we needed for condition 2 of Lemma 2.4.

3.2 Analysis for connecting a group to the root

Fix a group g. Denote by SUCCESS the event that g is connected to r and FAILURE denotes the event that no vertex of g got connected to r. Given an LP-solution x, we will construct another LP-solution x' with the property that

$$1 - 1/\log|g| \geq \mathbf{Pr}[\text{failure to connect } g \text{ using } x'] \geq \mathbf{Pr}[\text{failure to connect } g \text{ using } x]$$

In fact, we simply need $x'_e \leq x_e$ for all edges e. We will use the following claim to argue about x.

Claim 3.2. *If $x'_e \leq x_e$ for all edges e, then $\mathbf{Pr}[\textit{FAILURE using } x'] \geq \mathbf{Pr}[\textit{FAILURE using } x]$.*

Proof. This can be proved by considering one edge at a time, i.e. x' differs from x at only one edge. We can then use induction to prove the result. Say x' is smaller at edge e. Let the subtree rooted below e be R. It is easy to see that the events which connect leaves outside R are independent of events which connect vertices in R. So we simply need to argue about the subtree below e. For simplicity, we assume e has only 2 child-edges - the algebra for more than 2 children is more messy and adds no insight.

Let these two child-edges be e_1 and e_2. Consider the subtrees rooted at e_1 and e_2 and let the probabilities of connecting to the e_1 and e_2 in *these* respective subtrees be p_1 and p_2. Note that by changing x_e, p_1 and p_2 are unchanged. p_1 and p_2 are just probabilities to connect to e_1 and e_2, not to the real root. Since the probability of marking in these subtress are unaffected by changing x_e, p_1 and p_2 are unaffected.

Now the probability of FAILURE for vertices in R is simply

$$1 - x_e + x_e((1 - \frac{x_{e_1}}{x_e}) + \frac{x_{e_1}}{x_e} \cdot p_1)((1 - \frac{x_{e_2}}{x_e}) + \frac{x_{e_2}}{x_e} \cdot p_2)$$

The first term is the probability that e was not picked. The second term is the probability of failure assuming e was picked. Once we simplify the above expression, we get

$$1 - x_{e_1}(1 - p_1) - x_{e_2}(1 - p_2) + \frac{x_{e_1} x_{e_2}(1 - p_1)(1 - p_2)}{x_e}$$

which clearly increases as x_e is decreased. □

We now construct x' from x as follows.

1. Delete all edges (that is set their value to 0) incident on leaves from groups other than g. Also remove unnecessary edges, i.e. edges which cannot be on a path from the root to any vertex in g.

2. Reduce x values, so that the solution is minimally feasible. This implies that the flow going down to the leaves is exactly 1.

3. Take the new x values, and round down to the nearest power of 2. Note that we would reduce any x_e by at most half. So the cut value may reduce by at most a factor of 2. So the net flow now is $\geq \frac{1}{2}$

4. Next delete all edges with new x-value $\leq 1/4|g|$, and again delete unnecessary edges. The new cut (or flow) value may fall by at most $|g| \cdot 1/4|g| = 1/4$, since there are at most $|g|$ leaves. So the final flow is at least $1/4$.

5. If $x_e = x_{p(e)}$, then merge such edges if $p(e)$ does not have any other children edges. Else, "contract" e, so that its children become children for $p(e)$. This can be done because e will be picked with probability 1, if $p(e)$ is picked. So we can analyze on this new tree with a single edge with value $x_{p(e)}$.

Observation 3.3. *The height of the tree at the end of the above operations is at most* $O(\log |g|)$

Proof. At each level, the x value decreases by a factor 2, since all the edges were rounded down to powers of 2 and edges with the same weight were merged. Because of the 4th step above, the x-value can go down to at most $1/4|g|$. □

The final piece that we need to complete the picture is the following technical lemma, called Janson's inequality. Given a ground set S of items and $P_1, P_2, \ldots$ subsets of S, suppose we choose a subset S' randomly such that each element in S is added to S' independently with some probability. Let $\mathcal{E}_i$ be the event that $P_i \subseteq S'$, i.e., the entire set P_i is chosen by the random process.

Let $\mu = \sum_i \mathbf{Pr}[\mathcal{E}_i]$ and $\Delta = \sum_{i \sim j} \mathbf{Pr}[\mathcal{E}_i \cap \mathcal{E}_j]$, where $i \sim j$ denotes that $\mathcal{E}_i$ and $\mathcal{E}_j$ are dependent.

Lemma 3.4 (Janson's Inequality)**.** *Given the above definitions, if we have $\mu \leq \Delta$, then the probability that none of the events $\mathcal{E}_i$ happen is*

$$\mathbf{Pr}[\cap_i \bar{\mathcal{E}_i}] \leq exp\{\frac{-\mu^2}{2\Delta}\}$$

In our case, the set S is the set of all edges E of the graph, the subsets P_v are the edges on the path from some vertex $v \in g$ to the root, and hence the "good" event $\mathcal{E}_v$ denotes the event that all edges on P_v are marked, and hence vertex v is connected to the root in the random set $\widehat{E}$.

We define the events on the new tree and x' that was constructed above. As discussed earlier, we only need to bound the probability of FAILURE on this new tree and x'. Claim 3.2 will give the result for the original tree and x.

First note that $\mathbf{Pr}[\mathcal{E}_v] = x'_{\text{parent edge}}$. So, $1/4 \leq \sum \mathbf{Pr}[\mathcal{E}_v] \leq 1$, since we argued that the flow is at least $1/4$, and by minimality the flow started with 1.

We now claim the following.

Claim 3.5. $\Delta = \sum_{i\sim j} \mathbf{Pr}[\mathcal{E}_i \cap \mathcal{E}_j]$ *is at most* $O(\log n)$.

Assuming the claim above, we use Lemma 3.4 to bound the failure probability which is $\mathbf{Pr}[\cap\bar{\mathcal{E}}_i]$. Plugging the values of μ and Δ, we get that

$$\mathbf{Pr}[\cap\bar{\mathcal{E}}_i] \leq exp\{-\frac{1}{\log|g|}\} \approx 1 - \frac{1}{\log|g|}$$

So the probability of SUCCESS is at least $\frac{1}{\log|g|}$. This ensures condition 2 of lemma 2.4.

Now we prove Claim 3.5.

Proof. Suppose the height of the merged tree is H.

For some vertex $u \in g$, let us first show that $\Delta_u = \sum_{w\sim u} \mathbf{Pr}[\mathcal{E}_u \cap \mathcal{E}_w] \leq O(H)\cdot x'_u$. Summing this over all $u \in g$ gives us $O(H)$, since the x'_u's sum up to at most 1 by minimality of x', and hence completes the proof.

For any $w \in g$, let e be the lowest edge shared by the path from u to the root, and the path from w to the root. Abusing notation, we refer to the edges incident on u and w by the same letters. As noted earlier, $\mathbf{Pr}[\mathcal{E}_u] = x'_u$. Let the child-edge of e on the path to w be denoted by $c(e)$. Now,

$$\mathbf{Pr}[\mathcal{E}_w|\mathcal{E}_u] = \frac{x'_w}{x'_{p(w)}} \cdot \frac{x'_{p(w)}}{x'_{p(p(w))}} \cdots \frac{x'_{c(e)}}{x'_e} = \frac{x'_w}{x'_e}$$

Hence, $\mathbf{Pr}[\mathcal{E}_u \cap \mathcal{E}_w] = (x'_u x'_w / x'_e)$.

Now we sum over all w such that the paths from u to root and w to root have the edge e as their least common ancestor edge as above. Now use the fact that $\sum_{\text{such } w} x'_w \leq O(x'_e)$, since all the flow going to such w's must have been supported on the edge e before rounding down. Hence, the sum of $\mathbf{Pr}[\mathcal{E}_u \cap \mathcal{E}_w]$ over all the relevant w for some edge e is $O(x'_u)$.

Now there are at most H edges on the path from u to the root. So we get $\Delta_u = O(H\cdot x'_u)$, as promised.

Now using Observation 3.3, we get the desired bound on $\Delta = \sum_{i\sim j} \mathbf{Pr}[\mathcal{E}_i \cap \mathcal{E}_j]$.

□

References

[1] M. Grotschel, L. Lovasz and A. Schrijver, *The ellipsoid algorithm and its consequences in combinatorial optimization*, Combinatorica 1 (1981) 169-197.

[2] N. Garg, G. Konjevod, R. Ravi, *A polylogarithmic approximation algorithm for the group Steiner tree problem*, *SODA 2000*