601.435/635 Approximation Algorithms **Lecturer:** Michael Dinitz
Topic: Randomized Rounding: Group Steiner Tree on Trees **Date:** 3/5/26

13.1 Introduction

Today we're going to see a famous problem, Group Steiner Tree, which isn't discussed in the book, but which is important for a few reasons. It's important historically, but also is important for the techniques that we'll use. In particular, we'll use some probabilistic analysis techniques that are more sophisticated than Chernoff bounds, and we'll see randomized rounding techniques that are (slightly) more sophisticated than independent randomized rounding. Moreover, GST is one of the rare problems that is difficult even on trees, and in fact today we're really only going to focus on the tree case. This will motivate the next couple of lectures, when we'll talk about ways of *reducing* general graphs to trees.

13.2 Definition: Group Steiner Tree (GST)

- **Input:**
  - Graph $G = (V, E)$
  - Edge costs $c : E \to \mathbb{R}_{\geq 0}$
  - Root vertex $r \in V$
  - K groups $g_1, g_2, \ldots, g_k$, where each $g_i \subseteq V$
- **Feasible:** Tree T such that $\forall i \in [k]$, $\exists v \in g_i$ such that T has a path between r and v.
- **Objective:** $\min \sum_{e \in T} c(e)$

To get some intuition for this definition, note that if every group is a single vertex, then this is exactly Steiner Tree (the root vertex doesn't make any real difference). So the difference is that in GST, we only have to connect one vertex (of our choice) from each group. In other words, we now have an extra set of choices: instead of just deciding on how to connect the terminals, we now need to decide *which* terminals to connect (subject to connecting at least one from each group) and then decide how to connect them.

13.3 Hardness

Before we get to algorithms, let's think about what kind of approximations we can hope to get.

Theorem 13.3.1 *Set Cover is a special case of GST on trees.*

Proof: Let $(U, \mathcal{S})$ be a set cover instance. Then construct a star with

- Leaf for each $S \in \mathcal{S}$

- Group g_e for each $e \in U$ where $g_e = \{S \in \mathcal{S} : e \in S\}$.

Consider a set cover $S_1, \ldots, S_k$. Then $S_1, \ldots, S_k$ is a GST solution. Conversely, consider a GST solution $S_1, \ldots, S_k$. Then $S_1, \ldots, S_k$ is a set cover. ∎

Corollary 13.3.2 *It is NP-hard to approximate GST better than $\Omega(\log n)$.*

Proof: This follows from the $\Omega(\log n)$-hardness of Set Cover and Theorem 13.3.1. There is one subtlety: the n in Set Cover is the number of elements, while n in the instance of GST that we created is the number of sets (m). But the hard instances of Set Cover are ones in which $m = O(n^c)$ for some constant c, so $\log m = \log n$ and we get the same hardness (up to constants). ∎

It turns out that GST is even harder than Set Cover. In fact, it was basically the very first problem for which a polylogarithmic hardness result was shown:

Theorem 13.3.3 [Halperin, Krauthgamer 2003] For all constant $\epsilon > 0$, GST on trees is $\Omega(\log^{2-\epsilon} n)$-hard to approximate.

13.4 Approximation Algorithm

13.4.1 Assumptions

We're going to assume that G is a tree – we'll talk more about the non-tree case next class, but for now, note that this *is not* without loss of generality. But once we assume that G is a tree, we can assume without loss of generality that if $v \in g_i$ for any i, then v is a leaf (this is without loss of generality since we can always add an extra zero-cost edge to a new leaf for any internal terminal).

Theorem 13.4.1 [Garg, Konjevod, Ravi 2000] There exists an $O(\log n \log k)$-approximation to GST on trees.

13.4.2 Linear Programming Relaxation

Consider the following LP relaxation, which basically requires that for every cut which separates r from some group, at least one edge crosses the cut:

$$\begin{aligned} \text{minimize:} \quad & \sum_{e \in E} c_e \cdot x_e & \textbf{(GST-LP)} \\ \text{subject to:} \quad & \sum_{e \in (S, \bar{S})} x_e \geq 1 \quad \forall i \in [k],\ \forall S \subseteq V \text{ such that } r \in S,\ g_i \cap S = \emptyset \\ & 0 \leq x_e \leq 1 \quad \forall e \in E \end{aligned}$$

As usual, it's straightforward to prove that if we have integrality constraints, this is an exact formulation of GST.

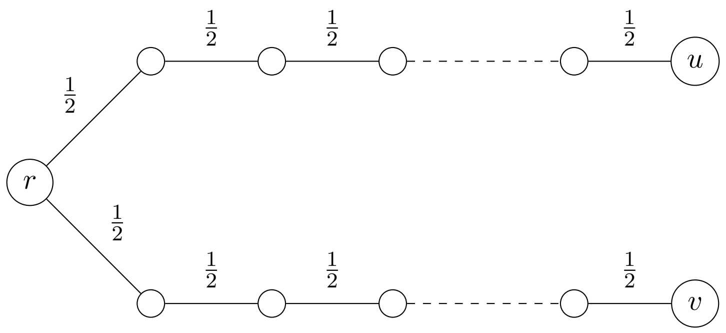
Notice that there are exponential number of constraints. However, we can still solve this LP on polynomial time by using the Ellipsoid algorithm with a separation oracle. Recall that for separation, we need to solve the following problem: given a (fractional) x, determine whether x is feasible or, if not, find a violated constraint. So for this LP, if we're given some x, we need to find whether there is a violated constraint: a set S with $r \in S$ and $g_i \cap S = \emptyset$ for some $i \in [k]$ so that $\sum_{e \in (S, \bar{S})} x_e < 1$. So we basically just need to find a min cut for each i! Slightly more formally, for each $i \in [k]$:

- Add terminal t_i adjacent to all nodes in g_i with edges of value 1.
- Compute the minimum $r - t_i$ cut by using any polynomial-time min-cut algorithm (e.g., Edmonds-Ford).
- If the minimum cut is less than 1, then we've found a violated constraint.

If for every i the minimum $s_i - t_i$ cut is at least 1, then all constraints are satisfied to x is feasible. Thus we have a polynomial time separation oracle, so can solve the LP in polynomial time.

It is also not hard to see based on max-flow min-cut that this is equivalent to an LP which requires us to send one unit of flow from each r to to t_i (the "fake" terminal adjacent to all of g_i). Then x_e variables then are interpreted as capacities, and we would have extra flow variables. This would give a compact formulation (which is how we would solve this in practice), but the cut-based interpretation is somewhat cleaner. But we will make use of this flow-based interpretation later.

Independent randomized rounding is not appropriate in this problem. Consider the following tree



where there are $\frac{n}{2} - 1$ nodes on both the $r - u$ and $r - v$ paths (not counting r, u, or v). Suppose that $g_1 = \{u, v\}$. Then if we sample each edge independently with probability equal to its LP value,

$$P(\text{connect } u \text{ to } r) = \frac{1}{2^{n/2}}$$

So if we did independent randomized rounding, we would have an extremely low probability of actually reaching a terminal from g_1. (This type of example can be extended easily to have a polynomial number of terminals in g_1 with all LP values being inverse polynomials, so simple inflation as in Set Cover will not work).

13.4.3 Rounding Algorithm

One quick definition and lemma:

Lemma 13.4.2 *Let $e \in E$, $p(e)$ be the parent edge of e (remember that G is a tree). Then in any optimal $\vec{x}$, $x_{p(e)} \geq x_e$.*

Proof: First, note that WLOG due to the Ford-Fulkerson proof of the max-cut min-flow theorem, we can assume that the minimum cut between r and each g_i is a set S which includes r and is connected. Suppose that $x_{p(e)} < x_e$. Let $e = \{v, w\}$ and let $p(e) = \{u, v\}$. Since x is optimal, we cannot reduce x_e while maintaining feasibility, so there is some cut S which cuts e, i.e., v (and thus u) are in S but w is not, so that $\sum_{e' \in (S,\bar{S})} x_{e'} = 1$. But then consider the cut $S' = S \setminus \{v\}$. Compared to S, in S' the edge $p(e)$ crosses the cut when it didn't before, and e (and possibly other edges) which did cross the cut no longer do. Thus $\sum_{e' \in (S',\bar{S}')} x_{e'} < 1$, which contradicts the feasibility of x. ■

Consider the following rounding algorithm:

Algorithm 1 GKR Rounding Algorithm for GST

```
for each $e \in E$ do
    Mark $e$ with probability $\frac{x_e}{x_{p(e)}}$. If $e$ is incident on $r$, then mark $e$ with probability $x_e$.
end for
Include $e$ if $e$ and all its ancestors are marked.
return $T$
```

Lemma 13.4.3 $\mathbf{Pr}$[*include* e] $= x_e$ *for all* $e \in E$.

Proof: Suppose e has i ancestors. Then

$$\begin{aligned}\mathbf{Pr}[e \text{ included}] &= \frac{x_e}{x_{p(e)}} \cdot \frac{x_{p(e)}}{x_{p^2(e)}} \cdot \frac{x_{p^2(e)}}{x_{p^3(e)}} \cdots \frac{x_{p^{i-1}(e)}}{x_{p^i(e)}} \cdot x_{p^i(e)} \\ &= x_e.\end{aligned}$$

■

Corollary 13.4.4 $\mathbf{E}[ALG] \leq LP$.

Proof:

$$\mathbf{E}[ALG] = \sum_{e \in E} c(e) \cdot \mathbf{E}[\mathbf{1}_{e \in ALG}] = \sum_{e \in E} c(e) \cdot x_e = LP.$$

■

Claim 13.4.5 *Using GKR rounding,* $\forall i \in [k]$,

$$\mathbf{Pr}[g_i \textit{ connected to } r] \geq \frac{1}{O(\log |g_i|)} \geq \frac{1}{O(\log n)}.$$

We will first show how to prove Theorem 13.4.1 by assuming **Claim 13.4.5**.

Proof of Theorem 13.4.1: First, suppose GKR rounding is run $\Theta(\log n \log k)$ times. Now fix some g_i and notice that

$$\mathbf{Pr}[g \text{ not connected to } r] \leq \left(1 - \frac{1}{O(\log|g|)}\right)^{\Theta(\log n \log k)} \leq e^{-\ln k} = \frac{1}{k},$$

(if we use appropriate constants in the asymptotic notation). Now for each $i \in [k]$, let P_i be the least expensive $r - g_i$ path. Then it is certainly true that $c(P_i) \leq OPT$. Now if g_i is not connected to r after the randomized rounding, then add P_i. Then notice that

$$\mathbf{E}[c(ALG)] \leq O(\log n \log k) \cdot LP + \sum_{i=1}^{k} \frac{1}{k} \cdot c(P_i) = O(\log n \log k) \cdot OPT.$$

This is to say that adding the shortest paths to the disconnected groups does not significantly hurt us because the probability that a group is disconnected is small. ■

The rest of these notes will be aimed at proving Claim 13.4.5. First we give a lemma that gives the general idea behind the proof. Let us fix some group g.

Definition 13.4.6 *Let FAIL be the event that g is not connected to r.*

Lemma 13.4.7 *If $x'_e \leq x_e$ for all $e \in E$, then*

$$\mathbf{Pr}\big[FAIL \text{ using } x'\big] \geq \mathbf{Pr}[FAIL \text{ using } x]$$

Proof: We prove this by considering one edge at a time and then using induction. So suppose that the only difference between x and x' is on one edge $e = \{v, u\}$ (with $v = p(u)$) where $x'_e < x_e$. Note that the probability of connecting every vertex outside of the subtree T rooted at u is exactly the same in x and in x', so we'll only need to worry about the subtree that goes through e.

Suppose for simplicity that there are two children edges $e_1 = \{u, w\}$ and $e_2 = \{u, z\}$ of u (the higher degree case has more complex math but everything works out basically the same). Let T_1 be the subtree rooted at the child node of e_1, and let T_2 be the subtree rooted at the child node of e_2. Using the fractional solution x, let

$$p_1 = \mathbf{Pr}[\text{fail to connect } T_1 \cap g \text{ to } w],$$
$$p_2 = \mathbf{Pr}[\text{fail to connect } T_2 \cap g \text{ to } z].$$

Then

$$\mathbf{Pr}[\text{fail to connect } T \cap g \text{ to } v] = (1 - x_e) + x_e \left(\left(1 - \frac{x_{e_1}}{x_e}\right) + \frac{x_{e_1}}{x_e} p_1\right)\left(\left(1 - \frac{x_{e_2}}{x_e}\right) + \frac{x_{e_2}}{x_e} p_2\right).$$

Here the first term is the probability that e is not picked, and then if e is picked we take the product of failing to connect T_1 to u and failing to connect T_2 to u (since to fail on T we would have to fail on both and they're independent). Each of these terms can be broken into the probability of not marking e_1 (or e_2) plus the probability of picking e_1 (or e_2) times to probability of failing in the tree below that (p_1 or p_2 respectively).

When we simplify this expression, we get that

$$\mathbf{Pr}[\text{fail to connect } T \cap g \text{ to } r] = 1 - x_{e_1}(1-p_1) - x_{e_2}(1-p_2) + \frac{x_{e_1}x_{e_2}(1-p_1)(1-p_2)}{x_e}.$$

This expression clearly increases as x_e decreases. Thus we are less likely to fail using x than we are using x'. ■

This lemma means that decreasing x values can't help us, so if $x'_e \le x_e$ for all $e \in E$ and probability of connecting g to r using x' is at least $\frac{1}{\log|g|}$, then the same is true using x (which is what we are trying to prove).

Now consider the following construction of x'.

1) Remove all leaves not in g and all unnecessary edges.

2) Reduce x values until minimally feasible (exactly one unit of flow is sent to g, or equivalently the min $r-g$ cut is equal to 1).

3) Round down to the next power of 2; now the flow is at least $\frac{1}{2}$ because all edges will be at least half of their original value (equivalently, the minimum $r-g$ cut is at least 1/2).

4) Delete all edges with $x_e \le \frac{1}{4|g|}$. Since there are at most $|g|$ leaves, the flow is still at least

$$\frac{1}{2} - |g| \cdot \frac{1}{4|g|} = \frac{1}{4}.$$

5) If $x_e = x_{p(e)}$, then contract e (since our rounding will mark e with probability 1 anyway).

Lemma 13.4.8 *The height of the tree is at most* $O(\log|g|)$

Proof: At each level, x values go down by at least a factor of 2 since we rounded to powers of 2 and contracted edges with the same value as their parent. Because of steps 2 and 4, we know that

$$\frac{1}{4|g|} \le x_e \le 1.$$

Hence the number of levels is at most $\log(4|g|) = O(\log|g|)$. ■

We now want to show that if we round using x', the probability we connect g to r is at least $\frac{1}{\log|g|}$. The natural way to do this would be to break this up into events corresponding to each terminal in g being connected to r, but unfortunately there's a lot of dependence among these events. So it's not at all clear how to analyze this. We'll do this next class using *Janson's Inequality.*

References

HK03 E. Halperin and R. Krauthgamer, Polylogarithmic Inapproximability. *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, 585-594, 2003.

GKR00 N. Garg, G. Konjevod, and R. Ravi, A polylogarithmic approximation algorithm for the group Steiner tree problem, *SODA* 2000.