Algorithms and Uncertainty, Summer 2020 Lecture 6 (5 pages)

Online Steiner Tree

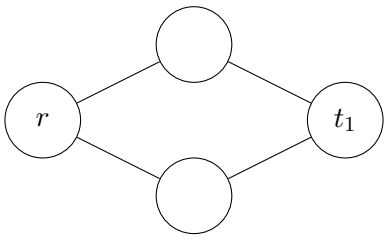
Thomas Kesselheim Last Update: May 6, 2020

Today, we will consider online versions of the Steiner tree problem. The problem will also reappear later this semester in a different setting of uncertainty. All of these use similar techniques, which we will introduce today.

Let us first state the classical offline Steiner tree problem. We state it here in a rooted variant because this makes our life easier in settings with uncertainty. We are given a connected graph $G = (V, E)$, edge weights $w(e) \geq 0$ for $e \in E$, a root $r \in V$, and a set of terminals $T \subseteq V$. Our task is to select a subset of the edges $S \subseteq E$ such that $\{r\} \cup T$ is connected in $G' = (V, S)$ and $\sum_{e \in S} w(e)$ is minimized. Observe that if $T = V$ then this problem is exactly the minimum spanning tree problem. It is an NP-hard problem.

In the online version, we only get to know the terminals in the set T one after the other. That is, $T = \{t_1, \ldots, t_k\}$ and in the i-th step, t_i is revealed. At any point in time, we are maintaining a set S such that S is a Steiner tree on $\{r\} \cup T$. We may only add but not remove edges from S. In other words, immediately when terminal t_i is revealed, we have to add edges to S such that $\{r, t_1, \ldots, t_i\}$ is connected in $G' = (V, S)$.

Example 6.1. *In the following example, all edge weights are* 1*. The first terminal has to be connected, either via the top or the bottom path. Both choice seem equally good at this point. However, in the following step, the top or the bottom vertex may become the second terminal and then it is either already connected or we have to add another edge.*



1 Simplifications

Observe that any edge $e = \{u, v\}$ in the Steiner tree can also be replaced by a path $u, x_1, \ldots, x_\ell, v$. This keeps the solution feasible. Indeed, we should never add an edge to the Steiner tree if the sum of edge weights on such a path is cheaper than the direct connection.

To simplify our life, we make this change implicit in the following. We assume that $G = (V, E)$ is a complete graph and we assume that the weights $w(e)$ fulfill the triangle inequality. That is, $w(\{u, v\}) \leq w(\{u, x\}) + w(\{x, v\})$ for all $u, v, x \in V$. Both assumptions are without loss of generality. In both cases, instead of using the (non-existing or more expensive) direct connection from u to v, we could take a shortest path from u to v.

2 Steiner Trees and Spanning Trees

As already mentioned, the spanning-tree problem is exactly the special case of the Steiner-tree problem in which all vertices are terminals. Indeed, Steiner trees can be approximated by minimum spanning trees. Such a spanning tree only uses edges between the nodes in the

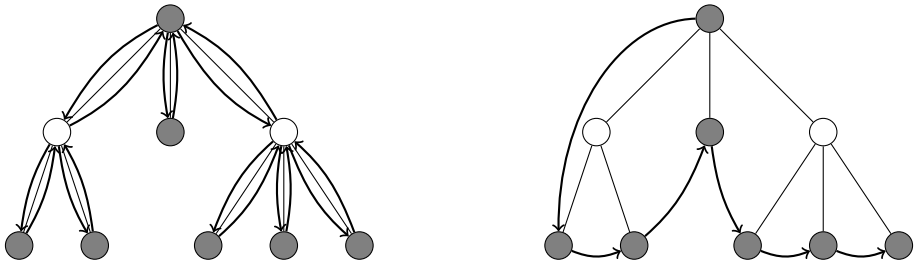


Figure 1: The idea of the proof of Lemma 6.2: Traverse the Steiner tree, then leave out Steiner vertices (white) and duplicate vertices.

set $\{r\} \cup T$ and no edges to other vertices (called Steiner vertices). Let $\text{MST}(T) \subseteq E$ be the minimum spanning tree on $G|_{\{r\}\cup T}$ and let $\text{Steiner}(T) \subseteq E$ be the optimal Steiner tree connecting $\{r\} \cup T$.

Lemma 6.2. *A minimum spanning tree on $G|_{\{r\}\cup T}$ is a 2-approximation for the min-cost Steiner tree on $\{r\} \cup T$, formally*

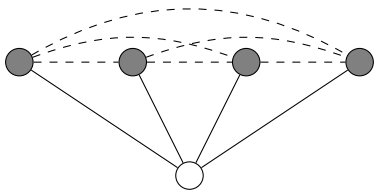
$$w(\text{MST}(T)) \leq 2 \cdot w(\text{Steiner}(T))$$

Proof. The idea is as follows: Traverse the optimal Steiner tree in a depth-first-search manner. You cross each edge twice: Once when entering the subtree and once when exiting it again. Equivalently, you can double each edge in the tree and consider an Euler tour through these duplicated tree edges. As each edge is crossed twice, the sum of edge costs on this run is $2 \cdot w(\text{Steiner}(T))$.

We get a sequence of vertices that contains r and each terminal from T at least once. Consider the path that shortcuts this sequence by only visiting r and the vertices in T exactly once. By triangle inequality, this path can only be shorter, so the sum of edge costs is at most $2 \cdot w(\text{Steiner}(T))$.

This path is a spanning tree of $G|_{\{r\}\cup T}$. The minimum spanning tree has at most its cost. □

Note that this bound is tight. Consider the following graph construction. The filled vertices are terminals. The solid edges have weight 1, the dashed ones weight 2.



The minimum spanning tree will only used the dashed edges and therefore have a cost of $2(k-1)$ whereas the optimal Steiner tree will use the solid edges and have a cost of k.

3 Greedy Algorithm for Online Steiner Tree

Our algorithm for Online Steiner Tree is the greedy algorithm that always connects an arriving terminal to the existing Steiner Tree via the cheapest edge. More formally: Initialize $S = \emptyset$. When t_i arrives, let e_i be the cheapest edge between t_i and one of the vertices $r, t_1, \ldots, t_{i-1}$. Add e_i to S.

Theorem 6.3. *Greedy is $O(\log k)$-competitive.*

For every terminal t_i there is a connection cost $w(e_i)$, which is the cost increase in the i-th step. To prove the theorem, we will use the following lemma, which shows that not all of the connection costs that we see can be very high.

Lemma 6.4. *The j-th highest connection cost among $w(e_1), \dots, w(e_k)$ is at most $\frac{w(\mathrm{MST}(T))}{j}$.*

Proof. Let $T_j \subseteq T$ be the set of terminals of size j for which the connection cost is highest. Our claim is now that $\min_{i \in T_j} w(e_i) \leq \frac{w(\mathrm{MST}(T))}{j}$.

Consider $\mathrm{MST}(T_j)$, that is, the minimum spanning tree on the graph induced by $\{r\} \cup T_j$. Because of the triangle inequality, this tree can only be cheaper than $\mathrm{MST}(T)$.

Furthermore, $\mathrm{MST}(T_j)$ contains exactly j edges. Therefore

$$\min_{e \in \mathrm{MST}(T_j)} w(e) \leq \frac{1}{j} \sum_{e \in \mathrm{MST}(T_j)} w(e) = \frac{w(\mathrm{MST}(T_j))}{j} \ .$$

Let e be the cheapest edge in $\mathrm{MST}(T_j)$. Defining $t_0 = r$, we can write $e = \{t_a, t_b\}$ for $a < b$. At the time t_b arrives, t_a is already a part of the Steiner tree. That is, t_b could be connected via edge e. The Greedy algorithm chooses the cheapest way to connect t_b. Therefore, $w(e_b) \leq w(e)$.

So, overall,

$$\min_{i \in T_j} w(e_i) \leq w(e_b) \leq w(e) \leq \frac{w(\mathrm{MST}(T_j))}{j} \leq \frac{w(\mathrm{MST}(T))}{j} \ . \qquad \square$$

Proof of Theorem 6.3. Because of Lemma 6.4, we can rewrite and bound the cost of the algorithm as

$$\sum_{i=1}^{k} w(e_i) \leq \sum_{j=1}^{k} \frac{w(\mathrm{MST}(T))}{j} = w(\mathrm{MST}(T)) \sum_{j=1}^{k} \frac{1}{j} \ .$$

Note that $\sum_{j=1}^{k} \frac{1}{j}$ is the k-th harmonic number. We can, for example, bound it by

$$\sum_{j=1}^{k} \frac{1}{j} \leq 1 + \int_1^k \frac{1}{x} \mathrm{d}x = 1 + \ln k \ .$$

So, the theorem now follows by Lemma 6.2. $\square$

4 A Probabilistic Input Model

The competitive ratio of $\Theta(\log k)$ is optimal for Online Steiner Tree. We skip the proof here. Instead, we will see how we can capture and use some prior information.

We assume that $t_1, \dots, t_k$ are random variables. They are drawn i.i.d. from a probability distribution, which is known to the algorithm in advance. That is, we have a vector of probabilities $(p_v)_{v \in V}$, where p_v denotes the probability that v arrives as a terminal in any step. We also assume that k is known.

We consider the following GreedyWithSample algorithm. Draw $t'_1, \dots t'_k$ from the probability distribution. Let $T' = \{t'_1, \dots t'_k\}$. Initialize $S = \mathrm{MST}(T')$. Now, like in the Greedy Algorithm, connect arriving new terminals to the current Steiner tree with the cheapest edge. That is, when t_i arrives, let e_i be the cheapest edge between t_i and one of the vertices $r, t'_1, \dots, t'_k, t_1, \dots, t_{i-1}$. Add e_i to S.

As we draw with replacement from our distribution, it may happen that a terminal appears multiple times in the sequence. In such a second or later appearance, we set $e_i = \{t_i\}$ with weight 0.

For this algorithm, we get a similar guarantee as in competitive analysis but now taken in expectation over all possible inputs from the distribution and not point-wise for every possible sequence.

Theorem 6.5. *For the final set S of edges chosen by* GreedyWithSample*, we have* $\mathbf{E}[w(S)] \leq 4 \cdot \mathbf{E}[w(\text{Steiner}(T)]$.

Proof. Our algorithm's cost consists of two components: The cost of $\text{MST}(T')$ and then the cost of $e_1, \ldots, e_k$.

Observe that T and T' are identically distributed. Therefore $\mathbf{E}[w(\text{MST}(T'))] = \mathbf{E}[w(\text{MST}(T))]$.

Next, we would like to bound $\mathbf{E}[w(e_i)]$. We consider $\text{MST}(\{t'_1, \ldots t'_{k-1}, t_i\})$ and let e'_i be defined as follows. If t_i is different from $t'_1, \ldots t'_{k-1}$, let e'_i be the edge connecting t_i towards the root in this tree. Otherwise, let $e'_i = \{t_i\}$. As e'_i would be a possible choice for e_i, we have $w(e_i) \leq w(e'_i)$ and it suffices to bound $\mathbf{E}[w(e'_i)]$. To this end, we determine $\{t'_1, \ldots t'_{k-1}, t_i\}$ in a different way. We first draw k times from the distribution, let the outcome be a multiset called U. Then draw one vertex from U uniformly at random and call it t_i. The remaining vertices are called $t'_1, \ldots t'_{k-1}$.

Note that $\text{MST}(\{t'_1, \ldots t'_{k-1}, t_i\}) = \text{MST}(U)$ does not depend on which vertex in U is t_i. Consider an edge $\{u_1, u_2\} \in \text{MST}(U)$, where u_2 is closer to the root. This edge becomes e'_i only if u_1 appears only once in U and u_1 is drawn to be t_i. So $\mathbf{Pr}[e'_i = \{u_1, u_2\} \mid U] \leq \frac{1}{k}$ because t_i is drawn uniformly from U.

So, we have

$$\mathbf{E}\left[w(e'_i) \;\middle|\; U\right] \leq \sum_{\{u_1,u_2\} \in \text{MST}(U)} \frac{1}{k} w(\{u_1, u_2\}) = \frac{1}{k} w(\text{MST}(U))$$

for any choice of U. Taking the expectation over U, we get

$$\mathbf{E}\left[w(e'_i)\right] = \sum_U \mathbf{Pr}[U \text{ is drawn}]\, \mathbf{E}\left[w(e'_i) \;\middle|\; U\right] \leq \sum_U \mathbf{Pr}[U \text{ is drawn}]\, \frac{1}{k} w(\text{MST}(U)) = \frac{1}{k}\mathbf{E}[w(\text{MST}(U))] \quad .$$

Combining all of these, we have

$$\mathbf{E}[w(e_i)] \leq \frac{1}{k}\mathbf{E}\left[w(\text{MST}(\{t'_1, \ldots t'_{k-1}, t_i\}))\right] \quad .$$

Finally, observe that $\{t'_1, \ldots t'_{k-1}, t_i\}$ is nothing but a set of k independently drawn terminals from the distribution. That is, it is distributed just the same way as T. Therefore $\mathbf{E}\left[w(\text{MST}(\{t'_1, \ldots t'_{k-1}, t_i\}))\right] = \mathbf{E}[w(\text{MST}(T)]$. So, we have

$$\mathbf{E}[w(e_i)] \leq \frac{1}{k}\mathbf{E}[w(\text{MST}(T)] \quad .$$

Overall, we get

$$\mathbf{E}\left[w(\text{MST}(T')) + \sum_{i=1}^{k} w(e_i)\right] = \mathbf{E}\left[w(\text{MST}(T'))\right] + \sum_{i=1}^{k} \mathbf{E}[w(e_i)] \leq 2\mathbf{E}[w(\text{MST}(T)] \quad .$$

The theorem now follows by Lemma 6.2. □

Reference

- Naveen Garg, Anupam Gupta, Stefano Leonardi, Piotr Sankowski: Stochastic analyses for online combinatorial optimization problems. SODA 2008 (Source of results from Section 4).